Deep Learning: Assignment 2 Knowledge Graphs

Samuel Joseph Mcmurray

April 2023

This assignment is to learn to utilize knowledge graph embedding in deep learning, as to perform link prediction on the incomplete knowledge graphs. The knowledge graph that was used within the experiment is a biomedical knowledge graph called PharmKG which contains 500,958 triplets that are separated into a train, validation, and test dataset. The train dataset contains 400,788 triplets, the validation and test datasets containing roughly 50,000 triplets, the number of entities is 7601 with 29 relation types. The package that is to be used in the implementation of the training and testing is Python KnowlEdge EmbeddiNgs (PyKEEN), which has 44 different knowledge graph embedding models as well as a number of built in tools.

1 Task One

The first task is to implement a TransE model, then do hyper-parameter tuning of at least 2 parameters evaluating the model on the Mean Reciprocal Rank (MRR) or as referred to in the PyKEEN documentation the Inverse Harmonic Mean Rank. The MRR is evaluated from 0 to 1 where the closer to 1 the better the ranking of the model, the further away to score gets from 1 the further down the ranking of the actual correct entity. The first step in the implementation which followed the structure of the PharmaKG implementation is to get the triples and their relations from the dataset, once this is accomplished it is necessary to map the relations and entities to id's. In order to use the PyKEEN models the datasets need to be converted to a triple factory. The implementation of the base model is carried out through the structure and parameters that were presented in the PharmaKG experimentation with the number of embedding dimensions being 300, the batch size of 32, early stopping frequency of 10, and a learning rate of 0.1, with parameter selection based on the MRR performance. The model performed poorly with a MRR value of 0.0032. For the tuning of the hyper-parameters a simple grid search was performed on the batch size, the embedding dimensions and the learning rate. The embedding dim used values between 200 and 800 with a step size of 100, the batch size of 32 to 128 with a step size of 32, and the learning rate 1e-5 to 0.1 with a step of 1e+1. The hyper-parameter tuning resulted in the embedding dimension being 200, the batch size being 96, and the learning rate of 0.001, with an MRR of 0.056.

2 Task Two

The second task was to implement the STransE model by extending the PyKEEN package, the STransE as described in Nguyen et al.[1] is a combination of the Structured Embedding model and the TransE model. The STransE model was

implemented within pykeen by creating an interaction class that uses the forward function from the abstract interaction class. The forward function takes the head, relation, and tail as the arguments. In the STransE model the weights of the relation for both the head and the tail need to be used, this is done by separating the relation into a relation head and relation tail tensor. The calculation is then done by taking those weights of the head relation by the head plus the relation minus the relation of the tail by the tail, then the negative norm is applied. The the STransE class with the ERModel as the abstract class from PyKEEN setting the interaction variable as the interaction class that was implemented. The 2 experiments that were conducted in this class were done using the same variables as the previous tasks with the base STransE model having an MRR of 0.027, and the hyper parameter model having an MRR of 0.044. It appears from the results that the original TransE model preformed better on the dataset when MRR was used although the hyper-parameters were tuned only on the previous model so it is possible that a better performance could be reached by the STransE model given the tuning was conducted.

3 Task Three

In the third task was to use the Tensorboard Embedding projector to project the models into a three-dimension space with the t-SNE plot and a 100 nearest neighbors of an entity view. This was done using the torch built in functionality with tensorboard utilizing the summary writer on each of the models to produce a metadata.tsv of the labels and a tensors.tsv of the embedding tensors. The base TransE model producing more tight knit clusters with a tighter nearest neighbor than the other t-SNE projections, the hyper-parameter tuned TransE producing a blob similar to a front part of a whale with little no cluster separation. The same can be seen with a different shape for the STransE tuned model, the STransE appears the worst in this sense as its just a spherical cluster with the nearest neighbors projected all over the with no clustering.

4 Conclusion

The conclusion being that in terms of projection the base model seems to have done the best while, the TransE hyper-tuned model had better performance in terms of MRR. The STransE model showed no improvement and the projections of the base model were the worst in terms of clusters and the distance of the nearest neighbors were projected too from the original.

References

[1] Dat Quoc Nguyen et al. “STransE: a novel embedding model of entities and relationships in knowledge bases”. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, June 2016, pp. 460–466. DOI: 10.18653/v1/N16-1054. URL: https://aclanthology.org/N16-1054.