# progress of deep learning on graph

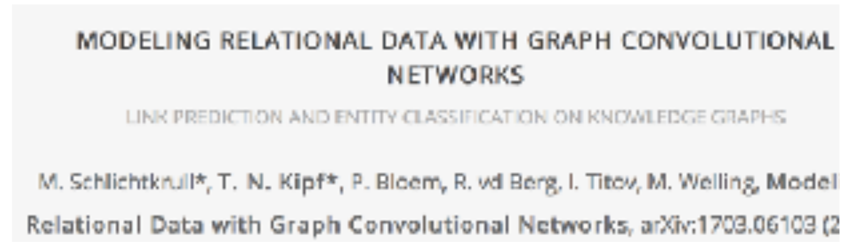
# sources

- Third Representation Learning for Graphs Workshop (ReLiG 2017)
- https://github.com/thunlp/NRLPapers
- https://truyentran.github.io/repLearn.html
- http://geometricdeeplearning.com/

# agenda

- Modeling Relational Data with GCN
- Motif-aware graph embeddings
- Graph Convolutional Matrix Completion
- Geometric deep learning: going beyond Euclidean data

# Modeling Relational Data with GCN


MODELING RELATIONAL DATA WITH GRAPH CONVOLUTIONAL NETWORKS

LINK PREDICTION AND ENTITY CLASSIFICATION ON KNOWLEDGE GRAPHS

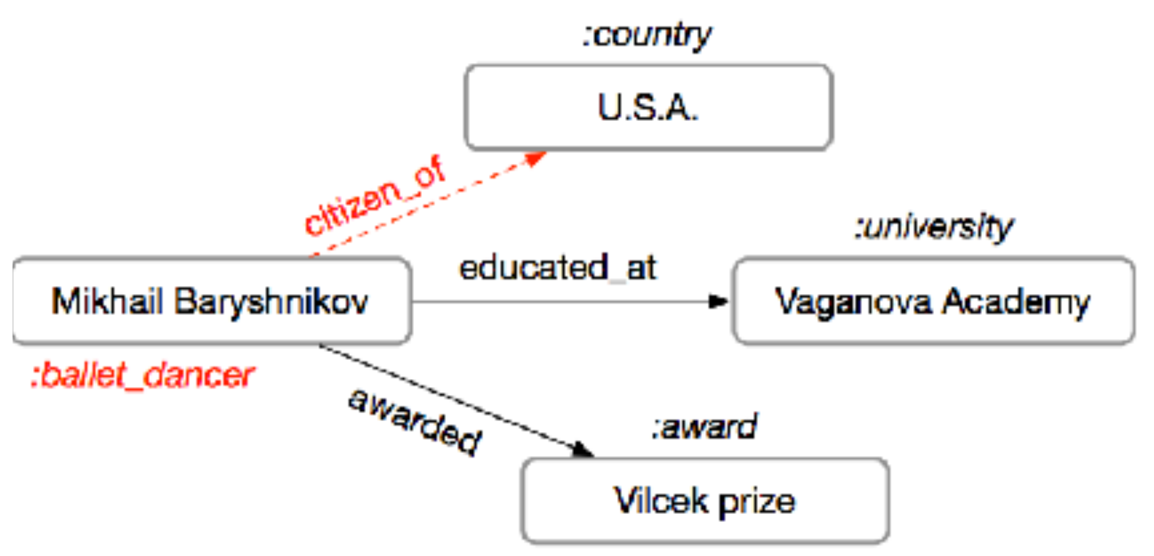
M. Schlichtkrull*, T. N. Kipf*, P. Bloem, R. vd Berg, I. Titov, M. Welling, Model Relational Data with Graph Convolutional Networks, arXiv:1703.06103 (2

**By the author of GCN**

**link**

**Contribution: consider relation and entity classification at the same time**
**Pro: utilize mutual information between relation and entity**

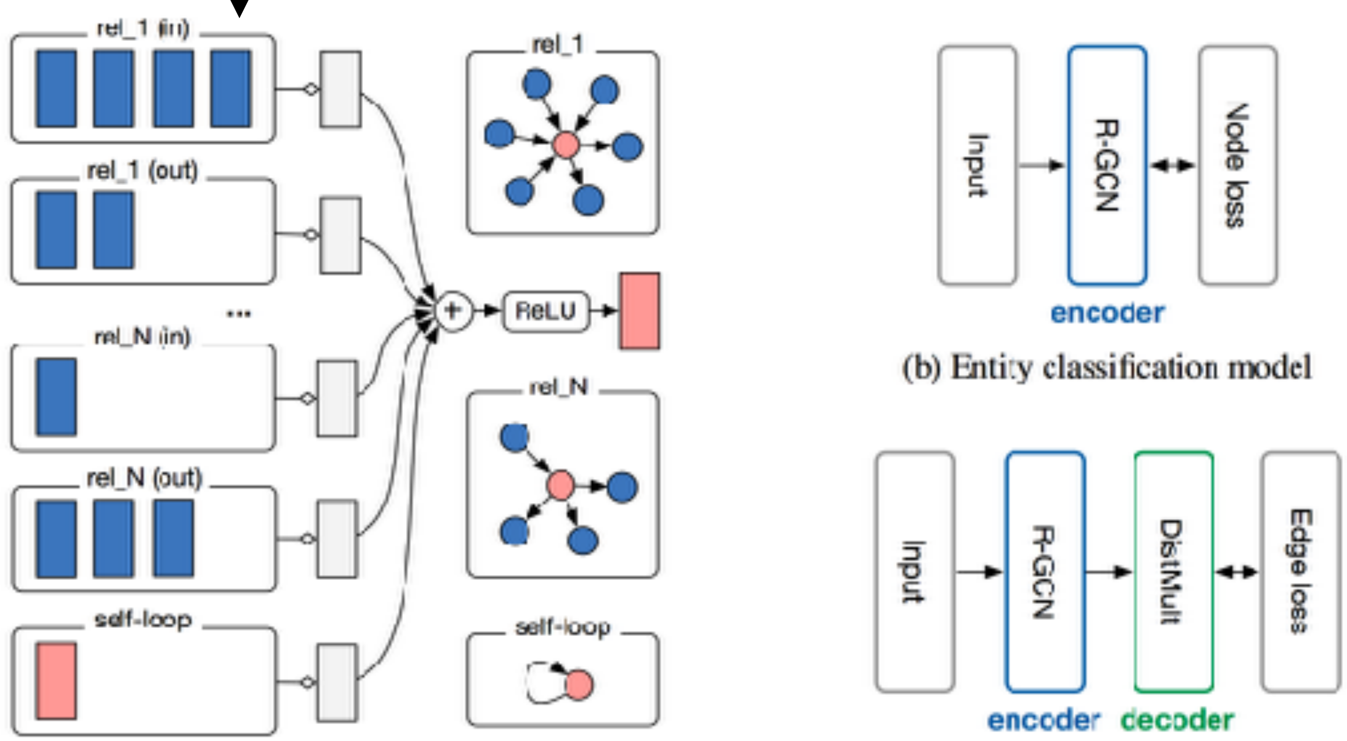
- **Concept**
  - Predicting missing information in knowledge bases is the main focus of statistical relational learning (SRL).
- **Motivation Example:**
  - Vilcek prize (an award honoring contributions of immigrants to the US society) implies having the US citizenship,
  - graduating from the Vaganova Academy probably means that the entity is a ballet dancer.



# Modeling Relational Data with GCN

**traverse all types of relations and neighbors**

$$h_i^{(l+1)} = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i^r}\frac{1}{c_{i,r}}W_r^{(l)}h_j^{(l)} + W_0^{(l)}h_i^{(l)}\right)$$




(a) Single R-GCN layer

(b) Entity classification model

(c) Link prediction model

# Modeling Relational Data with Graph Convolutional Networks

- **Methods for tasks:**

**DistMult** $f(s, r, o) = e_s^T R_r e_o$

triple $(s, r, o)$

(subject, relation, object)

- Link prediction

$$\mathcal{L} = -\frac{1}{(1+\omega)|\hat{\mathcal{E}}|} \sum_{(s,r,o,y)\in\mathcal{T}} y \log \sigma\big(f(s,r,o)\big) + (1-y) \log\big(1 - \sigma\big(f(s,r,o)\big)\big)$$

$\omega$ negative sample rate

$|\hat{\mathcal{E}}|$ Edge number

- Entity classification

$$\mathcal{L} = -\sum_{i\in\mathcal{Y}} \sum_{k=1}^{K} t_{ik} \ln h_{ik}^{(L)}$$

`categorical cross entropy`

$t_{ik}$ denotes its respective ground truth label.

$h_{ik}^{(L)}$ is the $k$-th entry of the network output

# Motif-aware graph embeddings

Contribution: motif-based graph classification
Pro: motif level information

## Significant graph

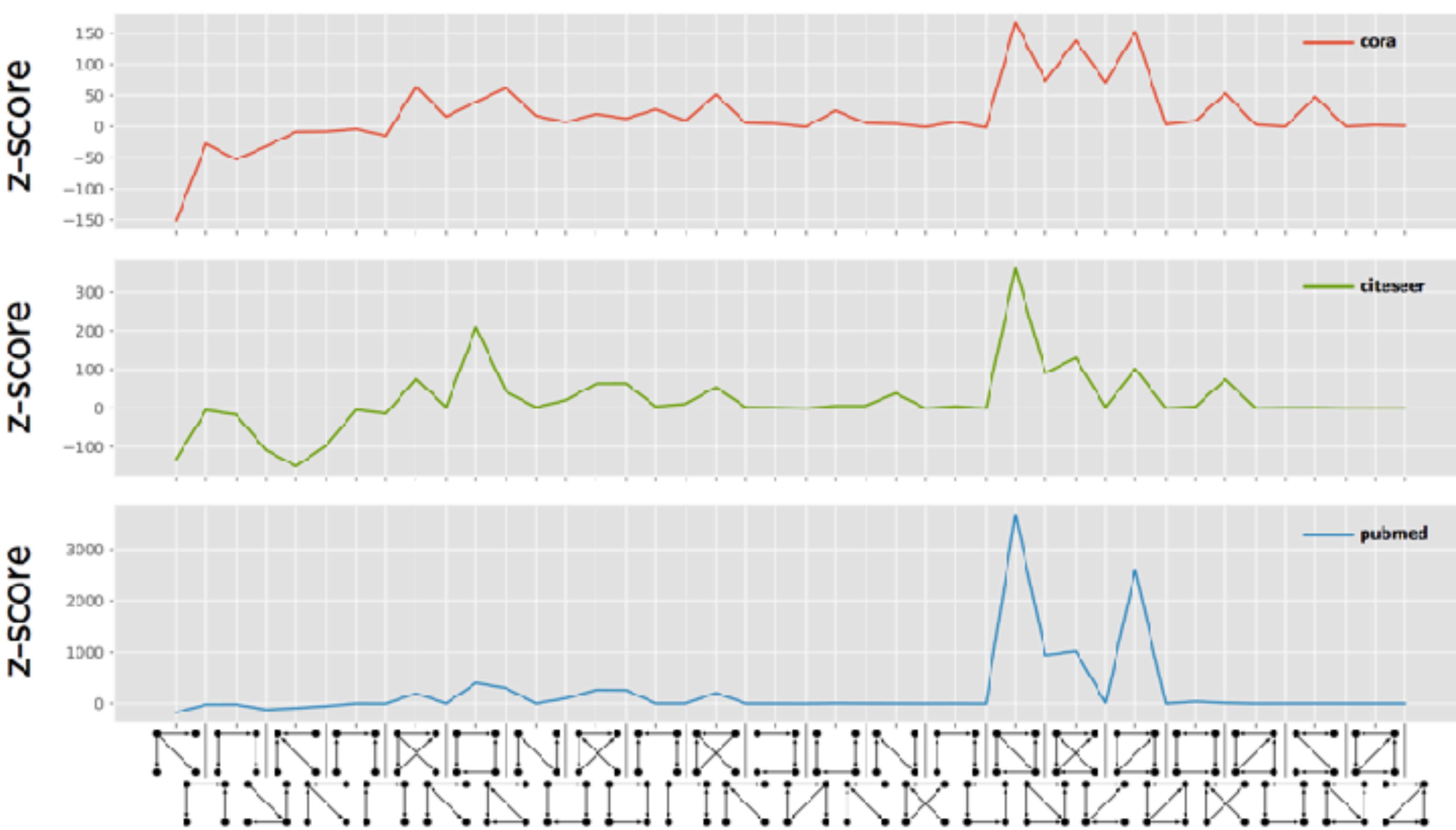
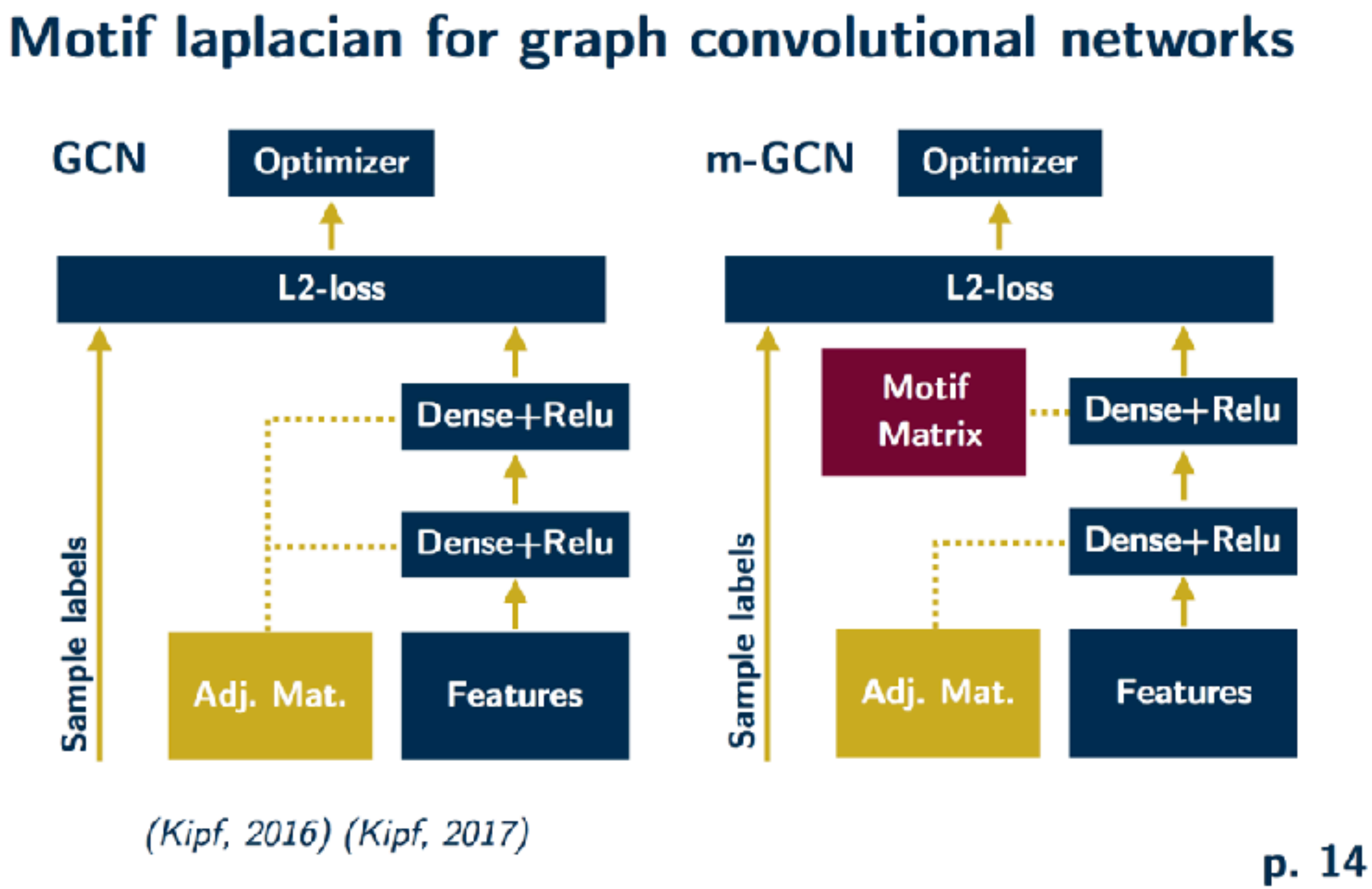



*Fig. 6 - Z-scores for motifs of size-4 in Citation networks*

# Motif-aware graph embeddings




**Motif co-occurance matrix**: select significant motifs, count if nodes exist in the same motif

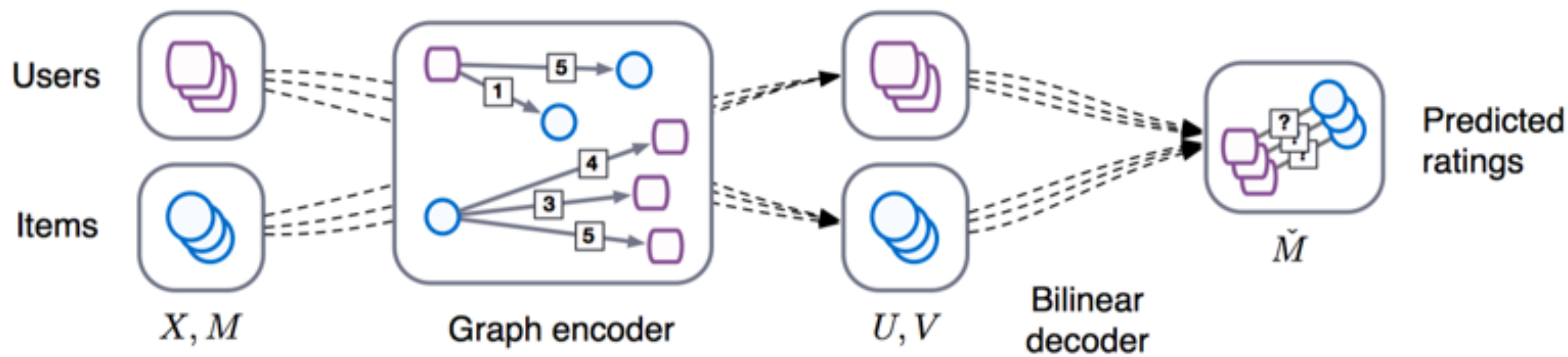
# How Much Information Does the Machine Need to Predict?

Y LeCun

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Unsupervised/Predictive Learning (cake)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

# Graph Convolutional Matrix Completion

Contribution: unsupervised graph completion
Pro: alleviate the pain of requiring mass data




of 1) a graph encoder model $Z = f(X, A)$, which take as input an $N \times D$ feature matrix $X$ and a graph adjacency matrix $A$, and produce an $N \times E$ node embedding matrix $Z = [z_1^T, \ldots, z_N^T]^T$, and 2) a pairwise decoder model $\check{A} = g(Z)$, which takes pairs of node embeddings $(z_i, z_j)$ and predicts respective entries $\check{A}_{ij}$ in the adjacency matrix. Note that $N$ denotes the number of nodes, $D$ the number of input features, and $E$ the embedding size.

**Variational Graph Auto-Encoders**, https://arxiv.org/pdf/1611.07308.pdf

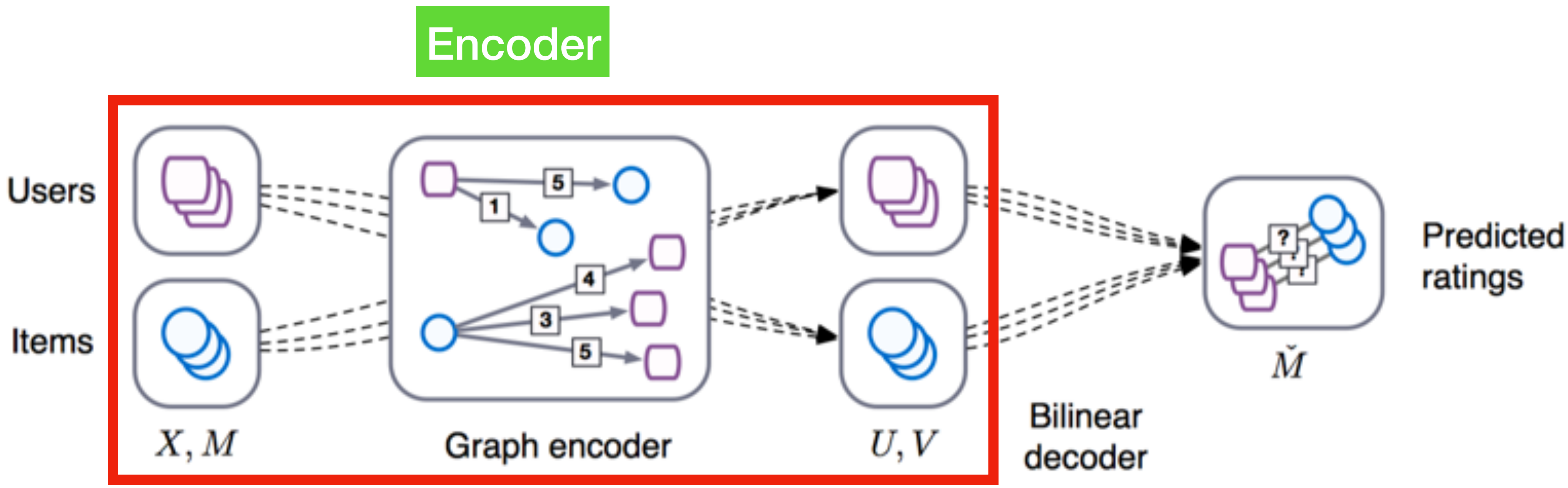
Encoder Decoder

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

**Non-probabilistic *graph auto-encoder* (GAE) model** For a non-probabilistic variant of the VGAE model, we calculate embeddings $\mathbf{Z}$ and the reconstructed adjacency matrix $\hat{\mathbf{A}}$ as follows:

$$\hat{\mathbf{A}} = \sigma\left(\mathbf{Z}\mathbf{Z}^\top\right), \quad \text{with} \quad \mathbf{Z} = \mathrm{GCN}(\mathbf{X}, \mathbf{A}). \tag{4}$$

# Graph Convolutional Matrix Completion



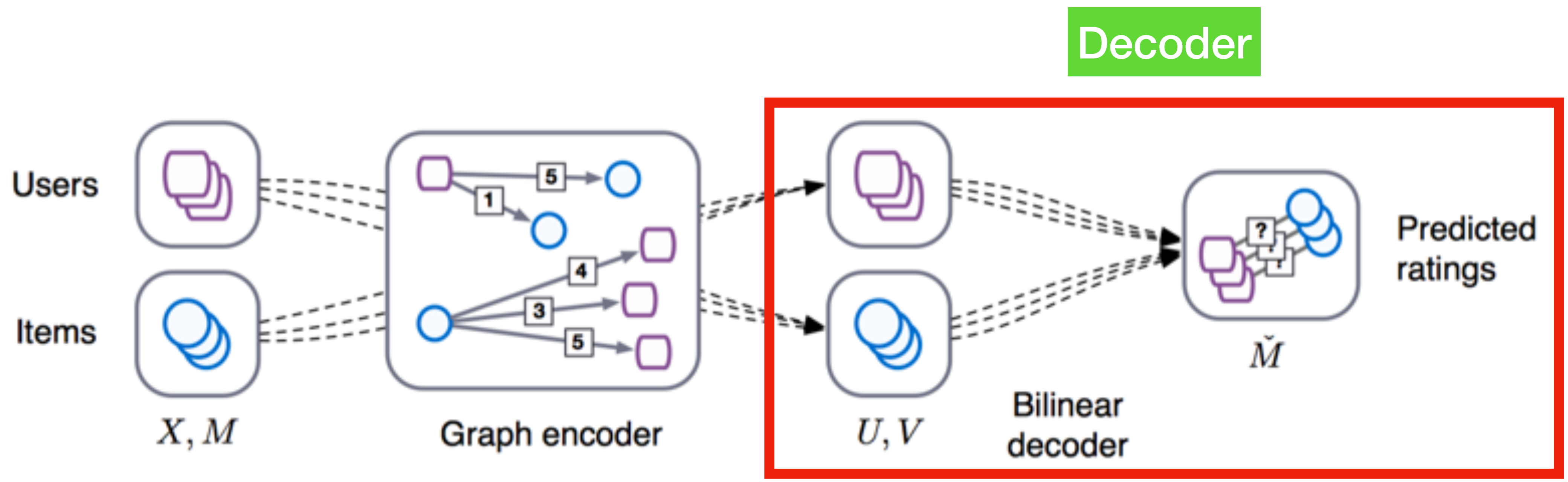

- Encoder
  - from items j to users i $\mu_{j \to i,r} = \frac{1}{c_{ij}} W_r x_j$

$c_{ij}$ is a normalization constant

$1/|\mathcal{N}_i|$ $\quad 1/\sqrt{|\mathcal{N}_i||\mathcal{N}_j|}$

$$h_i = \sigma\left[\text{accum}\left(\sum_{j \in \mathcal{N}_{i,1}} \mu_{j \to i,1}, \ldots, \sum_{j \in \mathcal{N}_{i,R}} \mu_{j \to i,R}\right)\right]$$

# Graph Convolutional Matrix Completion




- Decoder
  - probability distribution over possible rating levels

$$\check{M}_{ij} = g(u_i, v_j) = \mathbb{E}_{p(\check{M}_{ij}=r)}[r] = \sum_{r \in R} r \; p(\check{M}_{ij} = r)$$

  - The predicted rating is computed as $p(\check{M}_{ij} = r) = \frac{e^{u_i^T Q_r v_j}}{\sum_{s \in R} e^{u_i^T V_s v_j}}$