# Categorization of Social Actors in Social Network Analysis (SNA) using Representation Learning via Knowledge-Graph Embeddings and Convolution Operations (RLVECO)

Bonaventure C. Molokwu*
School of Computer Science - University of Windsor
Windsor, Ontario, Canada
molokwub@uwindsor.ca

Shaon Bhatta Shuvo*
School of Computer Science - University of Windsor
Windsor, Ontario, Canada
shuvos@uwindsor.ca

Ziad Kobti*
School of Computer Science - University of Windsor
Windsor, Ontario, Canada
kobti@uwindsor.ca

## ABSTRACT

Several activities, comprising animate and inanimate entities, can be examined by means of SNA. Classification tasks within social network structures remain crucial research problems in SNA. Inherent and latent facts about social graphs can be effectively exploited for training Artificial Intelligence (AI) models in a bid to categorize actors/nodes as well as identify clusters with respect to a given social network. Thus, important factors such as the individual attributes of spatial social actors and the underlying patterns of relationship binding these social actors must be taken into consideration. These factors are relevant to understanding the nature and dynamics of a given social graph. In this paper, we have proposed a hybrid model: RLVECO which has been modelled for studying and extracting meaningful facts from social network structures to aid in node classification and community detection problems. RLVECO utilizes an edge sampling approach for exploiting features of a social graph, via learning the context of each actor with respect to its neighboring actors, with the aim of generating vector-space embeddings per actor which are further exploited for unexpressed representations via a sequence of convolution operations. Successively, these relatively low-dimensional representations are fed as input features to a downstream classifier for solving community detection and node classification problems about a given social network.

## CCS CONCEPTS

• **Human-centered computing** → **Social network analysis**; • **Computing methodologies** → **Artificial intelligence**; **Machine learning**; **Neural networks**.

*Both authors contributed equally to this research.

*DLP-KDD 2020, August 24, 2020, San Diego, California, USA*
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06. . . $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

## KEYWORDS

Node Classification, Feature Learning, Feature Extraction, Dimensionality Reduction, Semi-supervised Learning, Deep Learning

**ACM Reference Format:**
Bonaventure C. Molokwu, Shaon Bhatta Shuvo, and Ziad Kobti. 2020. Categorization of Social Actors in SNA using RLVECO. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION AND RELATED LITERATURE

Humans inhabit a planet comprised of several systems and ecosystems; and interaction is a natural phenomenon and characteristic obtainable in any given system or ecosystem. Thus, relationship between constituent entities in a given system/ecosystem is a strategy for survival, and essential for the sustenance of the system/ecosystem. With respect to the recent advances in AI, real-world (complex) systems and ecosystems can be effectively represented as social network structures and analyzed by means of SNA. Furthermore, SNA plays a pivotal role especially in the modelling and impact analysis of pandemic outbreaks like Corona Virus Disease 2019 (COVID-19). Techniques for categorizing/classifying social actors in SNA can be implemented for predicting potential actors/nodes/entities as well as communities or clusters that are likely to be affected and/or infected by the COVID-19 virus as a result of social interaction and/or community spread within a given social network structure. Social (network) graphs [23] are non-static structures which pose analytical challenges to Machine Learning (ML) and Deep Learning (DL) models because of their complex links, random nature, and occasionally massive size. In this regard, we propose RLVECO which is a hybrid DL-based model for classification and clustering problems in social networks.

Actor/Node classification and community detection remain open research problems in SNA. The act of categorizing actors induces the formation of cluster(s). Consequently, clusters give rise to homophily in social networks. Herein our proposed methodology is based on an iterative learning approach [1] which is targeted at solving the problems of node classification and community detection using an edge sampling strategy. Basically, learning in RLVECO is induced via semi-supervised training; and RLVECO is capable of learning the non-linear distributed features enmeshed in a social network [9]. Hence, the novelty of our research contribution are stated below:

(1) Proposition of a DL-based and hybrid model, RLVECO, designed for resolving tie or link prediction problems in social network structures.
(2) Comprehensive benchmarking results which are based on classic objective functions used for standard classifiers.
(3) Comparative analyses, between RLVECO and state-of-the-art methodologies, against standard real-world social networks.

Also, we have evaluated RLVECO against an array of state-of-the-art models and Representation Learning (RL) approaches which serve as our baselines, viz:

(i) DeepWalk: Online Learning of Social Representations [20].
(ii) GCN: Semi-Supervised Classification with Graph Convolutional Networks [11].
(iii) LINE: Large-scale Information Network Embedding [26].
(iv) Node2Vec: Scalable Feature Learning for Networks [8].
(v) SDNE: Structural Deep Network Embedding [27].

## 2 PROPOSED METHODOLOGY AND FRAMEWORK

### 2.1 Definition of Problem

Definition 2.1. *Social Network, $SN$: As expressed via equation 1 such that $SN$ is a tuple comprising a set of actors/vertices, $V$; a set of ties/edges, $E$; a metadata function, $f_V$, which extends the definition of the vertices' set by mapping it to a given set of attributes, $V'$; and a metadata function, $f_E$, which extends the definition of the edges' set by mapping it to a given set of attributes, $E'$. Thus, a graph function, $G(V,E) \subset SN$.*

$$
\begin{aligned}
SN &= \{V, E, f_V, f_E\} \equiv \{G, f_V, f_E\} \\
V &: |\{V\}| = M \quad \text{set of actors/vertices with size, M} \\
E &: E \subset \{U \times V\} \subset \{V \times V\} \text{ set of ties/edges between V} \\
f_V &: V \rightarrow V' \quad \text{vertices' metadata function} \\
f_E &: E \rightarrow E' \quad \text{edges' metadata function}
\end{aligned} \tag{1}
$$

Definition 2.2. *Knowledge Graph, $KG$: $(\mathbb{E}, \mathbb{R})$ is a set comprising entities, $\mathbb{E}$, and relations, $\mathbb{R}$, between the entities. Thus, a KG [25][30] is defined via a set of triples, $t : (u, p, v)$, where $u, v \in \mathbb{E}$ and $p \in \mathbb{R}$. Also, a KG [28] can be modelled as a social network, $SN$, such that: $\mathbb{E} \rightarrow V$ and $\mathbb{R} \rightarrow E$ and $(\mathbb{E}, \mathbb{R}) \vdash f_V, f_E$.*

Definition 2.3. *Knowledge-Graph (Vector) Embeddings, $X$: The vector-space embeddings, $X$, generated by the embedding layer are based on a mapping function, $f$, expressed via equation 2. $f$ projects the representation of the graph's actors to a $q$-dimensional real space, $\mathbb{R}^q$, such that the existent ties between any given pair of actors, $(u_i, v_j)$, remain preserved via the homomorphism from $V$ to $X$.*

$$
\begin{aligned}
&f : V \rightarrow X \in \mathbb{R}^q \\
&f : (u, p, v) \rightarrow X \in \mathbb{R}^q \quad \text{Knowledge-Graph Embeddings}
\end{aligned} \tag{2}
$$

Definition 2.4. *Node Classification: Considering, $SN$, comprising partially labelled actors (or vertices), $V_{lbl} \subset V : V_{lbl} \rightarrow Y_{lbl}$; and unlabelled vertices defined such that: $V_{ulb} = V - V_{lbl}$. A node-classification model aims at training a predictive function, $f : V \rightarrow Y$, that learns to predict the labels, $Y$, for all actors or vertices, $V \subset SN$, via knowledge harnessed from the mapping: $V_{lbl} \rightarrow Y_{lbl}$.*

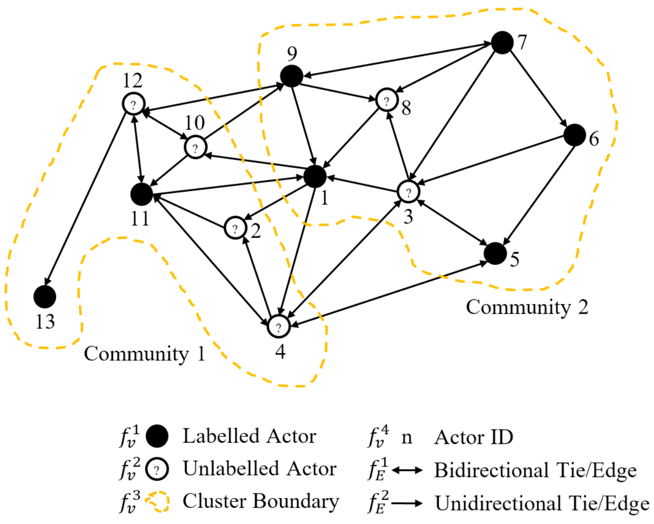

**Figure 1: Node classification task in social graphs**

### 2.2 Proposed Methodology

Our proposition, RLVECO, is comprised of two (2) distinct Feature Learning (FL) layers, and one (1) classification layer.

#### 2.2.1 *Representation Learning - Knowledge-Graph Embeddings Layer:*

Given a social network, $SN$, defined by a set of actors/vertices, $V : U \subset V \forall \{u_m, v_m\} \in V$, and $M : m \in M$ denotes the number of unique actors in $SN$. Additionally, let the ties/edges in $SN$ be defined such that: $E \subset \{U \times V\}$; where $u_i \in V$ and $v_j \in V$ represent a source_vertex and a target_vertex in $E$, respectively.

The objective function of the Knowledge-Graph Embeddings layer aims at maximizing the average logarithmic probability of the source_vertex, $u_i$, being predicted as a neighboring actor to the target_vertex, $v_j$, with respect to all training pairs, $\forall (u_i, v_j) \in E$. Formally, the function is expressed as in equation 3:

$$\mu = \frac{1}{M}\sum_{m=1}^{M}(\sum_{(u_i,v_j)\in E} logPr(u_i|v_j)) \tag{3}$$

Consequently, in order to compute $Pr(u_i|v_j)$, we have to quantify the proximity of each target_vertex, $v_j$, with respect to its source_vertex, $u_i$. The vector-embedding model measures this adjacency/proximity as the cosine similarity between $v_j$ and its corresponding $u_i$. Thus, the cosine distance is calculated as the dot product between the target_vertex and the source_vertex. Mathematically, $Pr(u_i|v_j)$ is computed via a softmax function as defined in equation 4:

$$Pr(u_i|v_j) = \frac{exp(u_i \cdot v_j)}{\sum_{m=1}^{M} exp(u_m \cdot v_j)} \tag{4}$$

Hence, the objective function of our vector-embedding (VE) model with respect to the $SN$ is as expressed by equation 5:

$$\sum_{(u_i,v_j)\in E} logPr(u_i|v_j) = \sum_{(u_i,v_j)\in E} log\frac{exp(u_i \cdot v_j)}{\sum_{m=1}^{M} exp(u_m \cdot v_j)} \tag{5}$$

*2.2.2 Representation Learning - Convolution Operations Layer:* This layer comprises three (3) RL or FL operations, namely: convolution, non-linearity, and pooling operations. RLVECO utilizes a one-dimensional (1D) convolution-operations layer [17] which is sandwiched between the vector-embedding and classification layers. Equation 6 expresses the 1D-convolution operation, viz:

$$FeatureMap(F) = 1D\_InputMatrix(X) * Kernel(K)$$

$$f_i = (X*K)_i = (K*X)_i = \sum_{j=0}^{J-1} x_j \cdot k_{i-j} = \sum_{j=0}^{J-1} k_j \cdot x_{i-j} \tag{6}$$

where $f_i$ represents a cell/matrix position in the Feature Map; $k_j$ denotes a cell position in the Kernel; and $x_{i-j}$ denotes a cell/matrix position in the 1D-Input (data) matrix.

The non-linearity operation is a rectified linear unit (ReLU) function which introduces non-linearity after the convolution operation since real-world problems usually exist in non-linear form(s). As a result, the rectified feature/activation map is computed via: $r_i \in R = g(f_i \in F) = max(0, F)$.

The pooling operation is responsible for reducing the input width of each rectified activation map while retaining its vital properties. In this regard, the *Max Pooling* function is defined such that the resultant pooled (or downsampled) feature map is generated via: $p_i \in P = h(r_i \in R) = maxPool(R)$.

*2.2.3 Classification - Multi-Layer Perceptron (MLP) Classifier Layer:* This is the last layer of RLVECO's architecture, and it succeeds the representation-learning layers. The pooled feature maps, generated by the representation-learning layers, contain high-level features extracted from the constituent actors in the social graph. Hence, the classification layer utilizes these extracted "high-level features" for classifying actors in a bid to identify clusters contained in the social graph. The objective of the MLP [5] classifier function, $f_c$, is to map a given set of input values, $P$, to their respective output labels, $Y$, viz:

$$Y = f_c(P, \Theta) \tag{7}$$

In equation 7, $\Theta$ denotes a set of parameters. The MLP [4] function, $f_c$, learns the values of $\Theta$ that will result in the best decision ($Y$) approximation for the input set, $P$. The MLP classifier output is a probability distribution which indicates the likelihood of a representation belonging to a particular output class. Our MLP [10] classifier is modelled such that sequential layers of Neural Network (NN) units are stacked against each other to form a Deep Neural Network (DNN) structure [3], [15].

*2.2.4 Node Classification - Proposed Algorithm:* RLVECO's architecture comprises two (2) discrete representation-learning layers, viz: a Knowledge-Graph Embeddings (VE) layer and a Convolution Operations (CO) layer [16]; which are both trained by means of unsupervised training. Basically, these layers are feature-extraction and dimensionality-reduction layers where underlying knowledge and viable facts are automatically extracted from the social network structure. The VE layer is responsible for projecting the feature representation of the social graph to a q-dimensional real-number space, $\mathbb{R}^q$. This is done by associating a real-number vector to every unique actor/node in the social network structure such that the (cosine) distance of any given tie or edge would capture a significant degree of correlation between its pair of associated actors. Furthermore, the Convolution Operations layer feeds on the Knowledge-Graph Embeddings layer; and it is responsible for further extraction of apparent features and/or representations from the social graph. Finally, a classification layer succeeds the representation-learning layers; and it is trained by means of supervised training. The classifier is based on a NN architecture assembled using deep (multi) layers of stacked perceptrons (NN units) [6]. Every low-dimensional feature ($X$), extracted by the representation-learning layers, is mapped to a corresponding output label ($Y$). These ($X, Y$) pairs are used to supervise the training of the classifier such that it can effectively/efficiently learn how to identify clusters and classify actors within a given social graph. Ergo, RLVECO's is formally expressed algorithm 1.

### 2.3 Proposed Architecture/Framework

Fig. 2 illustrates the architecture of our proposition, RLVECO.

## 3 DATASETS AND MATERIALS

### 3.1 Datasets

With regard to Table 1 herein, seven (7) real-world and benchmark social-graph datasets were utilized for experimentation and evaluation, viz: CiteSeer [24] [21], Cora [24] [21], Facebook Page-Page webgraph [22], Internet-Industry partnerships [12] [13] [2], PubMed-Diabetes [18], Terrorists-Relationship [31], and Zachary-Karate [29] [14].

**Algorithm 1** Proposed Node Classification Algorithm

```
Input: $\{V, E, Y_{lbl}\} \equiv$ {Actors, Ties, Ground-Truth Labels}

Output: $\{Y_{ulb}\} \equiv$ {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)

// $|E_{train}| = \sum indegree(V_{lbl}) + \sum outdegree(V_{lbl})$
$E_{train} = E_t : u_i, v_j \in V_{lbl}$
$E_{pred} = E_p : u_i, v_j \in V_{ulb}$
$f_c \leftarrow$ Initialize    // Construct classifier model

Training:
for $t \leftarrow 0$ to $|E_{train}|$ do
    $f : E_t \rightarrow [X \in \mathbb{R}^q]$    // Embedding operation

    $f_t \in F = (K*X)_t$    // Convolution operation

    $r_t \in R = g(F) = max(0, f_t)$
    $p_t \in P = h(R) = maxPool(r_t)$
    $f_c|\Theta : p_t \rightarrow Y_{lbl}$    // MLP classification operation
end for

return $Y_{ulb} = f_c(E_{pred}, \Theta)$
```

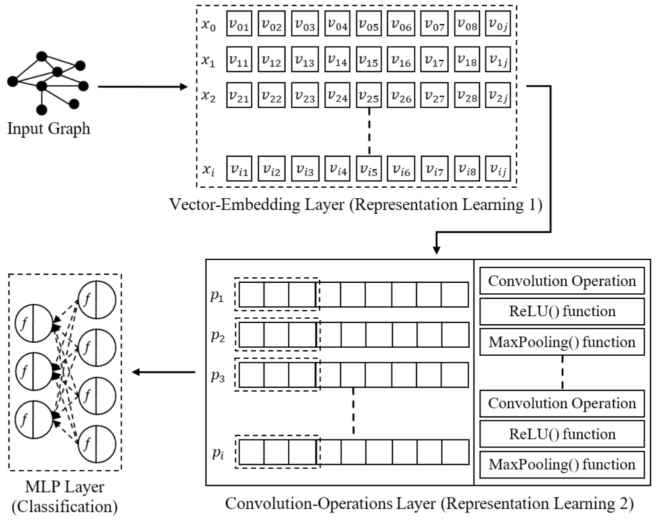
Input Graph
Vector-Embedding Layer (Representation Learning 1)
$x_0$ $x_1$ $x_2$ $x_i$
$v_{01}$ $v_{02}$ $v_{03}$ $v_{04}$ $v_{05}$ $v_{06}$ $v_{07}$ $v_{08}$ $v_{0j}$
$v_{11}$ $v_{12}$ $v_{13}$ $v_{14}$ $v_{15}$ $v_{16}$ $v_{17}$ $v_{18}$ $v_{1j}$
$v_{21}$ $v_{22}$ $v_{23}$ $v_{24}$ $v_{25}$ $v_{26}$ $v_{27}$ $v_{28}$ $v_{2j}$
$v_{i1}$ $v_{i2}$ $v_{i3}$ $v_{i4}$ $v_{i5}$ $v_{i6}$ $v_{i7}$ $v_{i8}$ $v_{ij}$
$p_1$ $p_2$ $p_3$ $p_i$
Convolution Operation
ReLU() function
MaxPooling() function
Convolution Operation
ReLU() function
MaxPooling() function
Convolution-Operations Layer (Representation Learning 2)
f
MLP Layer (Classification)

**Figure 2: Proposed architectural framework of RLVECO**

## 3.2 Data Preprocessing

All benchmark datasets ought to be comprised of actors and ties already encoded as discrete data (whole-number format). However, CiteSeer, Cora, Facebook-Page2Page, PubMed-Diabetes, and Terrorists-Relation datasets are made up of nodes and/or edges encoded in mixed formats (categorical and numerical formats). Thus, it is necessary to transcode these non-numeric (categorical) entities to their respective discrete (numeric) data representation, without semantic loss, via an injective function that maps each distinct entry in the categorical-entity domain to a distinct numeric value in the discrete-data codomain, $f_m : categorical \rightarrow discrete$. Thereafter, the numeric representation of all benchmark datasets are normalized, $f_n : discrete \rightarrow continuous$, prior to training against RLVECO and the baselines. Also, only edgelist ties, $E(U, V)$, whose constituent actors are present in the nodelist, $(U, V) : \forall \{u_m, v_m\} \in V \subset G$, were used for training/testing/validating our model.

**Table 1: Benchmark datasets**

| Dataset | Classes $\rightarrow$ {label: 'description'} |
|---|---|
| CiteSeer | $G(V, E) = G(3312, 4732)$ {C1: 'Agents', C2: 'Artificial Intelligence', C3: 'Databases', C4: 'Information Retrieval', C5: 'Machine Learning', C6: 'Human-Computer Interaction'} |
| Cora | $G(V, E) = G(2708, 5429)$ {C1: 'Case_Based', C2: 'Genetic_Algorithms', C3: 'Neural_Networks', C4: 'Probabilistic_Methods', C5: 'Reinforcement_Learning', C6: 'Rule_Learning', C7: 'Theory'} |
| Facebook Page2Page | $G(V, E) = G(22470, 171002)$ {C1: 'Companies', C2: 'Governmental Organizations', C3: 'Politicians', C4: 'Television Shows'} |
| Internet Industry | $G(V, E) = G(219, 631)$ {C1: 'Content Sector', C2: 'Infrastructure Sector', C3: 'Commerce Sector'} |
| PubMed Diabetes | $G(V, E) = G(19717, 44338)$ {C1: 'Diabetes Mellitus - Experimental', C2: 'Diabetes Mellitus - Type 1', C3: 'Diabetes Mellitus - Type 2'} |
| Terrorists Relation | $G(V, E) = G(851, 8592)$ {C1: 'Content Sector', C2: 'Infrastructure Sector', C3: 'Commerce Sector'} |
| Zachary Karate | $G(V, E) = G(34, 78)$ {C1: 'Community 1', C2: 'Community 2', C3: 'Community 3', C4: 'Community 4'} |

**Table 2: Configuration of RLVECO's hyperparameters**

| | |
|---|---|
| Training Set: 80% | Network Depth: 6 |
| Test Set: 20% | Network Width: 640 |
| Batch Size: 256 | Optimizer: $AdaMax$ |
| Epochs: $1.8 * 10^2$ | Activation: $ReLU$ |
| Dropout: $4.0 * 10^{-1}$ | Embed Dimension: 100 |
| Learning Rate: $1.0 * 10^{-3}$ | Learning Decay: 0.0 |

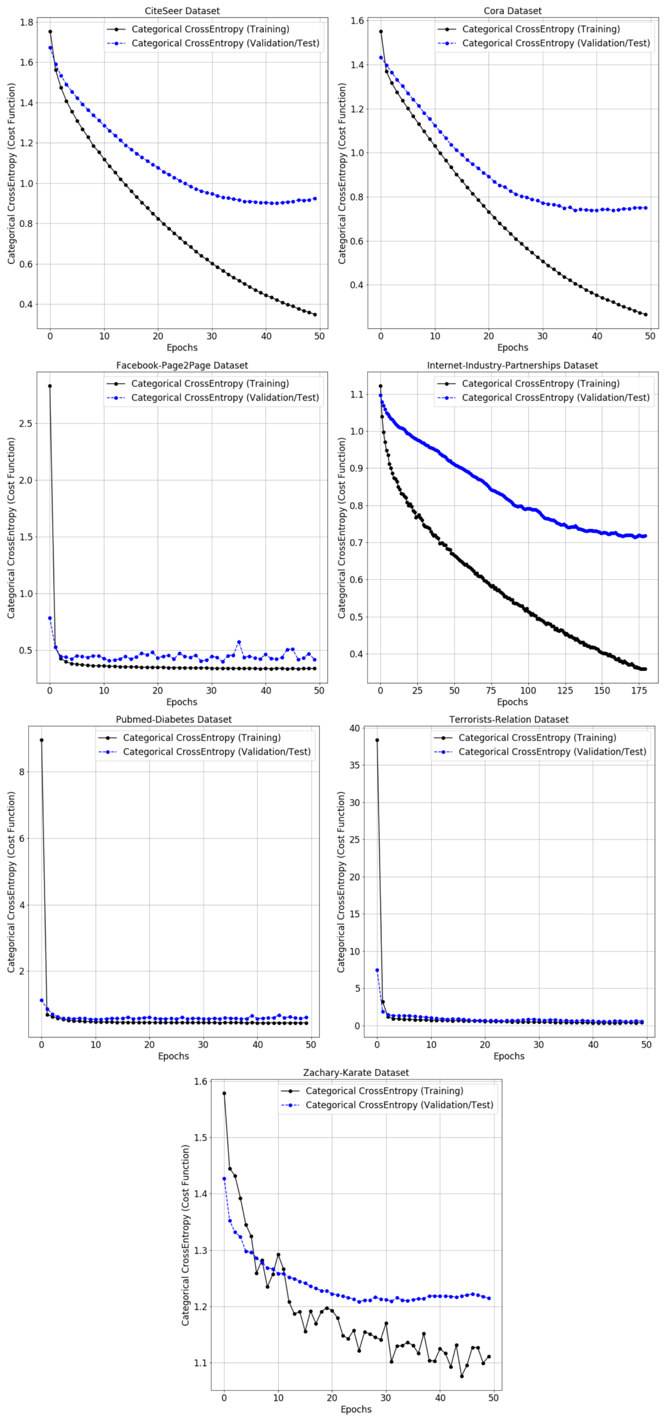

**Figure 3: RLVECO's learning-progress curves of during training over CiteSeer, Cora, Facebook-Webgraph, Internet-Industry-Partnership, PubMed-Diabetes, Terrorists-Relationship, and Zachary-Karate datasets - loss function vs training epochs.**

**Table 3: Categorization of actors using CiteSeer dataset with regard to the set apart validation sample - dataset *vs* models.**

| Model | Metric | CiteSeer Dataset C1 | C2 | C3 | C4 | C5 | C6 | $\mu$ | Points |
|---|---|---|---|---|---|---|---|---|---|
| RLVECO | PC | 0.76 | 0.81 | 0.78 | 0.43 | 0.88 | 0.60 | 0.71 | 12 |
| | RC | 0.84 | 0.83 | 0.79 | 0.60 | 0.79 | 0.65 | 0.75 | |
| | F1 | **0.80** | **0.82** | **0.79** | **0.50** | **0.83** | **0.63** | 0.73 | |
| | AC | 0.93 | 0.88 | 0.92 | 0.93 | 0.96 | 0.89 | 0.92 | |
| | RO | **0.90** | **0.87** | **0.87** | **0.78** | **0.89** | **0.79** | 0.85 | |
| | SP | 304 | 609 | 377 | 107 | 225 | 275 | 316 | |
| GCN | PC | 0.80 | 0.78 | 0.86 | 0.95 | 0.91 | 0.75 | 0.84 | 2 |
| | RC | 0.76 | 0.76 | 0.73 | 0.08 | 0.67 | 0.54 | 0.59 | |
| | F1 | 0.78 | 0.77 | **0.79** | 0.15 | 0.77 | **0.63** | 0.65 | |
| | AC | 0.88 | 0.87 | 0.88 | 0.91 | 0.89 | 0.83 | 0.88 | |
| | RO | 0.84 | 0.83 | 0.83 | 0.53 | 0.81 | 0.72 | 0.76 | |
| | SP | 119 | 134 | 140 | 50 | 102 | 118 | 111 | |
| Node2Vec | PC | 0.57 | 0.55 | 0.49 | 0.33 | 0.55 | 0.38 | 0.48 | 0 |
| | RC | 0.55 | 0.60 | 0.66 | 0.06 | 0.45 | 0.40 | 0.45 | |
| | F1 | 0.56 | 0.58 | 0.56 | 0.10 | 0.50 | 0.39 | 0.45 | |
| | AC | 0.85 | 0.82 | 0.78 | 0.92 | 0.86 | 0.78 | 0.84 | |
| | RO | 0.73 | 0.74 | 0.74 | 0.53 | 0.69 | 0.63 | 0.68 | |
| | SP | 119 | 134 | 140 | 50 | 102 | 118 | 111 | |
| DeepWalk | PC | 0.46 | 0.53 | 0.43 | 0.43 | 0.47 | 0.33 | 0.44 | 0 |
| | RC | 0.51 | 0.54 | 0.57 | 0.06 | 0.41 | 0.32 | 0.40 | |
| | F1 | 0.49 | 0.54 | 0.49 | 0.11 | 0.44 | 0.32 | 0.40 | |
| | AC | 0.81 | 0.81 | 0.75 | 0.92 | 0.84 | 0.76 | 0.82 | |
| | RO | 0.69 | 0.71 | 0.69 | 0.53 | 0.66 | 0.59 | 0.65 | |
| | SP | 119 | 134 | 140 | 50 | 102 | 118 | 111 | |
| SDNE | PC | 0.37 | 0.50 | 0.24 | 0.20 | 0.45 | 0.31 | 0.35 | 0 |
| | RC | 0.19 | 0.27 | 0.77 | 0.02 | 0.14 | 0.09 | 0.25 | |
| | F1 | 0.25 | 0.35 | 0.36 | 0.04 | 0.21 | 0.14 | 0.23 | |
| | AC | 0.80 | 0.80 | 0.42 | 0.92 | 0.84 | 0.80 | 0.76 | |
| | RO | 0.56 | 0.60 | 0.55 | 0.51 | 0.55 | 0.52 | 0.55 | |
| | SP | 119 | 134 | 140 | 50 | 102 | 118 | 111 | |
| LINE | PC | 0.18 | 0.30 | 0.28 | 0.60 | 0.22 | 0.27 | 0.31 | 0 |
| | RC | 0.15 | 0.47 | 0.39 | 0.06 | 0.12 | 0.21 | 0.23 | |
| | F1 | 0.16 | 0.36 | 0.32 | 0.11 | 0.15 | 0.24 | 0.22 | |
| | AC | 0.72 | 0.67 | 0.65 | 0.93 | 0.80 | 0.76 | 0.76 | |
| | RO | 0.50 | 0.59 | 0.56 | 0.53 | 0.52 | 0.55 | 0.54 | |
| | SP | 119 | 134 | 140 | 50 | 102 | 118 | 111 | |

## 4 EXPERIMENT, RESULTS, AND DISCUSSIONS

RLVECO has been tuned in accordance with the hyperparameters shown in Table 2. Our evaluations herein were recorded with reference to a range of objective functions. Thus, Categorical Cross Entropy was employed as the cost/loss function; while the fitness/utility was measured based on the following metrics: Precision (PC), Recall (RC), F-measure or F1-score (F1), Accuracy (AC), and Area Under the Receiver Operating Characteristic Curve (RO). Moreover, the objective functions have been computed against each benchmark dataset with regard to the constituent classes (or categories) present in each dataset. The Support (SP) represents the number of ground-truth samples per class/category contained in each dataset.

In a bid to avoid sample bias across-the-board, we have used exactly the same SP for all models inclusive of RLVECO.

**Table 4: Categorization of actors using Cora dataset with respect to the set apart validation sample - dataset *vs* models.**

| Model | Metric | Cora Dataset |  |  |  |  |  |  | | Points |
|---|---|---|---|---|---|---|---|---|---|---|
|  |  | C1 | C2 | C3 | C4 | C5 | C6 | C7 | $\mu$ |  |
| RLVECO | PC | 0.85 | 0.78 | 0.80 | 0.88 | 0.72 | 0.90 | 0.81 | 0.82 | 14 |
|  | RC | 0.86 | 0.93 | 0.81 | 0.87 | 0.75 | 0.91 | 0.78 | 0.84 |  |
|  | F1 | **0.86** | **0.85** | **0.81** | **0.87** | **0.74** | **0.91** | **0.79** | 0.83 |  |
|  | AC | 0.93 | 0.98 | 0.96 | 0.95 | 0.93 | 0.97 | 0.96 | 0.95 |  |
|  | RO | **0.90** | **0.96** | **0.90** | **0.92** | **0.85** | **0.95** | **0.88** | 0.91 |  |
|  | SP | 541 | 134 | 214 | 405 | 294 | 345 | 237 | 310 |  |
| GCN | PC | 0.87 | 0.95 | 0.89 | 0.92 | 0.85 | 0.89 | 0.87 | 0.89 | 3 |
|  | RC | 0.85 | 0.73 | 0.65 | 0.82 | 0.58 | 0.85 | 0.73 | 0.74 |  |
|  | F1 | **0.86** | 0.83 | 0.75 | **0.87** | 0.69 | 0.87 | **0.79** | 0.81 |  |
|  | AC | 0.89 | 0.93 | 0.91 | 0.92 | 0.88 | 0.93 | 0.91 | 0.91 |  |
|  | RO | 0.88 | 0.83 | 0.80 | 0.89 | 0.75 | 0.90 | 0.83 | 0.84 |  |
|  | SP | 164 | 36 | 43 | 85 | 70 | 84 | 60 | 77 |  |
| Node2Vec | PC | 0.58 | 0.78 | 0.72 | 0.81 | 0.80 | 0.84 | 0.82 | 0.76 | 0 |
|  | RC | 0.85 | 0.50 | 0.53 | 0.68 | 0.64 | 0.74 | 0.60 | 0.65 |  |
|  | F1 | 0.69 | 0.61 | 0.61 | 0.74 | 0.71 | 0.78 | 0.69 | 0.69 |  |
|  | AC | 0.77 | 0.96 | 0.95 | 0.92 | 0.93 | 0.94 | 0.94 | 0.92 |  |
|  | RO | 0.79 | 0.75 | 0.76 | 0.83 | 0.81 | 0.86 | 0.79 | 0.80 |  |
|  | SP | 164 | 36 | 43 | 85 | 70 | 84 | 60 | 77 |  |
| DeepWalk | PC | 0.57 | 0.58 | 0.72 | 0.58 | 0.68 | 0.72 | 0.63 | 0.64 | 0 |
|  | RC | 0.80 | 0.42 | 0.42 | 0.59 | 0.39 | 0.63 | 0.65 | 0.56 |  |
|  | F1 | 0.67 | 0.48 | 0.53 | 0.58 | 0.49 | 0.67 | 0.64 | 0.58 |  |
|  | AC | 0.76 | 0.94 | 0.94 | 0.87 | 0.90 | 0.90 | 0.92 | 0.89 |  |
|  | RO | 0.77 | 0.70 | 0.70 | 0.75 | 0.68 | 0.79 | 0.80 | 0.74 |  |
|  | SP | 164 | 36 | 43 | 85 | 70 | 84 | 60 | 77 |  |
| LINE | PC | 0.35 | 0.86 | 0.80 | 0.65 | 0.50 | 0.43 | 0.61 | 0.60 | 0 |
|  | RC | 0.85 | 0.17 | 0.19 | 0.35 | 0.20 | 0.15 | 0.23 | 0.31 |  |
|  | F1 | 0.50 | 0.28 | 0.30 | 0.46 | 0.29 | 0.23 | 0.34 | 0.34 |  |
|  | AC | 0.48 | 0.94 | 0.93 | 0.87 | 0.87 | 0.84 | 0.90 | 0.83 |  |
|  | RO | 0.59 | 0.58 | 0.59 | 0.66 | 0.59 | 0.56 | 0.61 | 0.60 |  |
|  | SP | 164 | 36 | 43 | 85 | 70 | 84 | 60 | 77 |  |
| SDNE | PC | 0.37 | 0.83 | 0.70 | 0.60 | 0.54 | 0.64 | 0.64 | 0.62 | 0 |
|  | RC | 0.91 | 0.14 | 0.16 | 0.35 | 0.20 | 0.27 | 0.12 | 0.31 |  |
|  | F1 | 0.53 | 0.24 | 0.26 | 0.44 | 0.29 | 0.38 | 0.20 | 0.33 |  |
|  | AC | 0.50 | 0.94 | 0.93 | 0.86 | 0.87 | 0.86 | 0.89 | 0.84 |  |
|  | RO | 0.62 | 0.57 | 0.58 | 0.65 | 0.59 | 0.62 | 0.55 | 0.60 |  |
|  | SP | 164 | 36 | 43 | 85 | 70 | 84 | 60 | 77 |  |

Since RLVECO is based on an edge-sampling technique; the SP recorded against RLVECO model represents the number of edges/ties used for computation as explained in algorithm 1. However, for other baselines (models) herein, SP capitalizes on the number of nodes/actors. Furthermore, the performance of RLVECO model during comparative analyses with respect to five (5) popular baselines (DeepWalk, GCN, LINE, Node2Vec, SDNE); and when evaluated against the validation/test samples of the benchmark datasets are as documented in Table 3, Table 4, Table 5, Table 6, Table 7, Table 8, and Table 9 respectively. Consequently, Fig. 3 graphically shows the learning-progress curves of our proposed model, RLVECO, during training over the benchmark datasets. Hence, the dotted-black lines represent learning progress over the training set; and the dotted-blue lines represent learning progress over the test set.

Tables 3, 4, 5, 6, 7, 8, and 9 have clearly tabulated our results as a multi-classification task over the benchmark

**Table 5: Categorization of actors/nodes using Facebook Page-Page webgraph dataset with respect to the reserved validation/test sample - dataset *vs* models.**

| Model | Metric | Facebook-Page2Page Dataset |  |  |  |  | Points |
|---|---|---|---|---|---|---|---|
|  |  | C1 | C2 | C3 | C4 | $\mu$ |  |
| RLVECO | PC | 0.87 | 0.95 | 0.91 | 0.87 | 0.90 | 8 |
|  | RC | 0.84 | 0.85 | 0.85 | 0.86 | 0.85 |  |
|  | F1 | **0.85** | **0.90** | **0.88** | **0.86** | 0.87 |  |
|  | AC | 0.96 | 0.90 | 0.94 | 0.97 | 0.94 |  |
|  | RO | **0.97** | **0.97** | **0.98** | **0.98** | 0.98 |  |
|  | SP | 9989 | 33962 | 16214 | 6609 | 16694 |  |
| Node2Vec | PC | 0.81 | 0.84 | 0.81 | 0.84 | 0.83 | 0 |
|  | RC | 0.82 | 0.87 | 0.85 | 0.67 | 0.80 |  |
|  | F1 | 0.81 | 0.85 | 0.83 | 0.74 | 0.81 |  |
|  | AC | 0.89 | 0.91 | 0.91 | 0.93 | 0.91 |  |
|  | RO | 0.87 | 0.90 | 0.89 | 0.82 | 0.87 |  |
|  | SP | 1299 | 1376 | 1154 | 665 | 1124 |  |
| DeepWalk | PC | 0.75 | 0.84 | 0.76 | 0.75 | 0.78 | 0 |
|  | RC | 0.81 | 0.85 | 0.82 | 0.52 | 0.75 |  |
|  | F1 | 0.78 | 0.84 | 0.79 | 0.62 | 0.76 |  |
|  | AC | 0.87 | 0.90 | 0.89 | 0.90 | 0.89 |  |
|  | RO | 0.85 | 0.89 | 0.87 | 0.75 | 0.84 |  |
|  | SP | 1299 | 1376 | 1154 | 665 | 1124 |  |
| LINE | PC | 0.53 | 0.66 | 0.72 | 0.66 | 0.64 | 0 |
|  | RC | 0.72 | 0.71 | 0.59 | 0.29 | 0.58 |  |
|  | F1 | 0.61 | 0.68 | 0.65 | 0.40 | 0.59 |  |
|  | AC | 0.73 | 0.80 | 0.83 | 0.87 | 0.81 |  |
|  | RO | 0.73 | 0.77 | 0.75 | 0.63 | 0.72 |  |
|  | SP | 1299 | 1376 | 1154 | 665 | 1124 |  |
| SDNE | PC | 0.49 | 0.80 | 0.70 | 0.65 | 0.66 | 0 |
|  | RC | 0.90 | 0.63 | 0.50 | 0.19 | 0.56 |  |
|  | F1 | 0.64 | 0.70 | 0.58 | 0.29 | 0.55 |  |
|  | AC | 0.70 | 0.84 | 0.82 | 0.86 | 0.81 |  |
|  | RO | 0.76 | 0.78 | 0.71 | 0.58 | 0.71 |  |
|  | SP | 1299 | 1376 | 1154 | 665 | 1124 |  |

datasets. Thus, for each class per dataset, we have laid emphasis on the F1 (weighted average of the PC and RC metrics) and RO; and we have highlighted the model which performed best (based on F1 and RO metrics) for each classification task using a **bold font**. Additionally, we have employed a point-based ranking standard to ascertain the fittest model for each node classification task. The model with the best mean ($\mu$) metrics and highest aggregate points signifies the fittest model for the specified task, and so on in a descending order of mean ($\mu$) metrics and aggregate points. Accordingly, as can be seen from our tabular results, RLVECO is at the top with the highest fitness points; and this encouraging performance can be attributed to two (2) primary factors, namely:

(1) The dual layers of Representation Learning (VE and Convolutional Neural Network (ConvNet)) in RLVECO's conceptual model.
(2) The high-quality data preprocessing techniques employed herein with respect to the benchmark datasets. We ensured that all constituent actors of a given social graph were transcoded to their respective discrete data

**Table 6: Categorization of actors/nodes using Internet-Industry partnerships dataset with respect to the set apart validation/test sample - dataset *vs* models.**

| Model | Metric | Internet-Industry Partnerships | | | | Points |
|---|---|---|---|---|---|---|
| | | C1 | C2 | C3 | $\mu$ | |
| RLVECO | PC | 0.33 | 0.96 | 0.29 | 0.53 | 5 |
| | RC | 0.65 | 0.77 | 0.76 | 0.73 | |
| | F1 | **0.44** | **0.86** | 0.42 | 0.57 | |
| | AC | 0.84 | 0.78 | 0.87 | 0.83 | |
| | RO | **0.76** | **0.81** | **0.82** | 0.80 | |
| | SP | 26 | 238 | 17 | 94 | |
| DeepWalk | PC | 0.50 | 0.81 | 0.36 | 0.56 | 1 |
| | RC | 0.12 | 0.93 | 0.44 | 0.50 | |
| | F1 | 0.20 | **0.86** | 0.40 | 0.49 | |
| | AC | 0.82 | 0.82 | 0.73 | 0.79 | |
| | RO | 0.55 | 0.79 | 0.62 | 0.65 | |
| | SP | 8 | 27 | 9 | 15 | |
| Node2Vec | PC | 0.00 | 0.68 | 0.75 | 0.48 | 1 |
| | RC | 0.00 | 1.00 | 0.33 | 0.44 | |
| | F1 | 0.00 | 0.81 | **0.46** | 0.42 | |
| | AC | 0.82 | 0.70 | 0.84 | 0.79 | |
| | RO | 0.50 | 0.62 | 0.65 | 0.59 | |
| | SP | 8 | 27 | 9 | 15 | |
| SDNE | PC | 0.00 | 0.61 | 0.00 | 0.20 | 0 |
| | RC | 0.00 | 1.00 | 0.00 | 0.33 | |
| | F1 | 0.00 | 0.76 | 0.00 | 0.25 | |
| | AC | 0.82 | 0.61 | 0.80 | 0.74 | |
| | RO | 0.50 | 0.50 | 0.50 | 0.50 | |
| | SP | 8 | 27 | 9 | 15 | |
| LINE | PC | 0.00 | 0.61 | 0.00 | 0.20 | 0 |
| | RC | 0.00 | 1.00 | 0.00 | 0.33 | |
| | F1 | 0.00 | 0.76 | 0.00 | 0.25 | |
| | AC | 0.82 | 0.61 | 0.80 | 0.74 | |
| | RO | 0.50 | 0.50 | 0.50 | 0.50 | |
| | SP | 8 | 27 | 9 | 15 | |

representations, without any loss in semantics, and normalized prior to training/testing/validation.

Since we have implemented a deep-layer architecture [10] [4], there arises the need to strike a balance between the width and depth of RLVECO's proposed architecture so as to obtain an efficient/effective model without increasing the chances of underfitting or overfitting in the NN model [7]. Thus, we have applied pruning of neuron(s) based on equation 8 as well as global search methods. $N_s, N_i, N_o$, and $N_m$ represent the sizes of training set, input layer, output layer, and hidden layer respectively.

$$N_m = \frac{N_s}{4 * (N_i + N_o)} \tag{8}$$

Dropout regularization has been implemented within the hidden layers of RLVECO. Also, $L2$ regularization ($L2 = 0.04$) [7] and early stopping (shown via Table 10) [19] were employed herein as addon regularization techniques to overcome overfitting incurred during the training of RLVECO. A mini-batch size of 256 was used for training, testing, and validating because we want to ensure that sufficient patterns are extracted by the model during training before its network weights are updated.

**Table 7: Categorization of actors using PubMed-Diabetes dataset based on the reserved test sample - dataset *vs* models.**

| Model | Metric | PubMed-Diabetes Dataset | | | | Points |
|---|---|---|---|---|---|---|
| | | C1 | C2 | C3 | $\mu$ | |
| RLVECO | PC | 0.76 | 0.83 | 0.84 | 0.81 | 6 |
| | RC | 0.60 | 0.88 | 0.91 | 0.80 | |
| | F1 | **0.67** | **0.86** | **0.87** | 0.80 | |
| | AC | 0.89 | 0.88 | 0.90 | 0.89 | |
| | RO | **0.92** | **0.94** | **0.95** | 0.94 | |
| | SP | 3300 | 7715 | 7170 | 6062 | |
| DeepWalk | PC | 0.65 | 0.57 | 0.58 | 0.60 | 0 |
| | RC | 0.15 | 0.67 | 0.71 | 0.51 | |
| | F1 | 0.24 | 0.62 | 0.63 | 0.50 | |
| | AC | 0.81 | 0.67 | 0.68 | 0.72 | |
| | RO | 0.56 | 0.67 | 0.69 | 0.64 | |
| | SP | 821 | 1575 | 1548 | 1315 | |
| Node2Vec | PC | 0.74 | 0.47 | 0.49 | 0.57 | 0 |
| | RC | 0.03 | 0.65 | 0.55 | 0.41 | |
| | F1 | 0.05 | 0.55 | 0.52 | 0.37 | |
| | AC | 0.80 | 0.57 | 0.60 | 0.66 | |
| | RO | 0.51 | 0.58 | 0.59 | 0.56 | |
| | SP | 821 | 1575 | 1548 | 1315 | |
| SDNE | PC | 0.65 | 0.43 | 0.74 | 0.61 | 0 |
| | RC | 0.05 | 0.96 | 0.17 | 0.39 | |
| | F1 | 0.10 | 0.59 | 0.27 | 0.32 | |
| | AC | 0.80 | 0.48 | 0.65 | 0.64 | |
| | RO | 0.52 | 0.56 | 0.56 | 0.55 | |
| | SP | 821 | 1575 | 1548 | 1315 | |
| LINE | PC | 0.48 | 0.42 | 0.44 | 0.45 | 0 |
| | RC | 0.05 | 0.60 | 0.46 | 0.37 | |
| | F1 | 0.08 | 0.50 | 0.45 | 0.34 | |
| | AC | 0.79 | 0.51 | 0.56 | 0.62 | |
| | RO | 0.52 | 0.53 | 0.54 | 0.53 | |
| | SP | 821 | 1575 | 1548 | 1315 | |

**Table 10: Early-stopping regularization against datasets**

| Dataset | Early stopping |
|---|---|
| CiteSeer, Cora, Facebook Page2Page, PubMed Diabetes, Terrorists Relation, Zachary Karate | after 50 epochs |
| Internet-Industry Partnerships | after 180 epochs |

## 5 LIMITATIONS AND CONCLUSION

The benchmark models (baselines) evaluated herein were executed using their default parameters. We were not able to evaluate GCN [11] against Facebook-Page2Page, PubMed-Diabetes, Internet-Industry-Partnership, and Zachary-Karate datasets; because these aforementioned datasets do not possess individual vector-based feature set which is required by the GCN model for input-data processing. Overall, RLVECO's remarkable performance with respect to the benchmarking results herein is primarily attributed to the presence of a biform RL/FL kernel.

**Table 8: Categorization of actors using Terrorists-Relationship dataset based on the reserved test sample - dataset *vs* models.**

| Model | Metric | Terrorists-Relation Dataset | | | | | Points |
|---|---|---|---|---|---|---|---|
| | | C1 | C2 | C3 | C4 | $\mu$ | |
| RLVECO | PC | 0.93 | 0.91 | 0.46 | 1.00 | 0.83 | 6 |
| | RC | 0.97 | 0.97 | 0.42 | 0.97 | 0.83 | |
| | F1 | **0.95** | **0.94** | 0.44 | **0.98** | 0.83 | |
| | AC | 0.95 | 0.98 | 0.89 | 0.99 | 0.95 | |
| | RO | **0.98** | **1.00** | 0.85 | **1.00** | 0.96 | |
| | SP | 1706 | 491 | 319 | 561 | 769 | |
| GCN | PC | 0.94 | 0.74 | 0.67 | 0.96 | 0.83 | 5 |
| | RC | 0.90 | 0.95 | 0.60 | 1.00 | 0.86 | |
| | F1 | 0.92 | 0.83 | **0.63** | **0.98** | 0.84 | |
| | AC | 0.92 | 0.95 | 0.88 | 0.99 | 0.94 | |
| | RO | **0.98** | 0.99 | **0.91** | **1.00** | 0.97 | |
| | SP | 92 | 21 | 30 | 27 | 43 | |
| DeepWalk | PC | 0.88 | 0.82 | 0.64 | 0.86 | 0.80 | 0 |
| | RC | 0.90 | 0.86 | 0.53 | 0.93 | 0.81 | |
| | F1 | 0.89 | 0.84 | 0.58 | 0.89 | 0.80 | |
| | AC | 0.88 | 0.96 | 0.86 | 0.96 | 0.92 | |
| | RO | 0.88 | 0.92 | 0.73 | 0.95 | 0.87 | |
| | SP | 92 | 21 | 30 | 27 | 43 | |
| Node2Vec | PC | 0.86 | 0.82 | 0.60 | 0.86 | 0.79 | 0 |
| | RC | 0.88 | 0.86 | 0.50 | 0.93 | 0.79 | |
| | F1 | 0.87 | 0.84 | 0.55 | 0.89 | 0.79 | |
| | AC | 0.86 | 0.96 | 0.85 | 0.96 | 0.91 | |
| | RO | 0.86 | 0.92 | 0.71 | 0.95 | 0.86 | |
| | SP | 92 | 21 | 30 | 27 | 43 | |
| LINE | PC | 0.82 | 0.82 | 0.58 | 0.92 | 0.79 | 0 |
| | RC | 0.92 | 0.86 | 0.37 | 0.85 | 0.75 | |
| | F1 | 0.87 | 0.84 | 0.45 | 0.88 | 0.76 | |
| | AC | 0.85 | 0.96 | 0.84 | 0.96 | 0.90 | |
| | RO | 0.84 | 0.92 | 0.65 | 0.92 | 0.83 | |
| | SP | 92 | 21 | 30 | 27 | 43 | |
| SDNE | PC | 0.77 | 0.90 | 0.56 | 1.00 | 0.81 | 0 |
| | RC | 0.92 | 0.86 | 0.30 | 0.85 | 0.73 | |
| | F1 | 0.84 | 0.88 | 0.39 | 0.92 | 0.76 | |
| | AC | 0.81 | 0.97 | 0.84 | 0.98 | 0.90 | |
| | RO | 0.80 | 0.92 | 0.62 | 0.93 | 0.82 | |
| | SP | 92 | 21 | 30 | 27 | 43 | |

The source codes for the classification of social actors using RLVECO proposed herein can be reviewed via Microsoft's GitHub software development version control platform. Thus, this can be publicly accessed via: https://github.com/bhevencious/RLVECN. Additionally, Table 11 herein describes the organization of the code repository.

## 6 FUTURE WORK

We intend to expand RLVECO's scope such that it can be applied for resolving other open research problems in SNA. Also, we are sourcing for additional baselines (benchmark models) and real-world social network datasets for extensive validation of RLVECO.

## ACKNOWLEDGMENTS

This research was supported by International Business Machines (IBM) via the provision of computational resources necessary to carry-out our experiments. Research was conducted

**Table 9: Categorization of actors using Zachary-Karate dataset based on the reserved test sample - dataset *vs* models.**

| Model | Metric | Zachary-Karate Dataset | | | | | Points |
|---|---|---|---|---|---|---|---|
| | | C1 | C2 | C3 | C4 | $\mu$ | |
| RLVECO | PC | 1.00 | 0.67 | 0.20 | 1.00 | 0.72 | 7 |
| | RC | 1.00 | 0.89 | 1.00 | 0.50 | 0.85 | |
| | F1 | **1.00** | **0.76** | **0.33** | 0.67 | 0.69 | |
| | AC | 1.00 | 0.81 | 0.69 | 0.77 | 0.82 | |
| | RO | **1.00** | **0.83** | **0.83** | **0.75** | 0.85 | |
| | SP | 3 | 9 | 2 | 12 | 7 | |
| SDNE | PC | 0.00 | 0.50 | 0.00 | 0.60 | 0.28 | 2 |
| | RC | 0.00 | 0.50 | 0.00 | 1.00 | 0.38 | |
| | F1 | 0.00 | 0.50 | 0.00 | **0.75** | 0.31 | |
| | AC | 0.86 | 0.71 | 0.86 | 0.71 | 0.79 | |
| | RO | 0.50 | 0.65 | 0.50 | **0.75** | 0.60 | |
| | SP | 1 | 2 | 1 | 3 | 2 | |
| LINE | PC | 0.00 | 0.50 | 0.00 | 0.60 | 0.28 | 2 |
| | RC | 0.00 | 0.50 | 0.00 | 1.00 | 0.38 | |
| | F1 | 0.00 | 0.50 | 0.00 | **0.75** | 0.31 | |
| | AC | 0.86 | 0.71 | 0.86 | 0.71 | 0.79 | |
| | RO | 0.50 | 0.65 | 0.50 | **0.75** | 0.60 | |
| | SP | 1 | 2 | 1 | 3 | 2 | |
| DeepWalk | PC | 0.00 | 0.40 | 0.00 | 0.50 | 0.23 | 0 |
| | RC | 0.00 | 1.00 | 0.00 | 0.33 | 0.33 | |
| | F1 | 0.00 | 0.57 | 0.00 | 0.40 | 0.24 | |
| | AC | 0.86 | 0.57 | 0.86 | 0.57 | 0.72 | |
| | RO | 0.50 | 0.70 | 0.50 | 0.54 | 0.56 | |
| | SP | 1 | 2 | 1 | 3 | 2 | |
| Node2Vec | PC | 0.00 | 0.00 | 0.00 | 0.25 | 0.06 | 0 |
| | RC | 0.00 | 0.00 | 0.00 | 0.33 | 0.08 | |
| | F1 | 0.00 | 0.00 | 0.00 | 0.29 | 0.07 | |
| | AC | 0.86 | 0.29 | 0.86 | 0.29 | 0.58 | |
| | RO | 0.50 | 0.20 | 0.50 | 0.29 | 0.37 | |
| | SP | 1 | 2 | 1 | 3 | 2 | |

**Table 11: Description of source-code repository**

| Subject | GitHub link/url |
|---|---|
| Home | https://github.com/bhevencious?tab=repositories |
| RLVECO for Node Classification | https://github.com/bhevencious/rlveco/blob/master/rlveco-node_classification.py |
| RLVECO's experiment results | https://github.com/bhevencious/rlveco/blob/master/eval_log.txt |
| Node-Classification Baselines (DeepWalk, LINE, Node2Vec, and SDNE) | https://github.com/bhevencious/Baselines_GraphEmbedding/blob/master/fused_baseline_models.py |
| DeepWalk, LINE, Node2Vec, and SDNE experiment results | https://github.com/bhevencious/Baselines_GraphEmbedding/blob/master/eval_log.txt |
| Node-Classification Baseline (GCN) | https://github.com/bhevencious/Baselines_GraphEmbedding/blob/master/kipf_gcn/gcnn_node_classification.py |
| GCN experiment results | https://github.com/bhevencious/Baselines_GraphEmbedding/blob/master/kipf_gcn/eval_log.txt |

on a high performance IBM Power System S822LC Linux Server. Consecutively, this research was supported jointly by SHARCNET and Compute Canada (www.computecanada.ca).

## REFERENCES

[1] Charu C. Aggarwal (Ed.). 2011. *Social Network Data Analytics.* Springer Science+Business Media, Boston, MA.

[2] Vladimir Batagelj, Patrick Doreian, Anuska Ferligoj, and Natasa Kejzar (Eds.). 2014. *Understanding Large Temporal Networks and Spatial Networks: Exploration, Pattern Searching, Visualization and Network Evolution.* John Wiley & Sons, Inc., Hoboken, NJ.

[3] Yoshua Bengio. 2009. Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning* 2 (2009), 1–113.

[4] Lei Min Deng and Dong H. Yu. 2014. *Deep Learning: Methods and Applications.* Now Publishers. https://books.google.ca/books?id=-Sa6xQEACAAJ

[5] Ian G. Goodfellow, Yoshua Bengio, and Aaron C. Courville. 2015. Deep Learning. *Nature* 521 (2015), 436–444.

[6] Ian G. Goodfellow, Yoshua Bengio, and Aaron C. Courville (Eds.). 2017. *Deep Learning.* MIT Press, Cambridge, MA.

[7] Aurlien Gron (Ed.). 2017. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly Media, Inc., Newton, MA.

[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 2016 (2016), 855–864.

[9] Geoffrey E. Hinton. 2007. Learning Multiple Layers of Representation. *TRENDS in Cognitive Sciences* 11, 10 (2007), 428–433.

[10] Geoffrey E. Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew W. Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. 2012. Deep Neural Networks for Acoustic Modeling in Speech Recognition. *IEEE Signal Processing Magazine* 29 (2012), 82–97.

[11] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. *International Conference on Learning Representations (ICLR)* abs/1609.02907 (2017).

[12] Valdis Krebs. 2002. Orgnet LLC. http://www.orgnet.com/netindustry.html.

[13] Valdis E. Krebs. 2008. Organizational Adaptability Quotient. In *IBM Global Services.*

[14] Jérôme Kunegis. 2013. KONECT: the Koblenz network collection. In *Proceedings of the 22nd International Conference on World Wide Web.* http://konect.cc/

[15] Bonaventure C. Molokwu and Ziad Kobti. 2019. Spatial Event Prediction via Multivariate Time Series Analysis of Neighboring Social Units using Deep Neural Networks. *2019 International Joint Conference on Neural Networks, IJCNN* (2019), 1–8.

[16] Bonaventure C. Molokwu and Ziad Kobti. 2020. Social Network Analysis using RLVECN: Representation Learning via Knowledge-Graph Embeddings and Convolutional Neural-Network. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence, IJCAI.*

[17] Bonaventure C. Molokwu, Shaon Bhatta Shuvo, Narayan C. Kar, and Ziad Kobti. 2020. Node Classification in Complex Social Graphs via Knowledge-Graph Embeddings and Convolutional Neural Network. *Computational Science – ICCS 2020* 12142 (2020), 183 – 198.

[18] Galileo Namata, Ben London, Lise Getoor, and Bert Huang. 2012. Query-driven Active Surveying for Collective Classification. In *Proceedings of the Workshop on Mining and Learning with Graphs, MLG-2012.*

[19] Josh Patterson and Adam Gibson (Eds.). 2017. *Deep Learning: A Practitioner's Approach.* O'Reilly Media, Inc., Newton, MA.

[20] Bryan Perozzi, Rami Al-Rfou', and Steven Skiena. 2014. DeepWalk: online learning of social representations. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* abs/1403.6652 (2014).

[21] Ryan A. Rossi and Nesreen K. Ahmed. 2015. The Network Data Repository with Interactive Graph Analytics and Visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence.* http://networkrepository.com

[22] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. 2019. Multi-scale Attributed Node Embedding. *ArXiv* abs/1909.13021 (2019).

[23] J. Scott (Ed.). 2017. *Social Network Analysis.* SAGE Publications Ltd, Newbury Park, CA.

[24] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective Classification in Network Data. *AI Magazine* 29 (2008), 93–106.

[25] Pedro Tabacof and Luca Costabello. 2020. Probability Calibration for Knowledge Graph Embedding Models. *International Conference on Learning Representations (ICLR)* abs/1912.10000 (2020).

[26] Jian Tang, Meng Qu, Mingzhe Wang, Mingjie Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web.*

[27] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural Deep Network Embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

[28] Shihui Yang, Jidong Tian, Honglun Zhang, Junchi Yan, Hao He, and Yaohui Jin. 2019. TransMS: Knowledge Graph Embedding for Complex Relations by Multidirectional Semantics. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI.*

[29] Wayne W. Zachary. 1977. An Information Flow Model for Conflict and Fission in Small Groups1. *Journal of anthropological research* 33 (11 1977). https://doi.org/10.1086/jar.33.4.3629752

[30] Qingheng Zhang, Zequn Sun, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019. Multi-view Knowledge Graph Embedding for Entity Alignment. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI,* Vol. abs/1906.02390.

[31] Bin Zhao, Prithviraj Sen, and Lise Getoor. 2006. Entity and Relationship Labeling in Affiliation Networks. In *Proceedings of the 23rd International Conference on Machine Learning, ICML.*