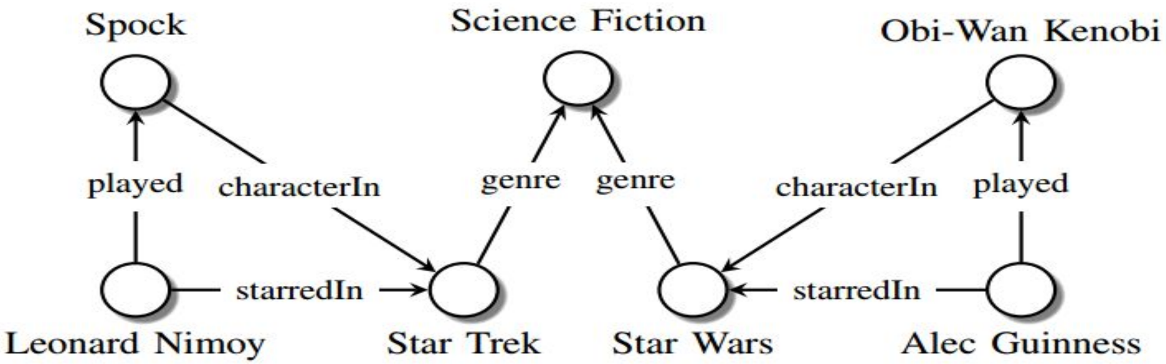
Affinity Dependent Negative Sampling for Knowledge Graph Embeddings

M M Alam, **H Jabeen**, M Ali, K Mohiuddin, and J Lehmann

Smart Data Analytics, University of Bonn
CEPLAS, University of Cologne
hajira.jabeen@uni-koeln.de

Introduction



Knowledge graph

- A special kind of relational data in terms of subject, predicate and object.
- Knowledge graph encodes available information based on entities and their relations.
- Example- DBpedia, Yago, Freebase, WordNet.

Negative sampling

- To contrast with already available data which is considered true.
- Essential step to help vector based embedding models to learn link prediction tasks.

Image Source: Maximilian Nickel et al. A Review of Relational Machine Learning for Knowledge Graphs: From Multi-Relational Link Prediction to Automated Knowledge Graph Construction

Knowledge Graph Embedding Models

- Encode the information contained in Knowledge graphs as
  - Vectors
  - Tensors
- Embeddings
  - Multidimensional vector representations for entities or relations
- Capture
  - Semantic similarity of entities
- Optimize
  - Translational objective for similarity scores (TransE : $h+r \approx t$)
  - Optimize bilinear scores (DistMult : $\langle h,t,r \rangle$)
- Applications
  - KG Completion

Related Work

- Random Negative Sampling [1]
- Corrupting Positive Triple (True Triple) Based on Relations [1]
- Typed Negative Sampling [1]
- Distributional Negative Sampling [2]
- Relational Sampling [1]
- Nearest Neighbor sampling [1]
- Near Miss sampling [1]

Adaptive Distributional Negative Sampling

- Inspired by the Distributional Negative Sampling (DNS)[2]
- Draws out most similar vectors of entities for corruption adaptively
- We select the similar entities for corruption from each batch
- Execution time improvement
- Vector based fitness function that extracts candidate entities

ADNS - Algorithm

INPUT: *Training set* $S_{\{h,r,t\}}$, *Entity Set* ξ, *Relation set* R, *batch Size* β, *Number of Negatives* C
OUTPUT: *For Given Batch* $\beta_{\{\mathrm{h,r,t}\}}$ *return negative triples* $N_{\{\mathrm{h',r,t'}\}}$

Function `Sample Negative`$(S_{\{h,r,t\}}, \xi, R, \beta, C)$:
  for *triple* $t \in \beta_{\{h,r,t\}}$ **do**
    $candidate\_position = Bern(t, R)$
      ▷ bern negative sampling to decide whether head or tail corruption
    $candidate = t_{\mathrm{candidate\_position}}$
    $\xi_c = \{\xi\}$ - $candidate$ ▷ subtract the candidate from total entity set
    $M_{\mathrm{score}} = CosineSimilarity(candidate, \xi_c)$
    $M_{\mathrm{score}} = max(0, M_{\mathrm{score}})$
    for $i \in length(M_{score})$ **do**
      $probability\_fitness_i = M_{\mathrm{score_i}} \div \sum_j M_{\mathrm{score_j}}$
        ▷ generate the fitness vector
    $selected\_entities = random\_choice(\xi_c, C, probability\_fitness)$
    $NegativeTriple_{(\mathrm{h',r,t'})} =$
    $FormNegative(t, selected\_entities, candidate\_position)$
      ▷ form C negative triples/positive
    $N_{\{\mathrm{h',r,t'}\}} = N_{\{\mathrm{h',r,t'}\}} \cup NegativeTriple_{(\mathrm{h',r,t'})}$
    ▷ append C negative triple per positive to the total batch negative set
  return $N_{\{\mathrm{h',r,t'}\}}$

Experiment

Hardware and Tools

- Core i7 4770 processor, 16 GB RAM, Nvidia RTX 2060 GPU
- Tools: Pytorch, Pandas, Numpy, Scipy,
- Model TransE [3] and DisMult [4]

Dataset

- Small to medium size data sets

Dataset	# of entity	# of relation	# Training triple	# Test triple	# Validation triple
UMLS	135	46	5216	661	652
Kinship	104	25	8544	1074	1068
Nations	14	55	1592	201	199

Table 1: Statistical information of the datasets.

Experiment

Evaluation Metrics

- Filter settings have been used for the standard knowledge graph embedding evaluation metrics.
  - Mean Rank (MR)
  - Mean Reciprocal Rank (MRR)
  - Hit@1
  - Hit@3
  - Hit@10

Results

Evaluation of Negative Sampling

Dataset	Random Negative	DNS	ADNS
Nations	MR: 5 MRR: 26.96% Hit@1: 0% Hit@3: 40.05% Hit@5: 60.95% Hit@10: 93.53%	MR: 5 MRR: 33.79% Hit@1: 13.68% Hit@3: 37.31% Hit@5: 59.95% Hit@10: 94.53%	MR: 4 MRR: 42.42% Hit@1: 20.15% Hit@3: 53.98% Hit@5: 73.88% Hit@10: 98.01%
Kinship	MR: 9 MRR: 25.66% Hit@1: 0.009% Hit@3: 39.80% Hit@5: 56.28% Hit@10: 76.44%	MR: 13 MRR: 24.89% Hit@1: 0.03% Hit@3: 38.13% Hit@5: 49.30% Hit@10: 64.20%	MR: 8 MRR: 28.47% Hit@1: 00.28% Hit@3: 47.77% Hit@5: 62.38% Hit@10: 78.91%
UMLS	MR: 3 MRR: 64.47% Hit@1: 39.86% Hit@3: 88.12% Hit@5: 93.95% Hit@10: 97.13%	MR: 3 MRR: 72.90% Hit@1: 56.81% Hit@3: 86.76% Hit@5: 92.06% Hit@10: 96.44%	MR: 2 MRR: 80.21% Hit@1: 64.45% Hit@3: 95.54% Hit@5: 97.05% Hit@10: 98.03%

Table 2: Evaluation of negative sampling of TransE

Results

Evaluation of Negative Sampling

Dataset	Random Negative	DNS	ADNS
Nations	MR: 3 MRR: 65.34% Hit@1: 49.50% Hit@3: 76.12% Hit@5: 88.31% Hit@10: 99.520%	MR: 2 MRR: 82.32% Hit@1: 73.63% Hit@3: 88.06% Hit@5: 94.53% Hit@10: 99.75%	MR: 2 MRR: 79.48% Hit@1: 68.91% Hit@3: 86.32% Hit@5: 93.78% Hit@10: 100.00%
Kinship	MR: 6 MRR: 48.45% Hit@1: 33.43% Hit@3: 54.24% Hit@5: 65.27% Hit@10: 85.38%	MR: 5 MRR: 57.19% Hit@1: 44.88% Hit@3: 61.78% Hit@5: 71.09% Hit@10: 83.80%	MR: 5 MRR: 58.74% Hit@1: 47.63% Hit@3: 61.87% Hit@5: 71.09% Hit@10: 85.94%
UMLS	MR: 8 MRR: 48.03% Hit@1: 34.72% Hit@3: 54.84% Hit@5: 64.07% Hit@10: 76.48%	MR: 5 MRR: 71.15% Hit@1: 63.69% Hit@3: 74.21% Hit@5: 78.97% Hit@10: 86.16%	MR: 6 MRR: 60.72% Hit@1: 49.24% Hit@3: 67.96% Hit@5: 74.96% Hit@10: 84.87%

Table 3: Evaluation of negative sampling of DisMult

Results

The figures show the convergence of loss function for UMLS data

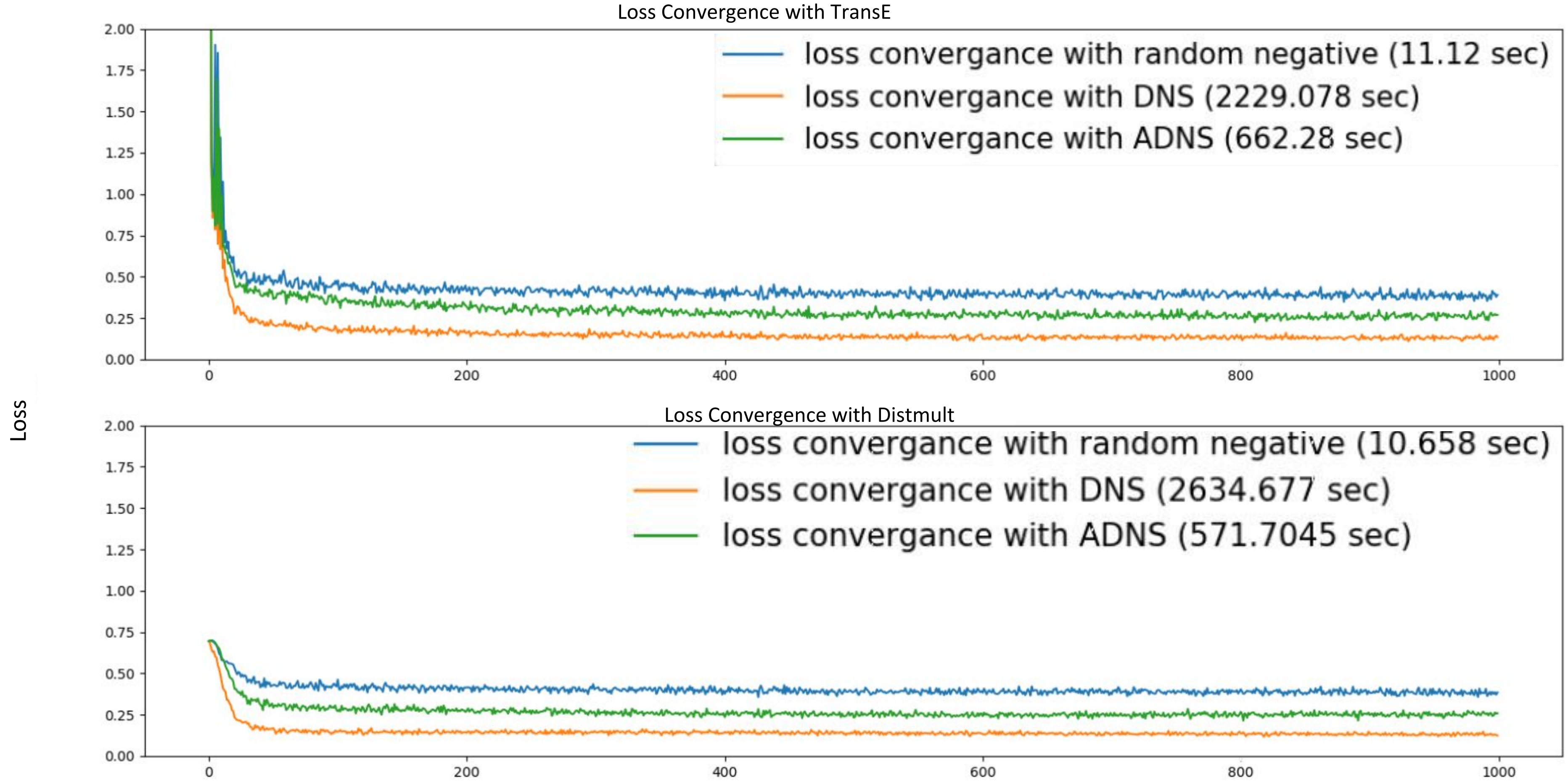


Figure 1: Convergence of loss with both models

Conclusions & Future work

- Proposed an effective and fast negative sampling method for embedding models
- The performance of the proposed approach is comparable with the existing approaches, while being less complex

Future Work

- Test on more recent KG embedding models. Example – Rotate [5], Tucker [6] or QuatE [7].
- Test with other similarity methods (Example:TF-IDF)
- Test with larger Data

REFERENCES

[1] Kotnis, B., & Nastase, V. (2017). Analysis of the impact of negative sampling on link prediction in knowledge graphs. arXiv preprint arXiv:1708.06816

[2] Dash, S., & Gliozzo, A. (2019). Distributional Negative Sampling for Knowledge Base Completion. arXiv preprint arXiv:1908.06178.

[3] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In Advances in neural information processing systems (pp. 2787-2795).

[4] Yang, B., Yih, W. T., He, X., Gao, J., & Deng, L. (2014). Learning multi-relational semantics using neural-embedding models. arXiv preprint arXiv:1411.4072.

REFERENCES

[5] Sun, Zhiqing, et al. "Rotate: Knowledge graph embedding by relational rotation in complex space." arXiv preprint arXiv:1902.10197 (2019).

[6] Balažević, Ivana, Carl Allen, and Timothy M. Hospedales. "Tucker: Tensor factorization for knowledge graph completion." arXiv preprint arXiv:1901.09590 (2019).

[7] Zhang, S.; Tay, Y.; Yao, L.; Liu, Q. Quaternion knowledge graph embeddings. In Proceedings of the 33th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada,

THANK YOU

Questions

Knowledge Graph Embedding Models

Distmult

1. The bilinear scoring function of DistMult model is obtained by multiplying their entity vectors (head and tail) with their corresponding relation matrix which is diagonal [5].
2. The entities are considered as ye1, ye2 and their corresponding diagonal relation matrix Mr, leads to the equation 1 [5].

5. Yang, B., Yih, W. T., He, X., Gao, J., & Deng, L. (2014). Learning multi-relational semantics using neural-embedding models. arXiv preprint arXiv:1411.4072.