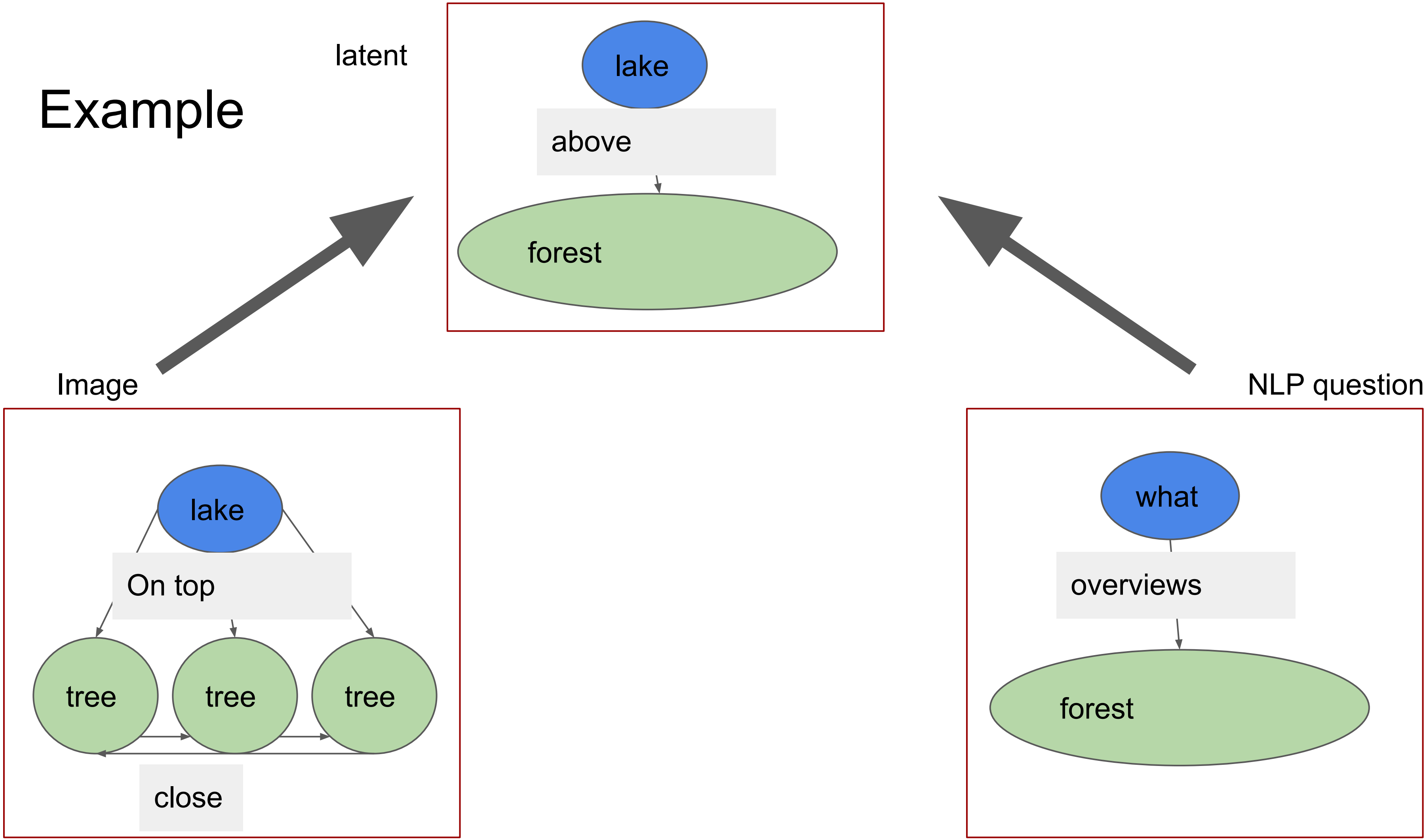
Multimodal architecture

Jannik Gut

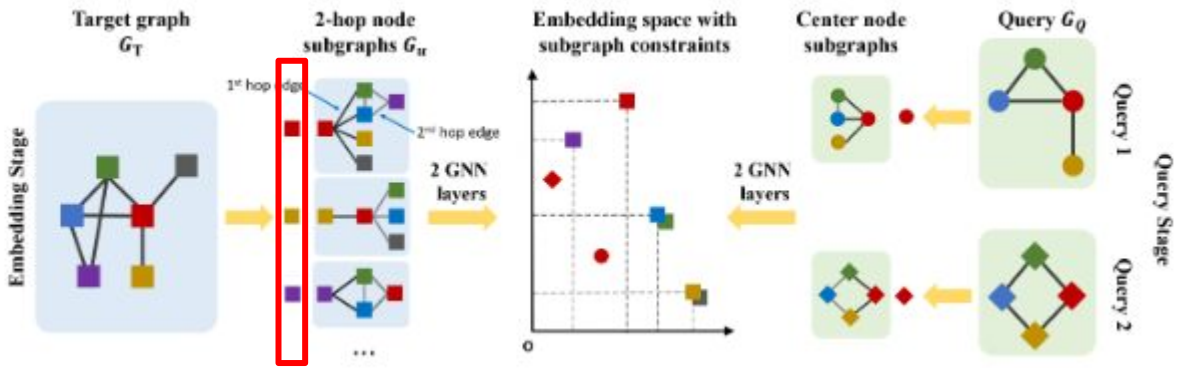
Overview

- Example
- *Overleaf*
- Papers
- Research
- Questions

Example


latent
lake
above
forest
Image
lake
On top
tree
tree
tree
close
NLP question
what
overviews
forest

NeuroMatch



- Message Passing
- Curriculum training
-
- Check **all neighbourhoods** and try to match them in embedding space
    - **The amount of neighbourhoods does not change**
- Much faster compared to combinatorial
-
- Custom Hinge loss in embeddings
    - Z is 2-dimensional

$$\mathcal{L}(z_q, z_u) = \sum_{(z_q, z_u) \in P} E(z_q, z_u) + \sum_{(z_q, z_u) \in N} \max\{0, \alpha - E(z_q, z_u)\}, \text{where}$$

$$E(z_q, z_u) = ||\max\{0, z_q - z_u\}||_2^2$$

Graph Auto Encoder

- Exists
  - by Kipf & Welling
- Try to **model the adjacency matrix A**
  - Amount of nodes given
- μ,σ estimated with GCN
- Every node has a z
- p(Z) was a weakness
  - Keep graph sparse

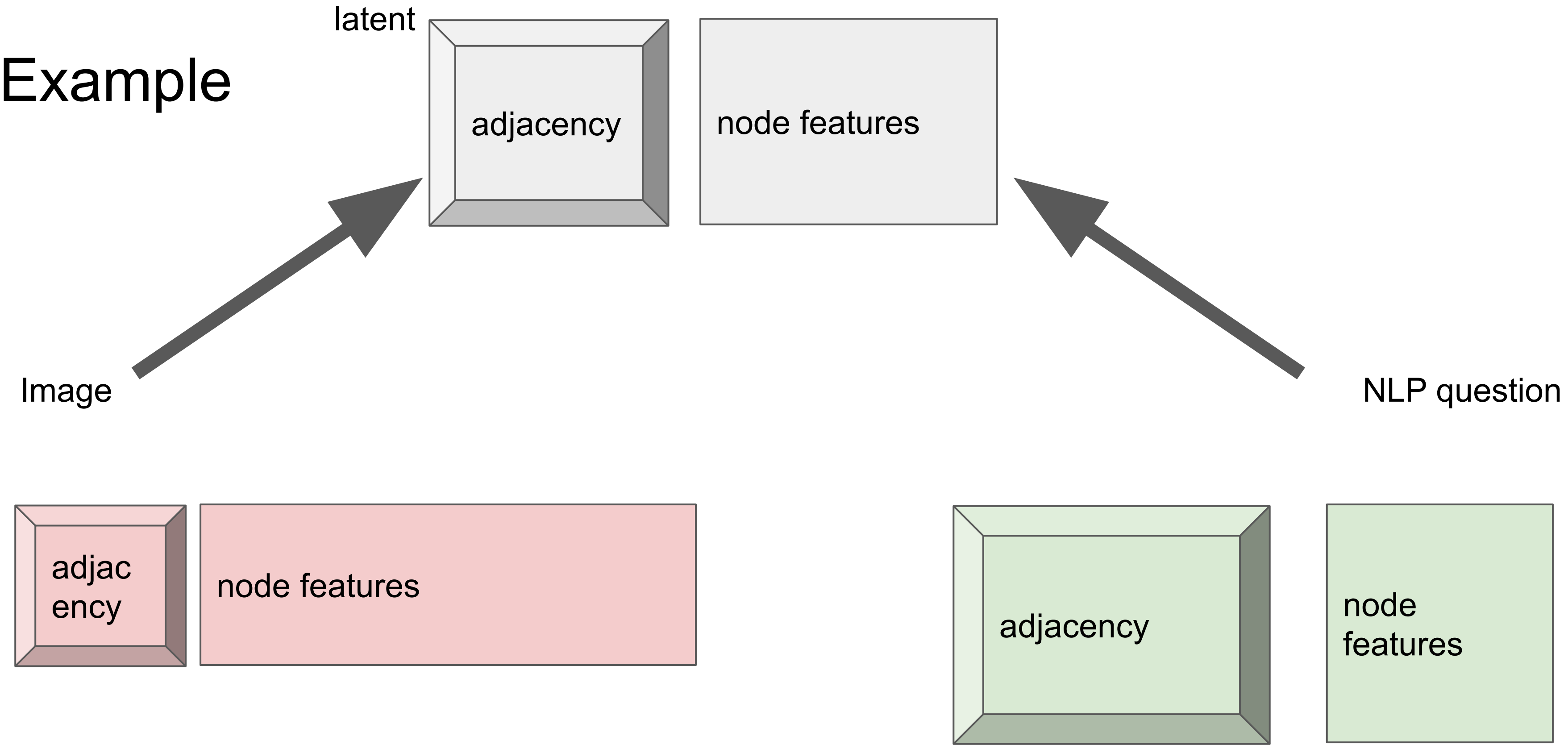
$$q(\mathbf{Z} \mid \mathbf{X}, \mathbf{A}) = \prod_{i=1}^{N} q(\mathbf{z}_i \mid \mathbf{X}, \mathbf{A}), \quad \text{with} \quad q(\mathbf{z}_i \mid \mathbf{X}, \mathbf{A}) = \mathcal{N}(\mathbf{z}_i \mid \boldsymbol{\mu}_i, \operatorname{diag}(\boldsymbol{\sigma}_i^2)).$$

$$p(\mathbf{A} \mid \mathbf{Z}) = \prod_{i=1}^{N} \prod_{j=1}^{N} p(A_{ij} \mid \mathbf{z}_i, \mathbf{z}_j), \quad \text{with} \quad p(A_{ij} = 1 \mid \mathbf{z}_i, \mathbf{z}_j) = \sigma(\mathbf{z}_i^\top \mathbf{z}_j),$$

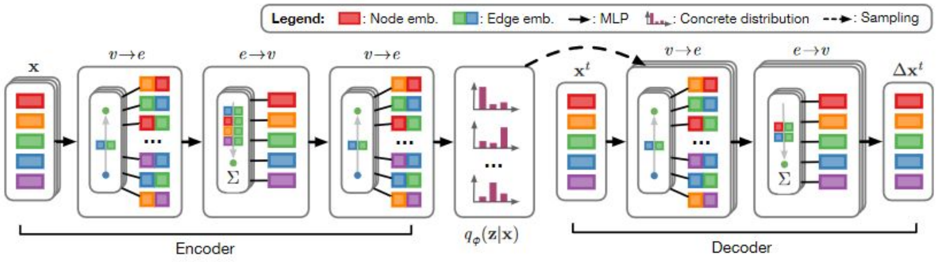
$$\mathcal{L} = \mathbb{E}_{q(\mathbf{Z}|\mathbf{X},\mathbf{A})}\big[\log p(\mathbf{A} \mid \mathbf{Z})\big] - \mathrm{KL}\big[q(\mathbf{Z} \mid \mathbf{X}, \mathbf{A}) \,||\, p(\mathbf{Z})\big],$$

$$\hat{\mathbf{A}} = \sigma(\mathbf{Z}\mathbf{Z}^\top), \quad \text{with} \quad \mathbf{Z} = \mathrm{GCN}(\mathbf{X}, \mathbf{A}).$$

Example


latent
adjacency
node features
Image
NLP question
adjac
ency
node features
adjacency
node
features

Neural Relational Inference



- Message passing
- Fully connected graph
  - But some edges are "no edge"
- Edges have types, which enforce a different cell architecture
-
- Encoder is a GNN, as in the image
  - Z has an entry for each edge
- Z is a graph
- Decoder predicts multiple timesteps from a single latent representation
  - Also is a GNN, can be extended with RNN cells

The ELBO objective, Eq. 3, has two terms: the reconstruction error $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]$ and KL divergence $\mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})]$. The reconstruction error is estimated by:

$$-\sum_j \sum_{t=2}^{T} \frac{||\mathbf{x}_j^t - \boldsymbol{\mu}_j^t||^2}{2\sigma^2} + \text{const} \qquad (18)$$

while the KL term for a uniform prior is just the sum of entropies (plus a constant):

$$\sum_{i \neq j} H(q_\phi(\mathbf{z}_{ij}|\mathbf{x})) + \text{const}. \qquad (19)$$

Papers

- Graph Transformer Networks
    - Learn metapaths, no metanodes, Laplacian method
- Dynamic joint variational graph autoencoders
    - **Already know** all next timesteps and predict them jointly
- Data-driven graph construction and graph learning: A review
    - Data points (e.g. patients) are embedded in a high dimensional space as nodes
    - Create edges based on kNN-ish approaches
    - Create edge weights
        - Similarity
        - Pearson correlation
        - OLS recreation of center with neighbours
    - Multigraph learning tries to combine some graphs with the same nodes to give each graph one weight for the edges
- Latent-Graph Learning for Disease Prediction
    - Different patients act as nodes, edges based on similarity, use a GCN to predict afterwards.
- A comprehensive Survey of Graph Embedding: Problems, Techniques and Applications
    - A graph can be embedded
        - Each node
        - Each edge
        - Each substructure **(node with extra steps)**
        - Global
        - **No graph**
- Hierarchical graph embedding in vector space by graph pyramid
    - Paywall

Research

- Matrix autoencoder
  - Smaller, quadratic matrix
  - PCA vectors are not quadratic and have other constraints
- Image that uses a smaller latent image
  - Convolution/spatial special kind of graph

Graph actions

- Structure
  - Add/remove edges
  - Add/remove nodes
- Content
  - Update edges
  - Update nodes
  - (Update universal graph)

Add/Remove Edges

- Add edges
- Remove edges
- → Fully connected graph with typed edges
    - “No edge”-type (separate neural net/gate for that)
- → otherwise not all new connections always possible
    - Neighbourhood assumptions
        - Neighbours of my neighbours have bigger chance of being my neighbour
            - Assumption data dependent (?)
        - Only look at some types
            - Assumption data dependent (?)
    - Sampling
        - What policy?

Add/Remove Nodes

- Add nodes
  - Where?
- Remove nodes
  - Tell neighbours?
- → Actions on edges
  - Add → Transform edge into new node
    - Node (+ adjacent edge) properties from original edge ends
    - How to add edge on the perimeter of the graph?
      - Self-edge?
  - Remove → Edge contraction
    - Node (+ adjacent edge) properties from original edge ends
    - How to deal with multiple neighbouring contractions?
- → Ordering, in general
  - Only one round of actions
    - Implications on amount of layers vs. size(difference) of graphs
  - Actions on new nodes+edges as well?
    - Sequential computation
  - → Test out, I guess

Content updates

- “As usual”
    - How to take new entities into account?
        - Compute structure gates before content
    - Different handling for new entities?
        - Pre-update?
    - Possible to implement “as usual”?
        - Changing graph structure

Expansion/Clumping layers

- Clumping → Summarising (supplementary information)
  - Clumping nodes
  - Removing edges
- Expansion → Generation (complementary information)
  - Adding nodes
  - Adding edges
- Stack together to form one layer
- Differentiation needed?
  - Makes some assumptions/computations easier

Assumptions/Support

- Contractions/Additions can and should utilise neighbourhood
- Scaling, either
  - The size progression of the graph is known in logarithmic measure
    - 4 layers → one edge can turn into maximally 7 nodes
  - Sequential computation works and is feasible
- We have enough resources for a fully connected graph

- Supports heterogeneity (as part of the state/features) and directions

Questions / Open problems for model

- How to exactly incorporate structural change?
  - Two phases
    - Structure
    - Update
- Ordering of actions
- How to deal with multiple adjacent contractions?
  - Has to be unordered
  - (Scalability) an issue in practice?
- Expanding and concatenation of same edge a problem?
- Exact implementation
  - Which GNN implementation?
  - More problems to follow, for sure

Losses

- Structure
    - Graph edit distance
        - NP hard to check
    - Predict adjacency matrix
        - Not a graph problem, but a matrix problem
        - Use 0-padding?
- Content
    - “As usual”
        - Cross-entropy, LS etc.
    - How to enforce good matching to structure
        - E.g. what in a ring
        - Given in an ordered matrix
- What do if not correct structure?
    - Take best graph edit distance and those, that can’t be mapped have bad luck?
- Balance structure vs. content loss with hyperparameter

Dataset

- NLP annotations
  - Dependency parsing vs. constituency parsing
    - Possibly same type, but different systems
  - Big enough, should be able to find easier and harder sentences
- More research

Other questions

- Good, easy dataset
  - NLP annotations
    - Too many edge types?
- How to tackle losses
  - Something instead of GED?
  - Matching
  - Bad structure?
- Adjacency matrix reasonable?
  - Padding?