Third National Conference on Computational Intelligence (NCCI 2019) IOP Publishing
Journal of Physics: Conference Series **1427** (2020) 012001 doi:10.1088/1742-6596/1427/1/012001

Knowledge graph using resource description framework and connectionist theory

Ravi Lourdusamy and Xavierlal J Mattam
Department of Computer Science, Sacred Heart College(Autonomous), Tirupattur, TN 635601, India.

Abstract. Interest in Knowledge Graph has peeked these years. The use of RDF triples for the construction of knowledge graph has been limited by the fact that the vector embedding of RDF to make it machine readable has been constraining factor. This article presents the use of weighted RDF as a vector embedding of RDF that could be used with Bayesian networks in Graph Neural Networks. The vector embedding helps in representing the weights on RDF that could be obtained using Bayes theorem of assessing probability. The resulting weighted RDF could be used in many applications. The use of it in a Clinical Decision Support System is given as a simple illustration together with the calculation of the weights using Bayes theorem. The weighted RDF in Graph Neural Network will represent the knowledge graph using RDF and connectionist theory.

1. Introduction

Cognitive Science has become a very advanced field of research which had it beginning in the mid 1950's. Cognitive science research was accelerated in the mid 1970's with advancements in the field of Artificial Intelligence (AI). The closeness between these two fields, cognitive science and AI, stems from the fact that cognitive science is an attempt to study the human brain using computational modelling and core of AI is to design, build and experiment with computational models. Although they look very similar, the difference is in the fact that cognitive sciences experiments with persons while AI attempts knowledge representation independent of human psychology [1].

In spite of the mutual separateness of the two fields, both AI and Cognitive Sciences are interdependent on each other for their successes and achievements. Cognitive Science Artificial Intelligence could be an interdisciplinary study that combines the two fields and the areas in which the goals of the cognitive science and AI overlaps could be examined. The Soar cognitive architecture, which was originally created by John Laird, Allen Newell, and Paul Rosenbloom and is now maintained at the University of Michigan, is an example [2].

Connectionism or connectionist theory is also another example from an advanced cognitive science research which aims to explain the working of the human intellect with the help of artificial neural networks. In contrast to the earlier classical theories of cognitive sciences which tried to model the intellect using symbolic logic, connectionism describes the human brain as a composition of neural units that have weights to define the strength of the connection between them [3].

The Resource Description Framework (RDF) is a common framework for representing resources using triples and graphs. It is a method of knowledge representation using the graph data model [4]. RDF represents the knowledge using the metadata of the semantic web. The rules of RDF were drafted by a pioneer of AI as an attempt to represent proposition that could be extended using description logic. The data representation using RDF is machine readable and helps achieving the goal of AI [5].

This article describes a method of creating a weighted knowledge graph that could use neural network algorithms to order solve top-k(the first or last k items of a list), spatial and temporal queries. Session 2 of the article is a brief discussion on the background and preliminaries of the work. The motivation that describes an existing problem of combining the semantics and neural networks is explained ins section 3. The section 4 is a brief explanation to a solution to the existing problem. The solution is then presented through a simple illustration section 5. Before the conclusion in section 7 of the article, a brief discussion about the future possibilities is presented in section 6.

2. Background and Preliminaries

There are a few state-of-the-art technologies that have to be taken to consideration for integrating semantics with connectionist theories. In the present section, a brief introduction to these technologies are made. An in-depth study of these technologies will be useful but is avoided due to space constraint of the article. The reference of these technologies will give a detailed understanding of the same.

2.1. Knowledge Graph

It is almost 40 years since knowledge graph has been first initiated by C. Hoede, and F.N. Stokman, mathematicians from the Netherlands. Lots of research has been done and different applications were produced. Starting with the Knowledge Integration and Structuring System (KISS) to Conceptual graphs and Mind graphs, the idea of the knowledge graph was tinkered with in many ways. The Word graph used in linguistics is another research contribution to the making of an automated knowledge graph formation [6].

Knowledge graphs, like any other graphs, has vertices (or nodes) and edges (or relationships). The vertices contain concepts or entities that refer to general categories of physical objects in the real world. The edges establish the relationship between these entities or concepts. The vertices together with the connecting edges together form a knowledge graph which acts as a knowledge repository that converts information to knowledge. The knowledge graph has two main advantages. It helps machine level understanding of natural language text and semi-structured Web tables and it is also a structured repository of knowledge that could be used by many applications [7].

Knowledge graphs became popular in commercial and research activities with the introduction of Google's Knowledge Graph in 2012. In recent years the application of knowledge graphs varies from semantic search assistant tool to acquiring new knowledge using machine learning techniques in order to reuse it in other applications. The Knowledge Graph has to acquire and integrate information on an ontology and form new knowledge using reasoners. The repository that is formed can be represented using Semantic Web standards such as RDF [8].

2.2. Graph Neural Networks

Graph Neural Networks (GNN) use connectionist models to represent nodes and edges of a graph and use machine learning for node classification, link prediction, and clustering. GNN is preferred over the standard neural networks like the convolutional neural networks (CNNs) and recurrent neural network (RNN) because in the standard neural networks input order of nodes are important unlike the GNN which can accept the nodes in any order and the edge(or relationship) between two nodes are considered as a feature of the nodes in standard neural networks while in GNN the edge is necessary to represent the nodes. Most importantly, the standard neural networks have only the ability to generate new knowledge by reordering data already present but GNN has the ability to generate knowledge even from unstructured data like scene pictures and story documents [9].

While the general usefulness of GNNs include node classification, graph classification, network embedding, graph generation, and spatial-temporal graph forecasting, practical applications using GNN are Computer Vision, Natural Language Processing, forecasting traffic speed, volume or the density of roads in traffic networks, Graph-based Recommender systems and many other reasoning and prediction systems. GNNs are powerful in learning graph data and therefore the applications are efficient when compared with applications using standard neural networks [10].

The challenge of using GNN by embedding graphs in vector space to run machine learning algorithm arises due to the limitations in the representational capacity. For example, it is difficult to distinguish two isomorphic graphs since their vector quantities will be the same although the objects of the nodes and edges might be different. One method of representing the isomorphic graph is to take the aggregate of different representations of the same graph. Computational cost of such algorithms will increase with the increase in the exactness of the representation [11].

Another challenge in the use of GNN is the constant updating or incrementing of graphs. Since most real-world data is dynamic, the graphs represented in GNN should also be dynamic. The constant evolution of graphs is both at specific roots and in the whole graph itself. In other words, it is both a global and a local phenomenon and therefore, the challenge to capture both dynamism for the GNN is a great challenge. The challenge in updating graphs is also because the time component should also be taken in consideration [12].

GNN has a pre-processing stage that converts the structured graph into vectors. During this process, the topological relationships among nodes could be lost. One way to avoid the loss is to encode the graph data with the topological relationships between the nodes. There are many ways to do the encoding but the correctness of the output of the GNN has to be ascertained [13].

2.3. Weighted RDF

RDF aims to create machine readable graph data using conceptual modelling of metadata or information about resources. Such modelling of metadata will allow information sharing among heterogeneous applications. To make the RDF graph machine readable, the encoding is done through a process known as reification. Traditional RDF knowledge representation using triples and query languages do not take into consideration ranked query processing and shortest path algorithms, both of which are important in knowledge applications. Therefore, a weighted RDF graph model which consists of a weighted extension to the traditional RDF representation, is used. The weights are used to index the knowledge statements that can also reduce the time complexity of search algorithms [14]. R2DB is one model to extend RDF graphs to include weights specific to an application together with the triples that are part of the adjacent nodes and connecting edge. The top-k queries can be made on the RDF using the R2DB by the application. R2DB also contains specific features that can be used to visualize large sets of query results [15]. R2DB is modelled on R2DF framework that produces utility ranked resource descriptions. The framework also allows the use of weight aggregation of sub-graph to acquire sum, min, max, average, or product. Another important feature of the framework is that the merging operations like join, union, project, filter, order, and distinct can be performed [16].

3. Motivation

The connectionist or sub-symbolic approaches employing artificial neural network fundamentally differs from the symbolic approaches that use logic and reasoning in a knowledge graph. The integration of both the approaches has many benefits. The primary benefit of such an integration is in the removal of the problem 'knowledge acquisition bottleneck'. The problem is caused because of the inability of the system to acquire knowledge automatically and so the knowledge base is limited. By using neural network algorithms with the semantic data, a solution is achieved. Moreover, the integration will lead to deductive reasoning that will help the reasoning process to be extremely quick and accurate with large and complex knowledge graphs [17].

The challenge in such an integration lies in conciliating the methodologies of statistics and logic which are two distinct areas. In achieving the integration, the complexity and expressiveness has to be taken care of. There are three theoretical methods to achieve the integration. The first method is to have a logical characterization of neural systems. The second method is to convert the RDF graphs into vectors. Finally, there is a hybrid method of embedding vectors in RDF graphs. Although these three methods are theoretically possible, the implementation of its practical application is still a challenge [18].

There have been significant contributions in obtaining logical expressions from trained neural networks resulting in the stimulation of the knowledge representation of the brain. But the research and implementation of it is far from the complete. There are numerous challenges that are still posed [19].

4. Using GNN with RDF graph for knowledge graph

Google began the use of the term Knowledge Graph to represent the semantic web searches using RDF datasets. The term and the structure were later used to refer to semantic web knowledge bases [20]. Estimating the importance of a node enhances the utility of the Knowledge Graphs and makes it useful in many applications. The GNN is used to estimate the node importance provided that scores could be embedded in the graph [21].

Knowledge Graph Embedding allows the various entities of a knowledge graph to be embedded in vector spaces without disturbing the inherent structure of the knowledge graph. There are various techniques for knowledge graph embedding that are currently in use for the different knowledge representation methods. Embedding of additional information like entity types, relational paths, textual descriptions and logic rules are also possible [22].

The embedding using RDF2Vec is done by generating the graph walks after knowing the depth of each subtree of a vertex. The depth from the root will give the number of iterations to be done and the number of paths that will be generated. The union of all the paths obtained will give the final sequence set of the graph. The Weisfeiler-Lehman Subtree graph kernel algorithm is then used to make comparison between the two paths. The final sequence set will be the tokens that will be embedded in the RDF. RDF2Vec can also be done using Word2Vec where each unique entity is assigned a weight [23].

5. A simple illustration using Clinical Decision Support System (CDSS)

A knowledge graph for a sample case for CDSS could be as follows: A patient P1 has a visible set of symptoms S (S ϵ {s_1, s_2, …, s_n}), that could be a caused by any one of the known set of diseases D (D ϵ {d_1, d_2, ..., d_n}). The medical practitioner M1 has a prior knowledge that combination of symptoms from S could probably cause one or more diseases from the set D depending on the proportion of the combination of the symptoms. RDF Graph for the sample case is as shown in figure 1.

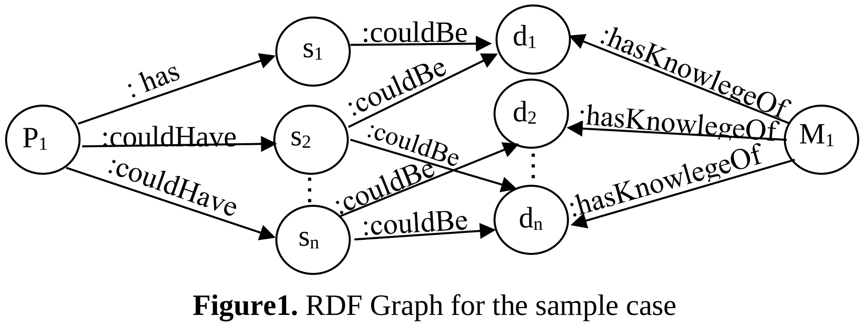

Figure1. RDF Graph for the sample case

The medical diagnosis D is based on P(D|S). Since M1 will not be able to make D without the knowledge of the combinations of S without further tests M1 will have to make suggestions on the type of tests to be carried out based on weights w on possible other related symptoms. So, the D will be then be P(D|S, w) which means the RDF graph will have to include w in its entities that will make it a weighted RDF graph. Using w embedded in the RDF graph could make the GNN based CDSS suggest the possible tests that could be carried to find the right proportion of the combination of the symptoms. This is a very small sample case for illustration. The actual CDSS will have more entities in its Knowledge Graph.

5.1. Note on the calculation of w [24]

Logic of the baysian network is used to calculate the w in P(D|S, w). A rough illustration of the bayes theorem is as follows:

Suppose 90% of patients with symptom s_1 has disease d_1. The conditional probability table will be as is table 1.

Table 1. Conditional probability $P(s_1|d_1)$

d_1 (With d_1)	0.90
$\overline{d_1}$ (Without d_1)	0.10

Now suppose the test for s_2 is conducted to ensure that d_1 is present in the patient P_1 with s_1. Suppose 99% of patients with $s_1 + s_2$ has d_1. The conditional probability will be

By byes theorem $P(d_1|(s_1 + s_2)) = \frac{P(d_1)\times P(s_2|d_1)}{P(s_2)}$ where $P(d_1)$ is the prior probability before s_2 is confirmed, $P(s_2|d_1)$ is the probability of d_1 if s_2 is confirmed and $P(s_2)$ is the probability of s_2 being confirmed.

$P(d_1|(s_1 + s_2)) = P(d_1)\times P(s_2|d_1)/P(s_2)$

$= P(d1)\times P(s_2|d_1)/(P(d_1)\times P(s_2|d_1)+P((\overline{d_1}))\times P(s_2|(\overline{d_1})))$

$= 0.90\text{x}0.99 / (0.90\text{x}0.99 + 0.10\text{x}0.99) = 0.9 = 90\%$

If a further test is carried out to find s_3 with 99% chance of confirmation, then the conditional probability table will be the same as table 2. Then $P(d_1|(s_1+s_2+s_3)) = 0.99*0.99 / (0.99*0.99 + 0.01*0.99) = 0.99 = 99\%$. So, w of P(D|S, w) where D= d_1 and S =$s_1+ s_2+s_3$ is 0.99. A generalised form to calculate w would be $P(d_n|s_n)=P(s_n|d_n)P(d_n)/\Sigma[P(s_n/d_{n-1})P(d_{n-1})]$ where d_n is the current disease diagnosis and s_n is the currently confirmed symptom together with all other confirmed symptoms. The denominator is the sum of all possible diagnosis of the disease with the confirmed symptoms.

Table 2. Conditional probability P(s2|d1)

Symptom	d_1	$\boldsymbol{\overline{d_1}}$
s_2 (s_2 confirmed)	0.99	0.01
$\overline{s_2}$ (s_2 not confirmed)	0.01	0.99

6. Discussion and further possibilities

A Knowledge Graph has to be machine readable. One way to make that possible is to use the binary of the entities in the Knowledge Graph. But that will increase the complexity of the system. The use of the semantics of the Knowledge Graph embedded with weights will can reduce the complexity of the system. The weighted RDF graph is can also help the system predict the connected nodes and its relations with the help of GNN. Such a system can complete Knowledge Graphs on its own provided there is the feasible amount of training sets.

As in case of neural networks, the higher the number the training sets, the greater will be the accuracy of prediction. As in the case of CDSS, applications will not have the possibility of having huge training sets to run in their systems. Therefore, the efficiency of the connectionist algorithm working with the RDF Knowledge Graph will brought down.

Another drawback is the scale of the weights and its updates. Since they are very important for the performance of the neural network, there should be some method to standardise the weights.

Moreover, the hidden layers in any neural network is hard to be explained. Therefore, users of applications with neural networks have very little confidence in the efficiency of the system and require reconfirmation at every stage leading to excessive loss.

Since the work on combining symbolic and connectionist theories are still at the primary research stages, there is very little to evaluate it with. A lot is left for future development. But the possibility and efficacy of using Knowledge Graph using RDF and GNN is unmistakably established.

7. Conclusion

There has been a clear demarcation between the semantic and connectionist theories in the past. The compartmentalisation is slowly disappearing since both Cognitive Science and Artificial Intelligence have developed to a stage where the boundary lines between the symbolic logic and neural networks have disappeared leading to hybridisation of both techniques. The Knowledge Graph using RDF and connectionist theories are a proof of the possibility of using semantics in neural networks. Using RDF and GNN is one possibility that can be implemented in applications like in CDSS. Although there is a lot of research and development to be done in the field, the possibility and its advantages cannot be denied. A clear way forward is shown but achieving the goal requires a lot of refinement.

Reference

[1] Thagard P 2018 Cognitive Science *Stanford Encyclopedia of Philosophy*

[2] Luber S 2011 Cognitive science artificial intelligence: Simulating the human mind to achieve goals *2011 3rd International Conference on Computer Research and Development* 207–10

[3] Buckner C and Garson J 2019 Connectionism *Stanford Encyclopedia of Philosophy*

[4] Schreiber A T and Raimond Y 2013 RDF 1.1 Primer *W3C*

[5] Halpin H 2004 The semantic web: The origins of artificial intelligence redux *Third international workshop on the history and philosophy of logic, mathematics, and computation*

[6] Sri Nurdiati S N, Hoede C. 25 years development of knowledge graph theory: the results and the challenge. Enschede: University of Twente, 2008 (Memorandum; 2/1876)

[7] Yan J, Wang C, Cheng W, Gao M and Zhou A 2016 A retrospective of knowledge graphs *Frontiers of Computer Science* **12** 55–74

[8] Ehrlinger L, and Wöß W 2016 Towards a Definition of Knowledge Graphs *1st International Workshop on Semantic Change & Evolving Semantics* **1695**

9] Zhou J, Cui G, Zhang Z., Yang C, Liu Z, and Sun M 2018 Graph neural networks: A review of methods and applications *Preprint* arXiv/1812.08434.

[10] Wu Z, Pan S, Chen F, Long G, Zhang C, and Yu P S 2019 A comprehensive survey on graph neural networks *Preprint* arXiv/1901.00596

[11] Xu K, Hu W, Leskovec J, and Jegelka S 2018 How powerful are graph neural networks? *Preprint* arXiv/1810.00826

[12] Ma Y, Guo Z, Ren Z, Zhao E, Tang J, and Yin D 2018 Streaming Graph Neural Networks

[13] Scarselli F, Gori M, Tsoi A C, Hagenbuchner M and Monfardini G 2009 The Graph Neural Network Model *IEEE Transactions on Neural Networks* **20** 61–80

[14] Cedeno J P 2010 A framework for top-K queries over weighted RDF graphs (Doctoral dissertation, Arizona State University).

[15] Liu S, Cedeno J P, Candan K S, Sapino M L, Huang S and Li X 2012 R2DB: A System for Querying and Visualizing Weighted RDF Graphs *2012 IEEE 28th International Conference on Data Engineering* pp 1313-16

[16] Cedeno J P, and Candan K S 2011 R2DF framework for ranked path queries over weighted RDF graphs *Proceedings of the international conference on web intelligence, mining and semantics* p. 40

[17] Hitzler P, Bianchi F, Ebrahimi M and Sarker M K 2019 Neural-symbolic integration and the Semantic Web *Semantic Web* pp 1–9

[18] Besold T R et al. 2017 Neural-symbolic learning and reasoning: A survey and interpretation *Preprint* arXiv/*1711.03902*

[19] Garcez A D A, Besold T R, De Raedt L, Földiak P, Hitzler P, Icard T, Kühnberge K, Lamb L C, Miikkulainen R and Silver D L 2015 Neural-symbolic learning and reasoning: contributions and challenges *AAAI Spring Symposium Series*

[20] Paulheim H 2016 Knowledge graph refinement: A survey of approaches and evaluation methods *Semantic Web* **8** 489–508

[21] Park N, Kan A, Dong X L, Zhao T, and Faloutsos C 2019 Estimating Node Importance in Knowledge Graphs Using Graph Neural Networks *Preprint* arXiv/*1905.08865*.

[22] Wang Q, Mao Z, Wang B and Guo L 2017 Knowledge Graph Embedding: A Survey of Approaches and Applications *IEEE Transactions on Knowledge and Data Engineering* **29** 2724–43

[23] Ristoski P and Paulheim H 2016 RDF2Vec: RDF Graph Embeddings for Data Mining *Lecture Notes in Computer Science The Semantic Web – ISWC 2016* 498–514

[24] Westbury C F 2010 Bayes' Rule for Clinicians: An Introduction *Frontiers in Psychology* **1** 192.