Event Prediction in Complex Social Graphs using One-Dimensional Convolutional Neural Network

Bonaventure Molokwu
School of Computer Science, University of Windsor, 401 Sunset Avenue, Ontario N9B-3P4, Canada
molokwub@uwindsor.ca

Abstract

Social network graphs possess apparent and latent knowledge about their respective actors and links which may be exploited, using effective and efficient techniques, for predicting events within the social graphs. Understanding the intrinsic relationship patterns among spatial social actors and their respective properties are crucial factors to be taken into consideration in event prediction within social networks. My research work proposes a unique approach for predicting events in social networks by learning the context of each actor/vertex using neighboring actors in a given social graph with the goal of generating vector-space embeddings for each vertex. Our methodology introduces a pre-convolution layer which is essentially a set of feature-extraction operations aimed at reducing the graph's dimensionality to aid knowledge extraction from its complex structure. Consequently, the low-dimensional node embeddings are introduced as input features to a one-dimensional ConvNet model for event prediction about the given social graph. Training and evaluation of this proposed approach have been done on datasets (compiled: November, 2017) extracted from real world social networks with respect to 3 European countries. Each dataset comprises an average of 280,000 links and 48,000 actors.

1 Introduction

A social network consists of finite set(s) of actors, and the relationship(s) defined between these actors [Scott, 2017]. Analyzing and learning intrinsic knowledge from communities, comprising social actors, using given sets of standard still remains a significant research problem in social network analysis (SNA). Furthermore, an event prediction problem can be expressed as a Satisfiability problem [Hans van Maaren and Walsh, 2009] such that an event is said to exist if the variables governing the event's formal definition reduces it to *true*. Hence, the Cook-Levin Theorem [Cook, 1971] has proven that Satisfiability problem is NP-Complete. The proposed methodologies herein are based on a neural network architecture assembled using deep layers of stacked processing units comprising ConvNet and MLP units. This architecture makes it feasible to develop and train neural network models to be capable of learning the nonlinear distributed representations enmeshed in the graph structures [Ian Goodfellow and Courville, 2017].

2 Problem Statement

Social network graphs are characterized by their complex size and dynamic nature; and this makes it relatively challenging and difficult to develop effective machine learning (ML) as well as deep learning (DL) models which can be trained to predict events over a given network graph with respect to its constituent vertices (or actors) and edges (or relationships).

3 Proposed Methodology

My proposition employs a Skip-gram neural-network model in the pre-convolution layer; and this Skip-gram model is responsible for unsupervised representation (or feature) learning where apparent features and viable facts (in the form of node embeddings) are automatically extracted from the complex graph data. In turn, these learnt node embeddings serve as input to the 1D-ConvNet classification layer. Thus, the 1D-ConvNet model is trained effectively upon the node embeddings with respect to its corresponding event labels using a supervised learning approach. Formally, a social network, SN, can be defined as in expression 1 where SN is a tuple defined such that it comprises a set of vertices: V; a set of edges: E; a metadata function: f_V which extends the definition of the vertices' set by mapping it to a given set of attributes: M; and a metadata function: f_E which extends the definition of the edges' set by mapping it to a given set of attributes: N.

$$\begin{aligned} SN &= (V, E, f_V, f_E) \\ G &: V, E \\ f_V &: V \rightarrow M \\ f_E &: E \rightarrow N \end{aligned} \tag{1}$$

Skip-gram neural network is a technique majorly employed in the domain of Natural Language Processing (NLP). Thus, given a large collection of text (text corpus); the Skip-gram model focuses on learning the low-dimensional features which can be used to effectively and efficiently represent each word in the text corpus [Jason Eisner and Poliak, 2017][Tomas Mikolov and Dean, 2013] in relation to a predefined words' vocabulary, $W : \forall\, w_m \in W$ where $M : m \in M$

is the number of unique words in the vocabulary. Given a target_word, w_t, within the text corpus; we define the "context" of w_t as the words surrounding it in a given size-L window within the text corpus.

$$\begin{aligned}\text{Text Corpus} &= w_{t-L-2}, ..., w_{t-L}, ..., w_t, ..., w_{t+L}, ..., w_N \\ \text{Leftward context of } w_t &= w_{t-L}, ..., w_{t-1} \\ \text{Rightward context of } w_t &= w_{t+1}, ..., w_{t+L}\end{aligned} \tag{2}$$

Thereafter, a collection of (target_word, context_word) pairs which we denote as D is generated to be used for training.

$$\forall\,(w_t, w_{t+l}) \in D \qquad l \in L\text{: window size of the context} \tag{3}$$

The goal of the Skip-gram model is to maximize the average logarithmic probability of the context_words, w_{t+L}, being predicted as contexts for the target_word, w_t, with respect to all training pairs, $\forall\,(w_t, w_{t+l}) \in D$. Formally, it is defined as:

$$\sum_{(w_t,w_{t+l})} logP(w_{t+l}|w_t) = \frac{1}{N}\sum_{n=1}^{N}\Big(\sum_{-L\leq l\leq L} logP(w_{t+l}|w_t)\Big) \tag{4}$$

To compute $P(w_{t+l}|w_t)$, we have to quantify the proximity of each target_word, w_t, with respect to its context_word, w_{t+l}. The Skip-gram model measures this proximity as the cosine distance between w_t and its corresponding w_{t+l}. Hence, every word comprising the text corpus with respect to W is encoded over a real number space, $\mathbb{R}$, such that $\forall\, w_t, w_{t+l}$:

$$\begin{aligned}&f_1 : w_t \to v_t && v_t \in \mathbb{R}\text{: target\_word vector} \\ &f_2 : w_{t+l} \to u_c && u_c \in \mathbb{R}\text{: context\_word vector} \\ &f_3 : w_m \to u_m && u_m \in \mathbb{R}\text{: } m_{th} \text{ word vector in } W\end{aligned} \tag{5}$$

The cosine distance is calculated as the dot product between the vector representations of the target_word and the context_word. Mathematically, $P(w_{t+l}|w_t)$ is computed as:

$$P(w_{t+l}|w_t) = P(u_c|v_t) = \frac{exp(u_c \cdot v_t)}{\sum_{m=1}^{M} exp(u_m \cdot v_t)} \tag{6}$$

Furthermore, extending this NLP methodology to graph theory, given a social network, SN, as defined by expression 1 above; the edge list, $E[i,j] \subset G$, which is a sequence of tuples is defined via equation 7. (u_i, v_j) denotes a link or tie from a source vertex, u_i, to a target vertex, v_j.

$$\begin{aligned}E[i,j] &:= \{(u_i, v_j)...(u_{i+m}, v_{j+n})\} \\ \forall\, u_i, v_j &\in \{V : v_0, v_2, ..., v_{n-1}\}\end{aligned} \tag{7}$$

Consequently, expression 8 defines the functions which map the graph domain, G, to the words' vocabulary, W.

$$\begin{aligned}&f_4 : G \to W \\ &f_5 : (u_i, v_j) \to (u_c, v_t) \\ &f_6 : u_m \to u_m\end{aligned} \tag{8}$$

where M signifies the number of unique nodes in the graph's set of vertices, V, such that: $\forall\, u_m \in V$. Therefore, the objective function of our Skip-gram layer with respect to a given graph, G, is as expressed by equation 9, viz:

$$\sum_{(u_i,v_j)\in E} logP(u_i|v_j) = \sum_{-L\leq l\leq L:l\neq 0} log\frac{exp(u_i \cdot v_j)}{\sum_{m=1}^{M} exp(u_m \cdot v_j)} \tag{9}$$

4 Results and Conclusion

Model	Data	Training		Validation
		Time(s)	Acc(%)	Acc(%)
1D-ConvNet	D1	1407.12	99.91	99.57
	D2	584.06	99.82	98.87
	D3	321.49	99.75	94.76

Table 1: Average performance of the proposed system

Training of the 1D-ConvNet model follows a supervised learning function, $f : X \to Y$, where Y denotes the set of event labels. Vector embeddings generated by the Skip-gram layer are passed to the downstream ConvNet layer for event prediction via classification based on corresponding event labels. Currently, I am validating my propositions with regard to expanding the experimentation scope to include more real world social network datasets as well as benchmark methods.

Acknowledgements

This research is supported by International Business Machines (IBM) via the provision of a high performance IBM Power System S822LC Linux Server for our laboratory work. Also, I want to acknowledge my doctoral advisor, Professor Ziad Kobti, for his guidance thus far. Consecutively, credit goes to Stanford University [Leskovec and Krevl, 2014] for making their compiled datasets readily available to us.

References

[Cook, 1971] Stephen Arthur Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing, Vol. 1*, pages 151–158, 1971.

[Hans van Maaren and Walsh, 2009] Armin Biere Hans van Maaren and Toby Walsh, editors. *Handbook of Satisfiability*. IOS Press, Amsterdam, The Netherlands, 2009.

[Ian Goodfellow and Courville, 2017] Yoshua Bengio Ian Goodfellow and Aaron Courville, editors. *Deep Learning*. MIT Press, Cambridge, MA, 2017.

[Jason Eisner and Poliak, 2017] Ryan Cotterell Jason Eisner, Benjamin Van Durme and Adam Poliak. Explaining and generalizing skip-gram through exponential family principal component analysis. In *EACL*, 2017.

[Leskovec and Krevl, 2014] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[Scott, 2017] John Scott, editor. *Social Network Analysis*. SAGE Publications Ltd, Newbury Park, CA, 2017.

[Tomas Mikolov and Dean, 2013] Gregory Corrado Tomas Mikolov, Kai Chen and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.