IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-3, Issue-5 E-ISSN: 2347-2693

Obtaining New Facts from Knowledge Bases using Neural Tensor Networks

Sagarika Sahoo[1*] and Avani Jadeja[2]

[1*,2] *Hasmukh Goswami College of Engineering, GTU, India*

www.ijcseonline.org

Received: Apr/26/2015 Revised: May/06//2015 Accepted: May/22/2015 Published: May/30/ 2015

***Abstract*—** Knowledge bases are an important resource for easily accessible, systematic relational knowledge. They provide applications with the benefit of question answering and other tasks but often suffer from their incompleteness, lack of knowledge and ability to purpose over new entities and relations. Much work has done to build and extend the relationship in knowledge base. This paper mainly focuses on completing a knowledge base by reasoning over entities relationship with Neural Tensor Network (NTN). New relationships can be predicted with Neural Tensor Network that can be added to the database. We make evident that the model is improved by making entities represented with vectors learned from unsupervised large corpora.

***Keywords*—** Neural tensor Network, knowledge base, entity vector, bilinear tensors, unsupervised text.

I. INTRODUCTION

Ontologies and knowledge bases such as WordNet [1], Yago [2] or the Google Knowledge Graph are extremely useful resources for query expansion [3], coreference resolution [4], question answering (Siri), information retrieval or providing structured knowledge to users. Much work focused on extending knowledge bases using pattern or classifiers on large text bodies. However, the knowledge that is recognizable is not expressed in the large text corpora. Due to this, they suffer from incompleteness and a lack of reasoning capability.

We introduce a model with the complementary goal of learning new facts from the existing facts in the knowledge bases. For instance, when told that Pablo Picaso is from Florence, then a person does not need to find textual evidence to know that Pablo Picaso belongs to the country Italy. Here we demonstrate a model that can accurately predict the likely truth of additional facts using an existing database. The goal is achieved by representing each entity (either individual or an object) in the database as a vector that can obtain facts about that entity and their feasibility of a certain relation.

The relationship between two entity vectors can explicitly be related using Neural Tensor Network and the relationships are defined by the parameters of Neural Tensor Network which is more powerful than the standard neural network. Furthermore, our model allows us to ask whether even entities that were not in the database are in certain relationships by simply using distributional word vectors. These vectors are learned by neural network model [7] using unsupervised text corpora. The model allow us to extend the database without any additional parsing of other textual resources or any manually designed rules and still captures the semantic and syntactic information from the same database.

Our model illustrated in Figure. 1 uses a different methodology from the previous knowledge base models [8, 9, 10, 16] by a large margin. Previous work [8, 9, 10, 16] represents each entity with one vector. However, does not allow the sharing of statistical strength if entity names share similar substrings. Instead, we represent each entity as the average of its word vectors, allowing the sharing of statistical strength between the words describing each entity as in Socher et. al. [16] e.g., Bank of India and India.

II. RELATED WORK

Vast amount of work has been done on extending the large text corpora but little work has been done on extending the knowledge bases purely based on the existing database. A related approach is by Sutskever et al. [10] who use tensor factorization and Bayesian clustering for learning relational structures. Our model purely relies on learned entity vectors instead of clustering the entities in a nonparametric Bayesian framework. Another related work is that by Bordes et al. [8] and Jenatton et al. [9] who also learn vector representations for entries in a knowledge base. Ranzato et al. [12] introduced a factored 3-way Restricted Boltzmann Machine which is also parameterized by a tensor. Our model implements their approach but outperform their model by initializing with unsupervised word vectors.

Neural tensor Network uses standard backpropagation with some modifications. Lastly, the model does not require multiple embedding for each entity. Instead, we consider the subunits (space separated words) of entity names. This allows more statistical strength to be shared among entities. Many methods that use knowledge bases as features such as

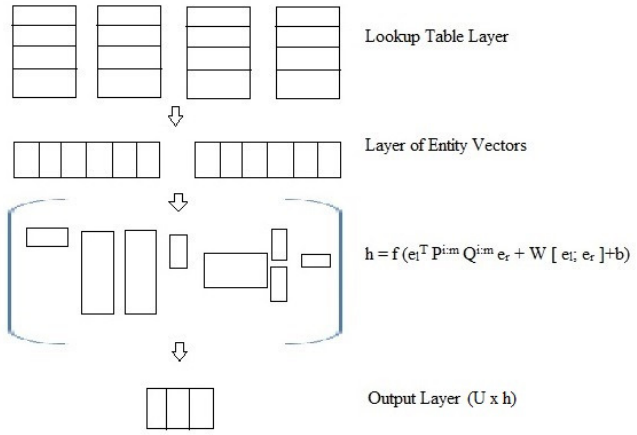

Figure 1: Visualization of the Neural Tensor Network layers.

[3, 4] could benefit from a method that maps the provided information into vector representations. We learn to modify unsupervised word representations via grounding in world knowledge.

III. NEURAL TENSOR NETWORKS

This section presents the Neural Tensor Networks model that reasons over databases by learning vector representations over them. As shown in Figure 1, each relation triple is described by a neural network and pairs of database entities which are given as input to that relation's model. If the entities are in that relationship then the model returns a high score otherwise a low one. Due to this, any fact can be scored with some certainty mentioned implicitly or explicitly in the database. Below we describe the Neural Tensor Networks model.

A. Neural Tensor Network for reasoning

The aim is to learn model which have the ability to realize the additional facts that hold purely due to the existing relations in the same database. The model is actually for common sense reasoning. The goal of the model will be to state whether two entities (e_l; e_r) are in certain relationship R where el represents left entity and er represents right entity. In another words, the model aims for link prediction in an existing network of relationships between entity nodes. For instance, whether the relationship (e_l; R; e_r) = (cat; has part; tail) is true and with what score of certainty. Suppose e_l, $e_r \in R^d$ be the vector representations (or features) of the two entities from the lookup table L.

The Neural Tensor Network (NTN) replaces a standard linear neural network layer with a bilinear tensor layer that directly relates the two entity vectors across multiple dimensions. A tensor is a geometric object that describes relations between vectors, scalars, and other tensors. It can be represented as a multi-dimensional array of numerical values. An advantage of the tensor is that it can explicitly model multiple interactions in data.

By this model, we can compute a score of how probable it is that two entities are in certain relationship. Let e_l, $e_r \in R^d$ be the vector representation of two entities where, d is the size of the entity vector. We apply a tensor-based transformation to the input vector. Formally, we use a 3-way tensor $V^{[1:m]} \in R^{dxdxk}$ to directly model the interaction. The output of each tensor product $z \in R^k$, where each dimension z_i is the result of the bilinear form defined by each tensor slice $V_i \in R^{dxd}$

$$z = e_l^T V^{[1:m]} e_r \quad (1)$$

$$z_i = e_l^T V_i e_r$$

$$= \sum e_l^T V_i e_r \quad \text{(summation from } i = 1 \text{ to m)}$$

The NTN based function that predicts the relationship of two entities can be described as follows:

$$h(e_l, r, e_r) = f (z + W [e_l; e_r] + b) \quad (2)$$

where f = tanh is a standard nonlinearity applied element-wise and z is same as in equation 1. The other parameters for relation R are the standard form of a neural network: $W \in R^k$ x 2d and $b \in R^k$.

Moreover, the additional tensor could bring millions of parameters to the model which makes the model suffer from the risk of overfitting. To remedy this, we propose a tensor factorization approach that factorizes each tensor slice as the product of two low-rank matrices. The tensor $V_i \in R^{dxd}$ is factorized into two low rank matrix $P \in R^{dxr}$ and $Q \in R^{rxd}$ where r << d as shown in figure 2.

$$V_i = P_i \text{ x } Q_i \quad \text{where } 1 < i < m \quad (3)$$

Now substituting equation (1) and (3) in equation (2) we get

$$h(e_l, r, e_r) = f (e_l^T P^{i:m} Q^{i:m} e_r + W [e_l; e_r] + b)$$

Instead of relating inputs implicitly through the nonlinearity where entity vectors are concatenated this model directly relates the two entity vectors. Figure 1 shows the conception of the NTN model. Intuitively, we can see each slice of the tensor as being responsible for one type of entity pair or instantiation of a relation, i.e. each tensor slice mediates the relationship between two entity vectors differently. The main advantage of this model is that it can directly relate the two inputs instead of only implicitly through the nonlinearity.

.

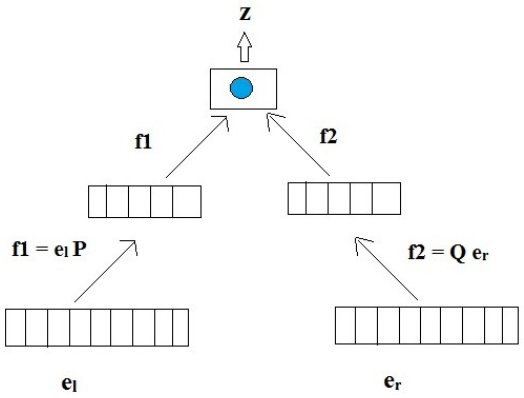

Figure 2: *Visualization of factorization of the vector V into two vectors P and Q.*

B. Representation of Entities

Previous work [8, 9, 10] assigned a single vector representation to each entity of the knowledge base, which does not allow the sharing of statistical strength between the words describing each entity. In contrast to previous models, our model has two important features: representing entities by their word vectors and initializing word vectors with pre-trained vectors. We model each vector as d-dimensional vector $\in R^d$ and compute each entity vector as configuration of its word vectors [16]. For example, if the training data includes a fact that homo sapiens is a type of hominid and this entity is represented by two vectors v_{homo} and $v_{sapiens}$, we may extend the fact to the previously unseen homo erectus, even though its second word vector for erectus might still be close to its random initialization. We represent the entity vector by averaging its word vectors as in Richard et. al. [16]. For example, $v_{homo\ sapiens} = 0.5(v_{homo} + v_{sapiens})$.

An additional advantage can be incurred by training word vectors that can benefit from pre-trained unsupervised word vectors, which in general capture some distributional syntactic and semantic information.

C. Classification

In this experiment, we ask the model to predict correct facts in the testing data in the form of (e_l, R, e_r). This could be seen as answering questions such as does a dog have a tail? using the scores g(dog, has part, tail) computed by the various models.

The model uses a development set fold to find a threshold T^R for each relation such that if $f(e_l, R, e_r) \geq T^R$, the relation (e_l, R, e_r) holds, otherwise it is considered false. The final accuracy is based on how many of triplets are classified correctly. In particular, we constrain the entities from the possible answer set for database by only allowing entities in a position if they appeared in that position in the dataset. For example, given a correct triplet (Ganesha, nationality, India), a potential negative example is (Ganesha, nationality, United States). We use the same way to generate the development set. This forces the model to focus on harder cases and makes the evaluation harder since it does not include obvious non-relations such as (Ganesha, nationality, Van Gogh).

IV. EXPERIMENTAL RESULTS

We will compare the proposed model with the MFS (Most Frequent Sense) model in terms of accuracy obtained in finding the correct triplet from the relations in the test data. In order to get highest accuracy we cross validate the data with the development set. The hyper-parameters used in the proposed model are:

(i) Vector initialization
(ii) Number of iterations are 500
(iii) The regularization function used $\lambda = 0.0001$
(iv) The dimensionality of hidden vectors d = 100
(v) The activation function used is tanh

The comparison between NTN (Neural Tensor Network) and MFS is done based on the fact that how many correct triplets are identified by both the models. The table below represents the percentage of accuracy in ascending order obtained from MFS model, MFS plus other models and the NTN model.

S. No #	Neural Network Model Names	Accuracy (in percentage)
1	MFS	67.1 %
2	MFS + All	72.3 %
3	NTN	81.3 %

Table 1: Comparison between MFS model and NTN model

As the comparison table above shows that the NTN model improves the accuracy by almost nine percentages. The proposed model is able to improve the accuracy on using the vector representation of words in the dataset and using tensors along with the improved parameters discussed above.

V. CONCLUSIONS AND FUTURE WORK

Our model Neural Tensor Networks is capable of predicting unseen facts from the knowledge bases and able to learn new relationship among entities. Unlike previous models for predicting relationships purely using entity representations in knowledge bases, our model allows direct interaction of entity vectors via a tensor. This architecture enables the extension of databases without manually designed extraction rules and even without external textual

resources. Our approach does not explicitly deal with polysemous words. One possible future extension is to incorporate the idea of multiple word vectors per word as in Huang et al. [22].

II. REFERENCES

[1] G.A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 1995.

[2] F.M. Suchanek, G. Kasneci, and G.Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, 2007.

[3] J. Graupmann, R. Schenkel, and G. Weikum. The SphereSearch engine for unified ranked retrieval of heterogeneous XML and web documents. In *Proceedings of the 31st international conference on Very large data bases*, VLDB, 2005.

[4] Ng and C. Cardie. Improving machine learning approaches to coreference resolution. In *ACL*, 2002.

[5] R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In *NIPS*, 2005.

[6] Fader, S. Soderland, and O. Etzioni. Identifying relations for open information extraction. In *EMNLP*, 2011.

[7] J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semisupervised learning. In *Proceedings of ACL*, pages 384–394, 2010.

[8] Bordes, X. Glorot, J. Weston, and Y. Bengio. Joint Learning of Words and Meaning Representation for Open-Text Semantic Parsing. *AISTATS*, 2012.

[9] Bordes, J. Weston, R. Collobert, and Y. Bengio. Learning structured embeddings of knowledge bases. In *AAAI*, 2011.

[10] Sutskever, R. Salakhutdinov, and J. B. Tenenbaum. Modelling relational data using Bayesian clustered tensor factorization. In *NIPS*, 2009.

[11] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *EMNLP*, 2012.

[12] M. Ranzato and A. Krizhevsky G. E. Hinton. Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images. *AISTATS*, 2010.

[13] R. Collobert and J. Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *ICML*, 2008.

[14] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3, March 2003.

[15] H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. Improving Word Representations via Global Context and Multiple Word Prototypes. In *ACL*, 2012.

[16] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks For Knowledge Base Completion. In Advances in Neural Information Processing Systems 26.

[17] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In EMNLP, 2012.

[18] Richard Socher, Recursive Deep Learning for Natural Language Processing and Computer Vision, 2014.

[19] R. Collobert and J.Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In ICML, 2008.

[20] R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In NIPS. MIT Press, 2011.

[21] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. Improving Word Representations via Global Context and Multiple Word Prototypes. In ACL, 2012.

[22] Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Chris Manning, Andrew Ng and Chris Potts. Conference on Empirical Methods in Natural Language Processing (EMNLP 2013, Oral).

[23] L. Deng and D. Yu, "Deep Learning: Methods and Applications "http://research.microsoft.com/pubs/209355/DeepLearning-NowPublishing-Vol7-SIG-039.pdf"

[24] Y. Bengio, A. Courville, and P. Vincent., "Representation Learning: A Review and New Perspectives," IEEE Trans. PAMI, special issue Learning Deep Architectures, 2013.

AUTHORS PROFILE

Sagarika Sahoo, received her Bachelor of Engineering in Information Technology degree from Hemachandracharya North Gujarat University. She has teaching experience of 3.5 years in engineering college. Currently she is pursuing Master of Engineering in Computer Engineering from Hasmukh Goswami College of Engineering, Gujarat Technological University, India.