A Trip to Sesame Street: Evaluation of BERT and Other Recent Embedding Techniques Within RDF2Vec

Machine Learning Master's thesis at IDLab, Ghent University – imec

Terencio Agozzino[1]

Advisor:
Prof. Dr. Femke Ongenae[2]

Supervisors:
Ir. Gilles Vandewiele[2] and Ir. Bram Steenwinckel[2]

[1] Haute École en Hainaut, Belgium
terencio.agozzino@std.heh.be
[2] IDLab, Ghent University – imec, Belgium
{firstname.lastname}@ugent.be

September 30, 2020

1 RDF2Vec

1.1 Introduction

Resource Description Framework To Vector (RDF2Vec[3]) [6] is an unsupervised, task-agnostic *algorithm* to numerically represent *nodes* in a Knowledge Graph (KG), allowing them to be used for downstream Machine Learning (ML) tasks. It does this by extracting *walks* in a given KG, which then serve as sentences for existing Natural Language Processing (NLP) techniques, such as Word2Vec[4], to learn *embeddings*[5] from.

1.2 Walks Extraction

As Word2Vec expects a corpus of sentences to train on, RDF2Vec creates these "*sentences*" by extracting different walks from a given *oriented graph.* To extract these walks, a *walking strategy* is needed by starting at a certain *root* node and iteratively *sampling* from the *neighborhood* until a certain *depth* is reached. Currently, the *random walking strategy* is the one used in the RDF2Vec implementation.

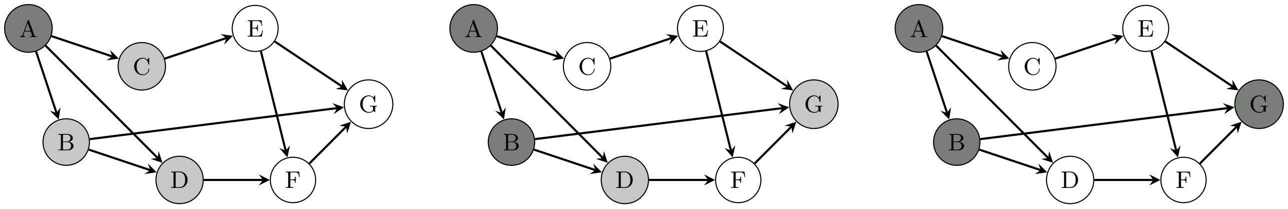

Figure 1. Example of Walks Extraction for an Oriented Graph.

In the graph of Figure 1, the walks extraction starts with A as the root node and B, C, D as possible candidates for the next *hop* in the walk because these are the neighbors of A. Once the hop is made, the list of candidates is updated with the neighbors of the last hop in the walk and the process iterates until it returns an exhaustive list of walks. In the illustrated example, the $A \rightarrow B \rightarrow G$ walk is extracted, among others (e.g., A, $A \rightarrow B$, $A \rightarrow C$). Word2Vec will then learn a representation for each "*word*" (which corresponds to a node in our KG) of the sentences.

[3] http://rdf2vec.org/

[4] NLP technique which takes sequences of words to embed words into vector spaces.

[5] Numerical representation of a node in a given KG.

1.3 Shortcomings

While the representations learned by RDF2Vec have already achieved great predictive performances on a large number of *datasets* in various domains, some *shortcomings* can still be identified:

- **The original RDF2Vec implementation does not incorporate the most recent insights from the NLP domain**: since the inception of RDF2Vec, on 2017, many advancements in the domain of NLP have been made. Most notably, Bidirectional Encoder Representations from Transformers (BERT) [2] is a recent word *embedding technique* that produces more expressive embeddings that result in better predictive performances for downstream ML tasks and can be seen as a more optimal alternative to Word2Vec, which is currently used.
- **The expressiveness of embeddings produced by random walks can be limited**: walks are only single chains, the information they capture is somewhat limited. Extracting more complex data-structures, such as *trees*, or modifying the walking algorithm to introduce extra inductive *biases* could result in more expressive embeddings, which in turn increase predictive performances.
- **RDF2Vec does not scale to large KGs**: as the number of possible walks that can be extracted grows exponentially with depth, RDF2Vec does not scale well to KGs with a large number of nodes, especially when it contains many highly-connected nodes.
- **RDF2Vec cannot deal well with numerical values in the KG**: currently, all hops in the walks, which correspond to nodes from the KG, are handled as categorical data. This is sub-optimal for ordinal and digital data.
- **RDF2Vec cannot deal with volatile data**: to create an embedding of a new node in a KG, walks have to be re-extracted and the NLP models have to be re-trained. Moreover, KGs with a temporal aspect can also not be handled by RDF2Vec.

1.4 Existing Extensions

Several extensions that partially solve some of the shortcomings discussed in the previous section have already been released. Nevertheless, improvements and innovations can still be made.

Cochez et al. [1] proposes a *sampling strategy* to better deal with larger KGs. A naive implementation randomly samples a fixed number of walks for each of the *entities* in order to keep the total number of walks limited. Cochez et al. submitted several *metrics* that can be used to calculate sampling *weights* while walking. Other metrics have also been suggested [5,7].

Added to that, Vandewiele et al. [8] provides several adaptations to walking strategies, which can result in enhanced predictive performances when compared to the random walk strategy.

Finally, other embedding techniques, different than Word2Vec, have been tried out as well. For instance, *KGloVe*[6] uses the Global Vectors for Word Representation (GloVe) embedding technique.

1.5 Python Implementation

The Internet Technology and Data Science Lab (IDLab) is maintaining `pyRDF2Vec`[7], a central implementation of the RDF2Vec algorithm. In addition, the extensions are made in Python, the most common language for ML or data science today.

2 Goals

In 2018, BERT was released and *outperformed* all other existing techniques on many different ML tasks related to a variety of fields. This Master's thesis aims to integrate BERT into `pyRDF2Vec`, but more specifically to *evaluate* its impact in terms of runtime, predictive performance as well as memory usage using different combinations of walking and sampling strategies. Moreover, By reading literature from general graph-based ML and NLP research, inspiration can be found to incept new sampling and walking strategies that result in improved predictive performances by exploiting the properties of BERT.

The goals are *ordered according to importance.* By tackling the tasks in this order, we ensure that the most important tasks are achieved when problems should occur (e.g., a task that takes more time than originally planned).

[6] https://datalab.rwth-aachen.de/embedding/KGloVe/

[7] https://github.com/IBCNServices/pyRDF2Vec/

2.1 Support BERT and Other Embedding Techniques for Comparison Purposes

At least the BERT embedding technique should be integrated into `pyRDF2Vec` and compared to the Word2Vec technique which is currently used. Additionally, other embedding techniques, including but not limited to GloVe and Embedding for Language Models (ELMo), can also be integrated to allow a complete comparison.

2.2 Evaluate the Impact of BERT

A thorough *benchmarking* study to evaluate the impact of the BERT embedding technique on different dimensions will have to be conducted. For this, datasets having different properties and stemming from different domains will have to be selected. Moreover, a rigorous *framework* that performs the required experiments and *logs* all of these results will have to be developed.

2.3 Support of New Walking Strategies for RDF2Vec

While an initial set of five simple walking strategies are already implemented in `pyRDF2Vec`, many other alternatives exist. Implementing more walking strategies would allow larger and detailed comparisons for embedding techniques.

2.4 Support of New Sampling Strategies for RDF2Vec

Similarly, other sampling strategies can be created and implemented in `pyRDF2Vec`. It will thus be possible to see the impact of the choice of a walking strategy and a sampling strategy with the performances related to BERT and other embeddings techniques.

References

1. Cochez, M., Ristoski, P., Ponzetto, S. P., and Paulheim, H. Biased graph walks for RDF graph embeddings. In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics, WIMS 2017, Amantea, Italy, June 19-22, 2017* (2017), R. Akerkar, A. Cuzzocrea, J. Cao, and M. Hacid, Eds., ACM, pp. 21:1–21:12.
2. Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (2019), J. Burstein, C. Doran, and T. Solorio, Eds., Association for Computational Linguistics, pp. 4171–4186.
3. Gesese, G. A., Biswas, R., Alam, M., and Sack, H. A survey on knowledge graph embeddings with literals: Which model links better literal-ly? *CoRR abs/1910.12507* (2019).
4. Kristiadi, A., Khan, M. A., Lukovnikov, D., Lehmann, J., and Fischer, A. Incorporating literals into knowledge graph embeddings. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part I* (2019), C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I. F. Cruz, A. Hogan, J. Song, M. Lefrançois, and F. Gandon, Eds., vol. 11778 of *Lecture Notes in Computer Science*, Springer, pp. 347–363.
5. Mukherjee, S., Oates, T., and Wright, R. Graph Node Embeddings using Domain-Aware Biased Random Walks. *CoRR abs/1908.02947* (2019).
6. Ristoski, P., Rosati, J., Noia, T. D., Leone, R. D., and Paulheim, H. Rdf2vec: RDF graph embeddings and their applications. *Semantic Web 10*, 4 (2019), 721–752.
7. Taweel, A. A., and Paulheim, H. Towards Exploiting Implicit Human Feedback for Improving RDF2vec Embeddings. *CoRR abs/2004.04423* (2020).
8. Vandewiele, G., Steenwinckel, B., Bonte, P., Weyns, M., Paulheim, H., Ristoski, P., Turck, F. D., and Ongenae, F. Walk Extraction Strategies for Node Embeddings with RDF2Vec in Knowledge Graphs. *CoRR abs/2009.04404* (2020).