CS 7301: Neurosymbolic Methods
Assignment 1

Siwen Yan and Harsha Kokel

Abstract

This report compares different neurosymbolic approaches on a simple task of link prediction. We aim to empirically evaluate DeepProbLog, PLogicNet, Neural LP, Metapath2Vec, ComplEx, ROCGCN, and BoostSRL on *ICML CoAuthors* dataset, to investigate the advantages of fusion of neural and symbolic approaches.

1 Introduction

The problem of link prediction (also, knowledge base completion) is the task of discovering connections between entities. Since knowledge bases are always incomplete (i.e. there are many missing links among entities) this problem is well studied in literature. Traditionally, two different approaches have been used to tackle the link prediction task.

One is **knowledge graph embedding** approach. Here, embeddings $\overrightarrow{E}, \overrightarrow{R}$ are learnt for all the entities E and the relations R in the domain, then a link between entities A and B is predicted if there exists a relation L such that $\overrightarrow{A} = \overrightarrow{B} + \overrightarrow{L}$. TransE (Bordes et al. 2013) and ComplEx (Trouillon et al. 2016) are two of the popular examples of this approach.

The other approach involves learning **first-order logical rules**. Here, the relationship in the graph are represented as first-order predicates and from training examples generalizable first-order clauses are learnt. A link between entities are predicted when those entities satisfy one or more rule. MLN (Richardson and Domingos 2006) and RDN-Boost (Natarajan et al. 2012) are two popular examples from this approach.

Recently, there is a surge in NeuroSymbolic approaches that leverage the strengths of both embedding and first-order logic. In the next section we briefly introduce these approaches and in section 3 we compare these approaches by empirically evaluating them on *ICML CoAuthors* link prediction task.

2 Background

DeepProbLog (Manhaeve et al. 2018) combines deep learning with probabilistic logic programming. The main idea is to integrate the reasoning capability of ProbLog with the detection capability of neural network. They achieve it by extending the probablistic logic to incorporate *neural predicates*. The output of a neural network which can be interpreted as a probability is encapsulated as a neural predicate. Leveraging the algebraic ProbLog formulation, the DeepProbLog network can be trained using gradients by back propagation. Inference is done similar to the ProbLog using Sentential Decision Diagram with output of neural network as probability for the ground annotated disjunctions.

PLogicNet (Qu and Tang 2019) combines MLNs and Knowlege Graph Embeddings (KGE). MLNs are good at capitalizing on the first order logic rules but struggle to efficiently perform inference when the domain size increases. KGEs on other hand can scale well but has no way to incorporate rich domain knowledge like rules. PLogicNet uses the variational EM to learn logic rules in MLN. In E step, inference is performed where the variational distribution is parameterized as a knowledge graph embedding model and in M step weights are optimized using both the observed facts and the facts inferred by the knowledge graph embedding model.

Neural LP Neural Logic Programming (Yang, Yang, and Cohen 2017) is an end-to-end differentiable model, which can learn probabilistic first-order logical rules. This approach is based on a previous differentiable logic TensorLog (Cohen 2016). It is a neural controller system with an attention mechanism to select a subset of TensorLog's operations and do those operations on contents chosen from the memory. Then after the system is trained, logical rules can be recovered by tracking the attention of the system on examples. It performs well in statistical relation learning, grid path finding, knowledge base completion and question ansewring against knowledge base.

Metapath2Vec (Dong, Chawla, and Swami 2017) constructs heterogeneous neighborhood of each node by doing meta-path-based random walks, and generates node (entity) embeddings by a heterogeneous skip-gram model. It extends word2vec (Mikolov et al. 2013a,b)-based network

Method	Recall	Precision	F1	AUC-PR	AUC-ROC
MLN-Boost	0.651±0.23	0.10±0.07	0.146±0.09	0.23±0.09	0.8±0.05
RDN-Boost	0.638±0.18	0.1±0.06	0.154±0.08	0.235±0.05	0.750±0.07
PLogicNet[1]	0.585±0.05	0.867±0.06	**0.699±0.05**	**0.728±0.08**	0.933±0.05
Neural-LP	**0.903±0.10**	0.095±0.01	0.172±0.02	0.257±0.08	0.912±0.05
Metapath2Vec	0.884±0.05	0.169±0.03	0.283±0.05	0.373±0.09	**0.941±0.01**
ComplEx	0.755±0.14	0.097±0.02	0.171±0.04	0.185±0.10	0.863±0.07
ROCGCN	0.389	**1.0**	0.561	0.556	

Table 1: Results of different methods.

	MLN/RDN Boost	PLogic Net	Neural LP	Metapath -2Vec	ComplEx	ROCGCN
Used Validation Set?	×	✓	✓	✓	×	✓
Used negative examples in training?	✓	×	×	✓	×	✓
Provides Logic Rules?	✓	×	×	×	×	×
Documentation available?	✓	×	✓	✓	✓	×
GPU used?	×	✓	×	×	×	×
No of iterations?	20 Trees	10000	$5-10$	$1+\leq 2000$	4000	200
Embedding Dimension?	NA	100	128	128	50	16

Table 2: Setting used in each method

representation learning from homogeneous networks to heterogeneous networks. The further latent-space representation learning helps model similarities between nodes without connected meta-paths. Hence it performs well in heterogeneous network mining tasks. It can be used for node classification, clustering, similarity search and structural and semantic correlations between diverse network objects.

ComplEx (Trouillon et al. 2016), use complex valued embeddings which works well for binary, symmetric and antisymmetric relations. In knowledge base, each entity can appear as subject or object. So for each entity, it is supposed to have different embeddings as subject and object. The complex conjugate of the object embedding vector of each entity is its subject embedding vector in ComplEx. ComplEx is scalable since it only uses Hermitian dot product and it is linear in both space and time complexity. As evaluated in (Trouillon et al. 2016), ComplEx outperforms DistMult (Yang et al. 2015) which does not model antisymmetry and TransE (Bordes et al. 2013) which though is not a symmetric model.

ROCGCN Relational One-Class GCN (ROCGCN) is a relational extension to Graph Convolutional Network (GCN). It leverages the success of statistical relational learning (SRL) by constructing a secondary graph using relational density estimation. ROCGCN outperforms many recent methods on link prediction and node classification tasks over multiple data sets.

[1]Used TransE variant in PLogicNet

3 Empirical Evaluations

We compare these NeuroSymbolic approaches for link prediction on an *ICML CoAuthors* prediction task. This dataset is extracted from the publication information of ICML 2018 and consists of 556 entities and 5 relations. Entities included are of 5 types: *author*, *institute*, *type-of-institute* (for e.g. company or university), *location*, *topic*. Relationships include *is-type-of* (an institute is a type of university or company), *interested-in-topic* (an author is interested in some research topic), *affiliated-with* (an author is affiliated with some institute), *located-at* (every institute is located at some place), and *coauthor*. The target predicate is *coauthor*, whether two authors have a publication together.

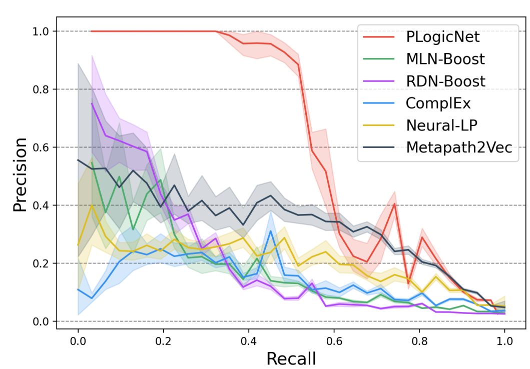

Figure 1: Precision Recall Curve

The positive and negative target predicates are splitted randomly into 5 folds with roughly equal sizes. About 31 positive examples and 1299 negative examples in each fold.

Each fold is considered as test data, and the remaining folds are considered as train data. So, about 124 positive examples and 5196 negative examples for training. For experiments where we needed validation set, we train on 3 folds, validate on 1 and test on 1. Table 2 summarizes our experimental setup for each method.

We note that *we were not able to evaluate the DeepProbLog*, since we could not get the parameter learning in ProbLog to work. Hence, we skip DeepProbLog from our evaluation. In addition to the above mentioned NeuroSymbolic method, we include the MLN-Boost and RDN-Boost in our evaluations.

The average and standard deviation of recall, precision, F1, AUC-PR and AUC-ROC of 5 folds is shown in the Table 1. The results on individual folds can be found in the Table 3 in *Appendix*.

Since the *ICML CoAuthors* domain is highly imbalanced, we present the PR curves in Figure 1. We see that the PLogicNet outperforms all the other approaches. We think this is because PLogicNet iterates between predicting links and learning MLN. *While the other approaches optimize on only the observed data, PLogicNet optimizes even on inferred data from the KGEs*. However, we acknowledge that pLogicNet used a validation set and the improved result could be an artifact of that. To remove the use of validation set and observing the performance without it remains for future analysis.

It seems that a NeuroSymbolic method can perform very well when both the neural and symbolic parts are optimized simultaneously or iteratively, which improves the overall performance.

References

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, 2787–2795.

Cohen, W. W. 2016. Tensorlog: A differentiable deductive database. *arXiv preprint arXiv:1605.06523* .

Dong, Y.; Chawla, N. V.; and Swami, A. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 135–144.

Manhaeve, R.; Dumancic, S.; Kimmig, A.; Demeester, T.; and De Raedt, L. 2018. Deepproblog: Neural probabilistic logic programming. In *Advances in Neural Information Processing Systems*.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Natarajan, S.; Khot, T.; Kersting, K.; Gutmann, B.; and Shavlik, J. 2012. Gradient-based boosting for statistical relational learning: The relational dependency network case. *Machine Learning* 86(1): 25–56.

Qu, M.; and Tang, J. 2019. Probabilistic logic neural networks for reasoning. In *Advances in Neural Information Processing Systems*.

Richardson, M.; and Domingos, P. 2006. Markov logic networks. *Machine learning* .

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex embeddings for simple link prediction. International Conference on Machine Learning (ICML).

Yang, B.; Yih, W.-t.; He, X.; Gao, J.; and Deng, L. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations*.

Yang, F.; Yang, Z.; and Cohen, W. W. 2017. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems*, 2319–2328.

Appendix

Method	Test Fold	Recall	Precision	F1	AUC-PR	AUC-ROC
MLN-Boost	fold 1	0.935	0.034	0.067	0.368	0.816
MLN-Boost	fold 2	0.645	0.054	0.10	0.097	0.746
MLN-Boost	fold 3	0.870	0.039	0.074	0.280	0.808
MLN-Boost	fold 4	0.516	0.20	0.29	0.237	0.888
MLN-Boost	fold 5	0.290	0.16	0.20	0.177	0.780
MLN-Boost	average	0.651±0.23	0.10±0.07	0.146±0.09	0.23±0.09	0.8±0.05
RDN-Boost	fold 1	1.00	0.023	0.046	0.184	0.657
RDN-Boost	fold 2	0.580	0.05	0.09	0.25	0.707
RDN-Boost	fold 3	0.548	0.078	0.137	0.223	0.754
RDN-Boost	fold 4	0.612	0.165	0.260	0.32	0.848
RDN-Boost	fold 5	0.45	0.170	0.24	0.2	0.780
RDN-Boost	average	0.638±0.18	0.1±0.06	0.154±0.08	0.235±0.05	0.750±0.07
PLogicNet	fold 1	0.645	0.952	0.769	0.790	0.972
PLogicNet	fold 2	0.548	0.809	0.653	0.584	0.828
PLogicNet	fold 3	0.548	0.772	0.641	0.702	0.954
PLogicNet	fold 4	0.64	0.909	0.754	0.815	0.965
PLogicNet	fold 5	0.548	0.894	0.68	0.751	0.951
PLogicNet	average	0.585±0.05	0.867±0.06	0.699±0.05	0.728±0.08	0.933±0.05
Neural-LP	fold 1	0.935	0.105	0.190	0.235	0.923
Neural-LP	fold 2	0.742	0.077	0.140	0.163	0.820
Neural-LP	fold 3	0.903	0.086	0.157	0.206	0.931
Neural-LP	fold 4	1.0	0.105	0.190	0.314	0.966
Neural-LP	fold 5	0.935	0.101	0.182	0.367	0.920
Neural-LP	average	0.903±0.10	0.095±0.01	0.172±0.02	0.257±0.08	0.912±0.05
Metapath2Vec	fold 1	0.871	0.214	0.344	0.366	0.942
Metapath2Vec	fold 2	0.935	0.167	0.283	0.286	0.956
Metapath2Vec	fold 3	0.935	0.146	0.252	0.368	0.943
Metapath2Vec	fold 4	0.871	0.131	0.228	0.328	0.938
Metapath2Vec	fold 5	0.806	0.189	0.307	0.517	0.926
Metapath2Vec	average	0.884±0.05	0.169±0.03	0.283±0.05	0.373±0.09	0.941±0.01
ComplEx	fold 1	0.645	0.079	0.141	0.104	0.814
ComplEx	fold 2	0.581	0.075	0.132	0.116	0.772
ComplEx	fold 3	0.774	0.102	0.180	0.197	0.893
ComplEx	fold 4	0.871	0.102	0.183	0.344	0.924
ComplEx	fold 5	0.903	0.126	0.221	0.165	0.912
ComplEx	average	0.755±0.14	0.097±0.02	0.171±0.04	0.185±0.10	0.863±0.07
ROCGCN	average	0.389	1.0	0.561	0.556	

Table 3: Detailed results of 5 folds.