Professorship of Data Mining and Analytics
Department of Informatics
Technical University of Munich

Large-Scale Graph Mining

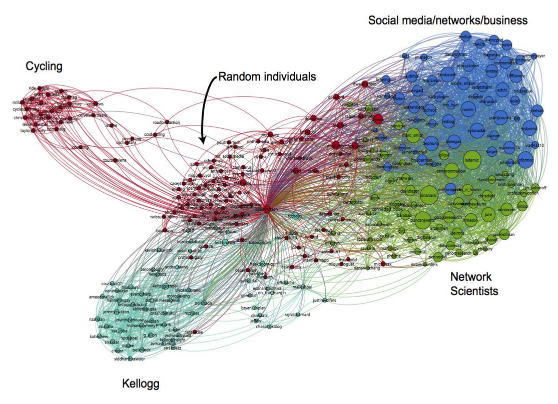
Machine Learning Lab Course – Summer Term 2019

Lukas Frickenstein, Andreas Grafberger, Paul Ursulean

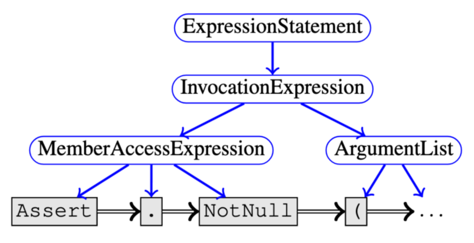
Deep Learning on Graphs

Motivation

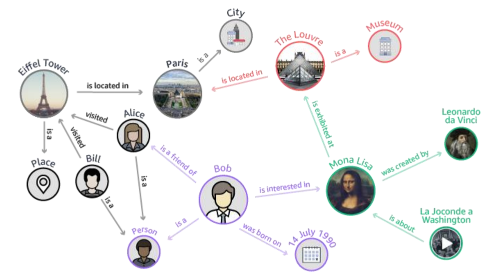
Many relevant problems either come in the form of graphs or can be represented as such.



Social Network Graph
Source: social-dynamics.org

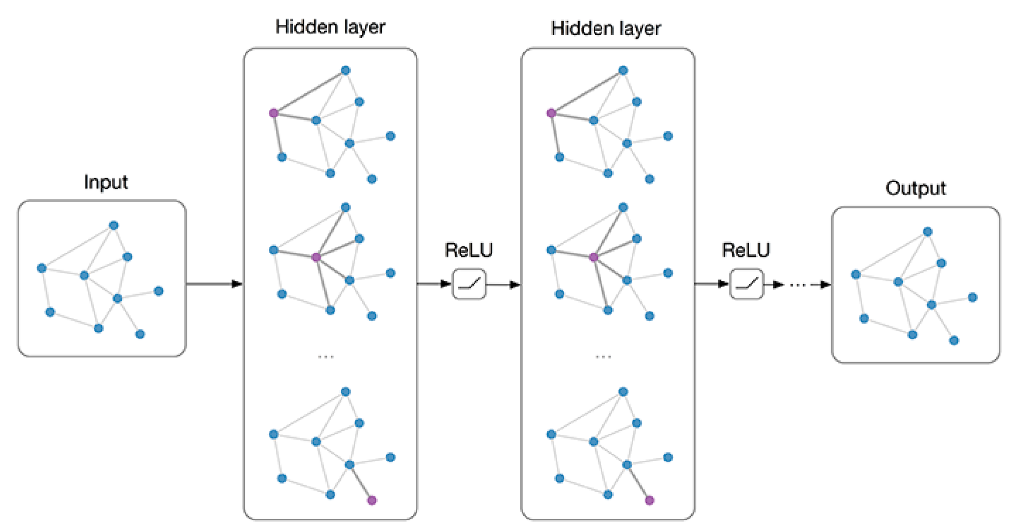


Code as Abstract Syntax Tree[1]



Knowledge Graph
Source: aws.amazon.com

Recent research has successfully started applying Deep Learning to various graph-structured problems. Graph Neural Networks (GNNs) operate on graphs by aggregation of embeddings from connected nodes. This process is called message passing.



Message passing visualized

Figure 1: For each node, all embeddings of its neighbors are aggregated and passed through a neural network. For each layer in such a network, information from each node reaches nodes one hop further away. Source: tkipf.github.io

GNN research is accelerating quickly. However, many questions remain unanswered.

This project's focus is:

1. Discovering relationships between objects in complex systems using GNNs.
2. Tackling scalability issues that arise when operating on large graphs.

Neural Relational Inference

Motivation

Complex systems can often be represented as interactions between their constituent primitives. This hierarchical complexity allows for layers of abstraction, a fundamental concept not only in computing but also in nature. Learning these interactions can give rise to insights into more complex phenomena, and can serve as a stepping stone to model these behaviours more effectively. Here, interactions can be represented as graphs with a variable number of relationship types.

Example Scenarios

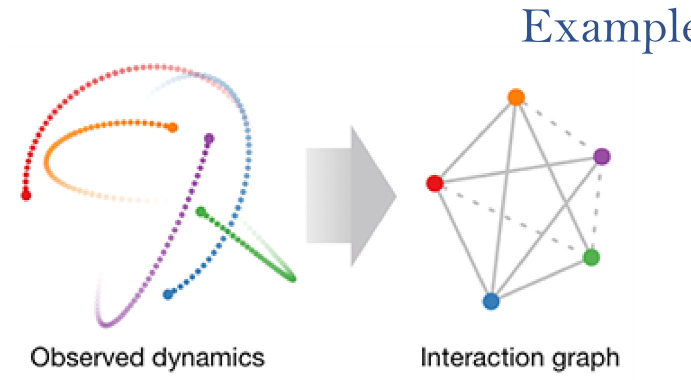


Figure 2: Dynamic system of particles connected by invisible springs (left), represented as a latent interaction graph (right). [2]

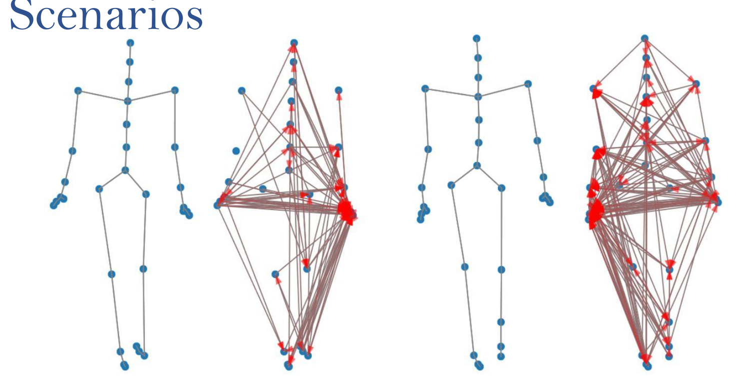


Figure 3: Learned latent interaction graph of the human body based on motion capture data. [2]

Neural Relational Inference Model by Kipf et al. [2]

The NRI is a model based on the Variational Auto-Encoder and the Graph Neural Network. It learns to infer interactions of parts in a system while simultaneously learning the dynamics purely from observational data.

- Encoder: Predicts pairwise interactions given the observed states.
  If no other information is given, it performs message-passing on a fully-connected graph.
- Latent interaction graph: Categorical distribution for each potential pairwise interaction.
- Decoder: Predicts future states given the sampled interaction graph and past sequences.

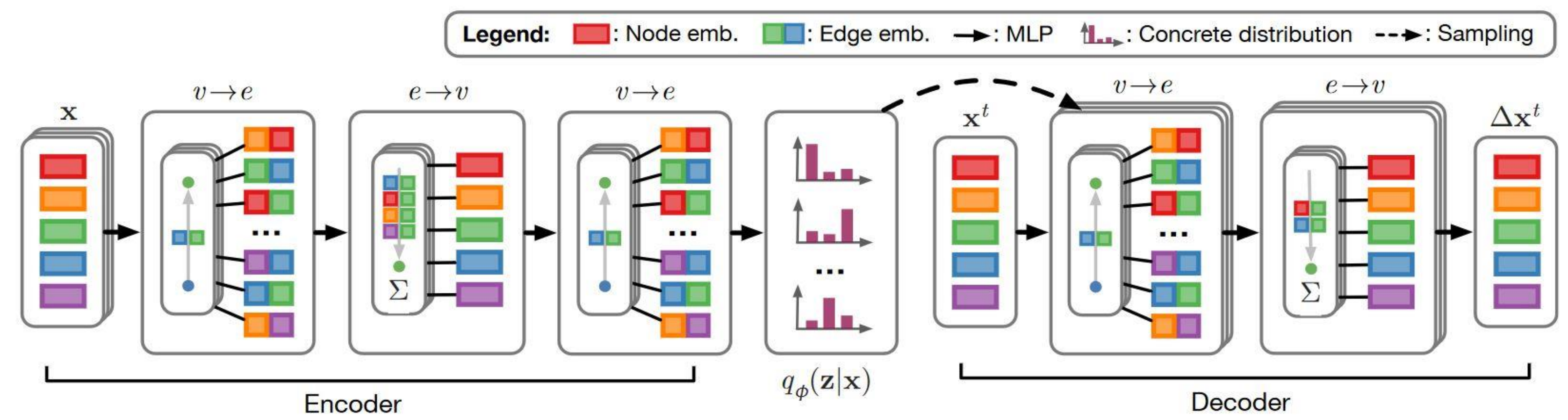


Figure 4: General structure of the NRI model with its respective encoder and decoder components. [2]

Optimizing the ELBO $\mathcal{L} = \mathbb{E}_{q_{\phi(z|x)}}[\log p_\theta(x|z)] - KL[q_\phi(z|x)||\, p_\theta(z)]$:

- Forces the model to use the latent graph to predict future states.
- Can enforce sparsity by modifying the prior $p_\theta(z)$.

Conclusion

- NRI model can often predict future states better than other baselines.
- Latent graph is interpretable and can help humans better understand various systems.

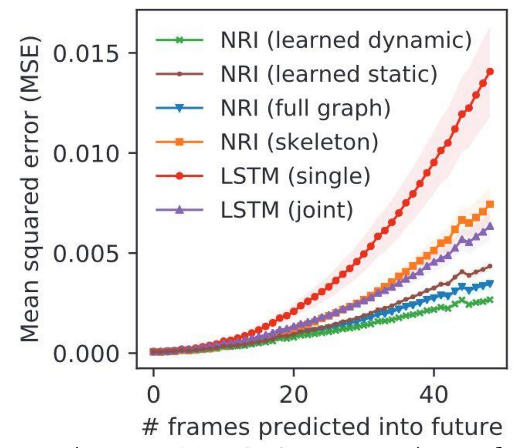


Figure 5: Test MSE comparison of different models in one domain.[2]

Applying NRI Model to Meteorology Domain

- Goal: Move towards arbitrary sequential data with multiple entities and see if the found latent interactions are interpretable or improve weather forecasts.
- Data set comprises different configurations of five weather stations in Spain, each measuring sequences of daily temperature over 100 days. An example sequence is displayed on the right in Figure 6.

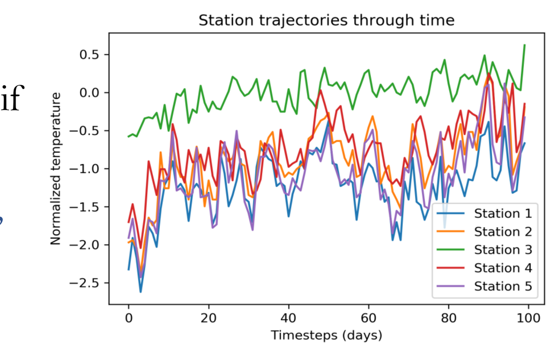


Figure 6: Example sequence from our data set.

Learned Relationships vs. True Distances

- Error function measures how well each learned relation type rel corresponds with the real distance between stations i, j.

$$\mathcal{E}_{i,j,rel} = \begin{cases} d_{graph}(i,j,rel), for\ normal\ rel. types \\ 1 - d_{graph}(i,j,rel), for\ None\ relations \end{cases}$$

with $d_{graph}(i,j,rel) = |(1 - d_{norm}(i,j)) - graph_{rel}(i,j)|$

- We also apply this error function to the correlation matrix for comparison. Results are shown in Table 1.

	Error w.r.t real distance
NRI	0.3864
Correlation	**0.1354**

Table 1: Custom error function applied to the latent graph. Lower values are better. Latent interaction types of NRI are worse at predicting station distance than the simple correlation matrix.

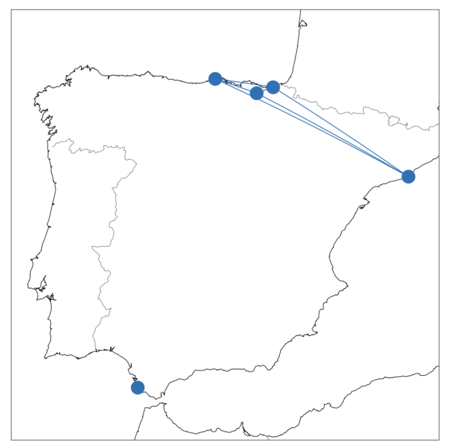


Figure 7: Latent interaction graph of five weather stations. The station far into the south is correctly predicted to have no interaction with others.

Conclusion

- NRI improves the capability to forecast weather further into the future.
- Prediction generally suffers from very challenging and noisy weather domain.
- Latent graph often shows difficult to interpret patterns that very likely stem from noise in input sequences.
- In most cases, the simple pairwise correlation is more interpretable.

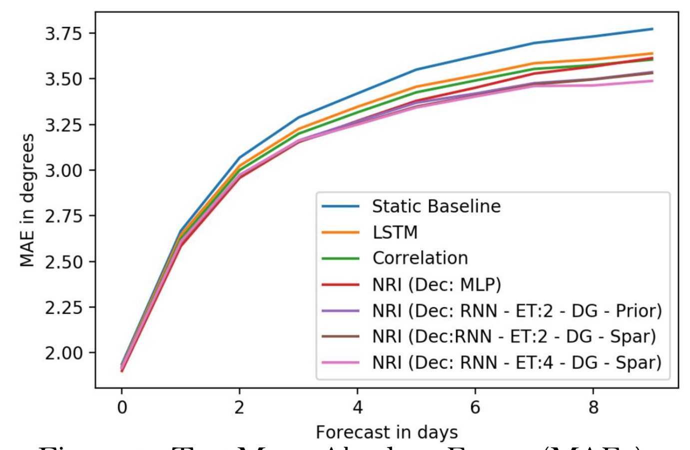


Figure 8: Test Mean-Absolute-Errors (MAEs) in °C for different model setups over increasing forecasting days.

Scalability

Motivation

The previous NRI experiments revealed an obvious bottleneck in pairwise node operations, leading to $O(N^2)$ complexity in the number of nodes. To isolate this problem, we consider a simpler model that suffers from similar problems.

Variational Graph Auto-Encoder

- The Variational Graph Auto-Encoder[3] (VGAE) by Kipf & Welling seeks to recover a graph's adjacency matrix by learning node embeddings.
- The probability that two nodes are connected is proportional to their similarity in the embedding space.

More formally:

- X is the node feature matrix and A the adjacency matrix.
- Encoder learns node embeddings:

$$q(z_i|\,X, A) = \mathcal{N}(z_i\,|\,\mu_i, diag(\sigma_i^2))$$
$$\mu = GCN_\mu(X, A)\ \ \log\sigma = GCN_\sigma(X, A)$$

- Decoder recovers graph edges:

$$p(A_{ij} = 1\,|\,z_i, z_j) = \sigma(z_i^T z_j)$$

However, performing this pairwise comparison of embeddings leads to **$O(N^2)$ complexity** and is **infeasible to compute** for larger graphs.

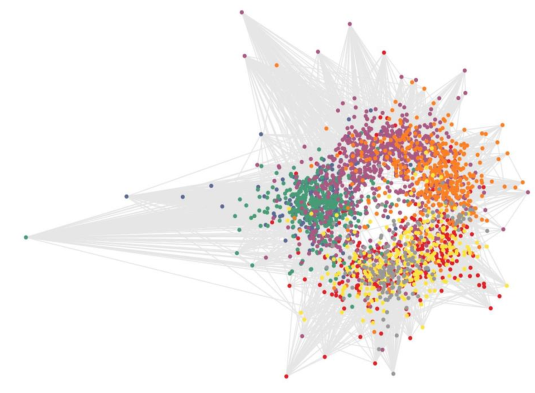


Figure 9: Two-dimensional embedding space with corresponding connections. Labels (colors) not shown during training. [3]

Avoiding Quadratic Complexity via LSH

- Locality Sensitive Hashing functions approximate distance in the expectation.
- Different hashing schemes correspond to different distance functions.
- $O(N)$ approximate duplicate detection.

Distance Measure	Corresponding LSH
Cosine Similarity	Random Hyperplanes
Euclidean Similarity	Random Projections
Dot Product	Augmented Random Hyperplanes[4]

Incorporating LSH into the VGAE Model

- Find optimal distance threshold for a given pair to be considered connected.
- Use threshold to tune LSH hyperparameters such that it is sensitive to the right range.

Reconstructing Full Naïve Matrix	
Memory Savings	98.28%
Precision	98.35%
Recall	99.83%

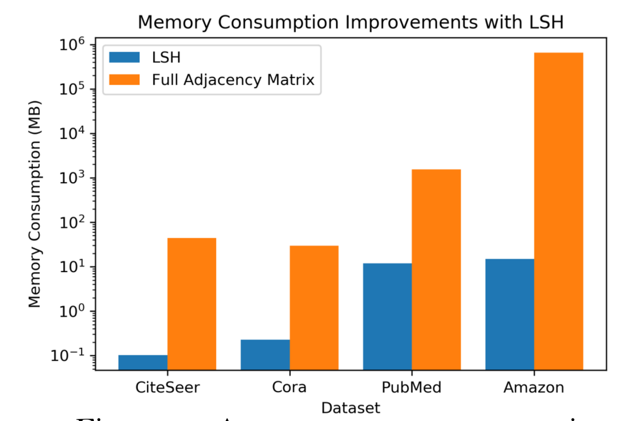


Figure 10: Average memory consumption improvements for all LSH runs.

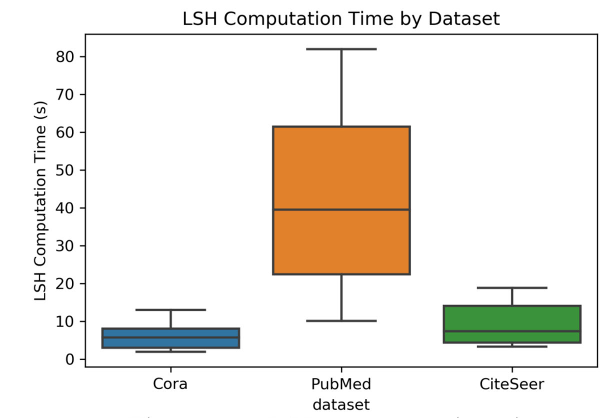


Figure 11: LSH computation time distribution over all runs.

Conclusion

- LSH drastically improves memory consumption for the VGAE model, enabling operation on large graphs which would not fit into memory otherwise, as seen in Figure 10.
- These improvements are observed even when fully reconstructing the naïve matrix, although doing so constitutes a time-memory trade-off, as seen in Figure 11.

References

[1] M. Allamanis, M. Brockschmidt, and M. Khademi, 'Learning to Represent Programs with Graphs', in *International Conference on Learning Representations*, 2018.
[2] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, 'Neural Relational Inference for Interacting Systems', in *Proceedings of the 35th International Conference on Machine Learning*, Stockholmsmässan, Stockholm Sweden, 2018, vol. 80, pp. 2688–2697.
[3] T. N. Kipf and M. Welling, 'Variational Graph Auto-Encoders', *arXiv:1611.07308 [cs, stat]*, Nov. 2016.
[4] B. Neyshabur and N. Srebro, 'On Symmetric and Asymmetric LSHs for Inner Product Search', *arXiv:1410.5518 [cs, stat]*, Oct. 2014.