Event detection and forecasting using KPGNNs on temporally dynamic graphs

imporved with KL-divergence loss and Graph-ODE encoder

Ding, Yanna Gao, Suyi (Stanley) Saha, Bishwajit

Rensselaer Polytechnic Institute

Knowledge-Preserving Heterogeneous Graph Neural Network (KPGNN)

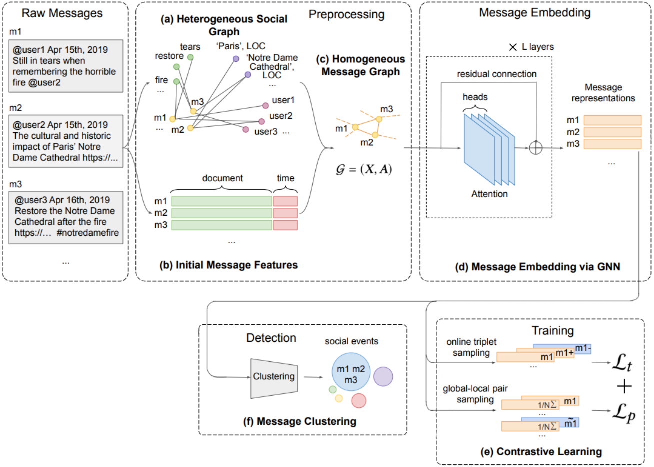

Figure 1: Workflow of KPGNN

KPGNN

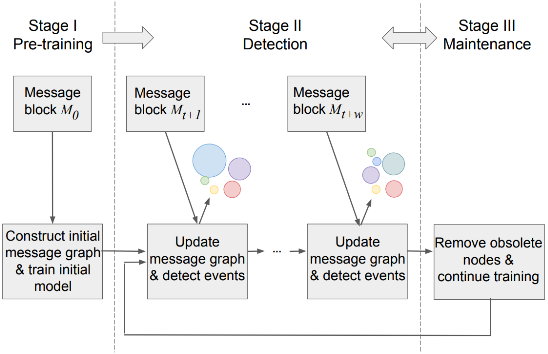

Figure 2: Training Stages of KPGNN

KPGNN vanilla

Algorithm 1: Model training in mini-batches (Stage 5)

Input: Batch size β, number of batches B, number of layers L

function train($\mathcal{G}, B, \beta, L$)
- **for** $b = 1, 2, \cdots, B$ **do**
  - $M^{(b)} = \{m_i\}_{i=1}^{\beta} \leftarrow$ sample messages from $\mathcal{G}$
  - **for** $l = 1, 2, \cdots, L$ **do**
    - $\mathbf{h}_{m_i}^{(l)} \leftarrow \Big\|_{\text{heads}} \left(\mathbf{h}_{m_i}^{(l-1)} \oplus \underset{m_j \in \mathcal{N}(m_i)}{Aggregator} \left(Extractor \left(\mathbf{h}_{m_j}^{(l-1)} \right) \right) \right)$
  - $\mathbf{h}_{m_i} \leftarrow \mathbf{h}_{m_i}^{L}$ among $M^{(b)}$
  - $T = \left\{ (m_i, m_i^+, m_i^-) \right\}_{i=1}^{\beta} \leftarrow$ triplet sampling from $M^{(b)}$
  - $\mathcal{L}_t \leftarrow$ triplet loss
  - $\mathbf{s}, \tilde{\mathbf{h}}_{m_i} \leftarrow$ calculate summary and corrupted representation
  - $\mathcal{L}_p \leftarrow$ cross entropy loss
  - Back-propagation to update parameters

KPGNN improved with GODE and KL divergence loss

Algorithm 2: Model training in mini-batches (Stage 5)

Input: Batch size β, number of batches B, number of layers L

function train($\mathcal{G}, B, \beta, L$)
- **for** $b = 1, 2, \cdots, B$ **do**
  - $M^{(b)} = \{m_i\}_{i=1}^{\beta} \leftarrow$ sample messages from $\mathcal{G}$
  - **for** $l = 1, 2, \cdots, L$ **do**
    - $\mathbf{h}_{m_i}^{(l)} \leftarrow$ (new) GODE encoder
  - $\mathbf{h}_{m_i} \leftarrow \mathbf{h}_{m_i}^{L}$ among $M^{(b)}$
  - $T = \left\{(m_i, m_i^{+}, m_i^{-})\right\}_{i=1}^{\beta} \leftarrow$ triplet sampling from $M^{(b)}$
  - $\mathcal{L}_t \leftarrow$ triplet loss
  - $\mathbf{s}, \tilde{\mathbf{h}}_{m_i} \leftarrow$ calculate summary and corrupted representation
  - $\mathcal{L}_p \leftarrow$ cross entropy loss
  - $\mathcal{L}_{kl} \leftarrow$ KL divergence loss
  - Back-propagation to update parameters

KL-divergence Loss

KL-divergence loss, also known as Kullback-Leibler divergence loss is used to measure the distance between two distributions. By including this loss in the model, we can obtain better clustering results.

$$\mathcal{L}_{kl} = \mathrm{KL}(P \| Q) = \sum_{ij} p_{ij} \log \left(\frac{p_{ij}}{q_{ij}} \right)$$

Predicted soft assignment:

$$q_{ij} = \frac{\left(1 + \frac{1}{\alpha} \left\| z_i - \mu_j \right\|^2\right)^{-\frac{1}{2}(\alpha+1)}}{\sum_{j'} \left(1 + \frac{1}{\alpha} \left\| z_i - \mu_{j'} \right\|^2\right)^{-\frac{1}{2}(\alpha+1)}}$$

Target distribution:

$$p_{ij} = \frac{q_{ij}^2 \Big/ \sum_i q_{ij}}{\sum_{j'} \left(q_{ij'}^2 \Big/ \sum_i q_{ij'} \right)}$$

KL-divergence Loss

- For p_i, a naive approach would be setting it to a delta distribution (to the nearest centroid) for data points above a confidence threshold and ignoring the rest.
- However, because q_i is soft assignment, it is more flexible to use softer probabilistic targets.
- Here p_i is defined by q_i and the intent of optimization is to match Q with P, thus we can regard it as a kind of self-training.
- Strengthen the predictions by putting more importance on data points assigned with high confidence that eventually improves cluster purity.
- Normalize loss contribution of each centroid to prevent large clusters from distorting the hidden feature space.

Graph ODE Encoder

We replace the original encoder with a continuous version of layered graph neural networks and refer it to the Graph Ordinary Differential Equation (GODE) model. The embedding $\mathbf{h}_{m_i}$ for node i is obtained by solving an initial value problem (IVP) for a coupled ODE system.

$$\mathbf{h}_{m_i}^{(t_1)} = \mathbf{h}_{m_i}^{(t_0)} + \int_{t_0}^{t_1} \underset{m_j \in \mathcal{N}(m_i)}{Aggregator}(Extractor(\mathbf{h}_{m_i}(t)))dt$$

where $\mathbf{h}_{m_i}^{(t_0)}$ is the initial condition of the hidden vector.
The corresponding discrete transformation is

$$\mathbf{h}_{m_i}^{(t)} = \mathbf{h}_{m_i}^{(t-1)} + \underset{m_j \in \mathcal{N}(m_i)}{Aggregator}\left(Extractor(\mathbf{h}_{m_j}^{(t-1)})\right)$$

Graph ODE Encoder: Expressivity

We use an example from the Neural Ordinary Differential Equations model to show the expressivity of ODE-based neural networks.

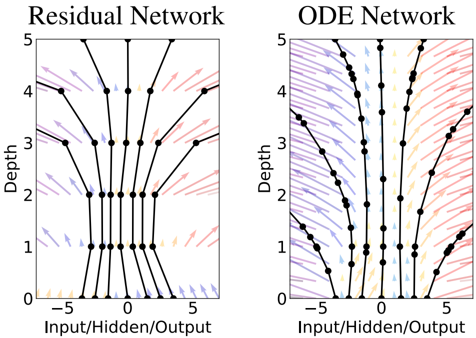

Figure 3: Neural ordinary differential equations, by Chen et al. The residual network has an update rule of $h_{t+1} = h_t + f(h_t)$, and the ODE network defines the final embedding as the solution to an IVP $h = h^{(0)} + \int_{t_0}^{t_1} f(h(t), t)dt$.

Graph ODE Encoder

We use the following instance of the above model. The expression for the hidden state of message m_i can be written as:

$$\mathbf{h}_{m_i}^{(T)} = \mathbf{h}_{m_i}^{(0)} + \frac{1}{\sqrt{k_i}} \int_0^T \sigma \left[\sum_{m_v \in \mathcal{N}(m_i)} \frac{1}{\sqrt{k_v}} (W \mathbf{h}_{m_v}(t) + b) \right] \mathrm{d}t \,, \quad \mathbf{h}_{m_i}^{(0)} = f_{\mathsf{enc}}(\mathbf{x}_{m_i})$$

The ODE is numerically solved using Runge-kutta4 (Dormand & Prince (1980)).

Experiment Setup

We conduct experiments on a subset of the Twitter dataset. Message block 0 includes 500 messages. Blocks 1-21 contains 100 messages and the corresponding heterogeneous knowledge representation graph consist of 100 nodes.

We adopt the *latest message strategy* to train the model. In the initial stage, the model is trained on message block M_0. For blocks $i = 1, \ldots, 21$, we first perform inference to get the test NMI for block i and if $i = 0$ mod *window size*, we train T iterations on block i.

NMI is a metric commonly used to calculate the clustering quality between the predicted labels and ground truth labels. A value of 1 indicates perfect clustering while a value of 0 indicates completely wrong class labels. Higher is better.

Test Result: KL-divergence

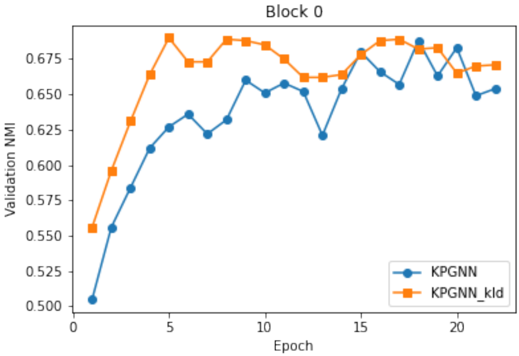

Figure 4: KPGNN vs KPGNN with kld loss at training stage.

Test Result: KL-divergence

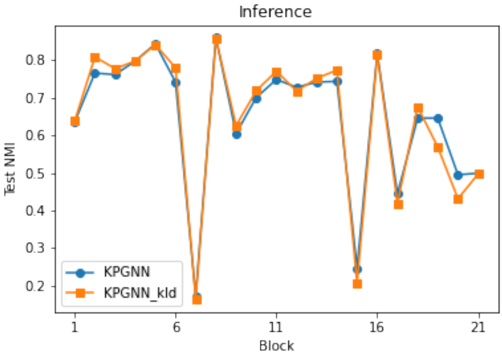

Figure 5: KPGNN vs KPGNN with kld loss at inference stage.

Test Result: GODE

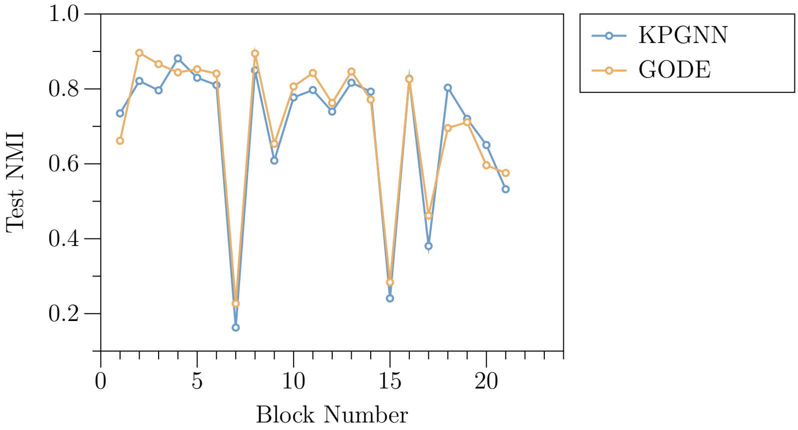

Figure 6: Test NMI for KPGNN and GODE.

Test Result: GODE

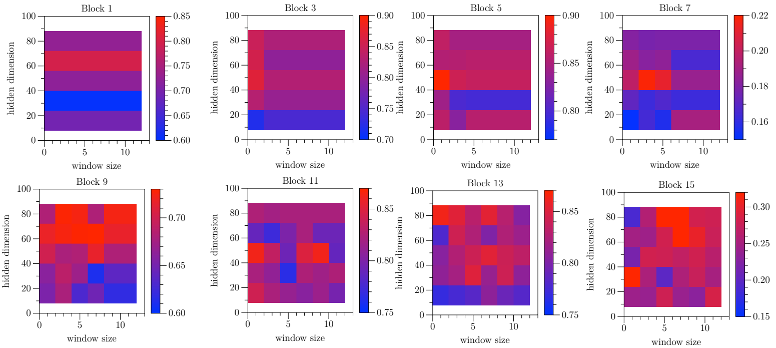

Figure 7: GODE with different hyperparameters. We fix the batch size at 150 and vary window size from $\{1, 3, 5, 7, 9\}$ and the embedding dimension from $\{16, 32, 48, 64, 80\}$.

Test Result: KL-divergence + GODE

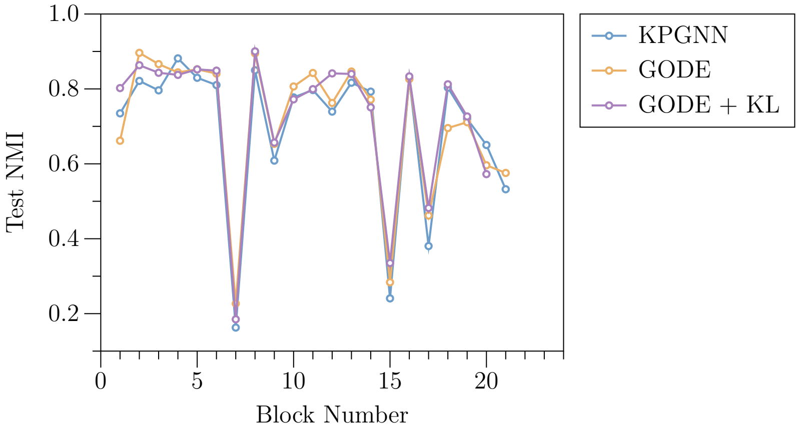

Figure 8: Test NMI for KPGNN, GODE, and the combination of GODE and KL-divergence.

References

Thank you for listening!

Yuwei Cao, Hao Peng, and Philip S Yu. Multi-information source hin for medical concept embedding. In Advances in Knowledge Discovery and Data Mining: 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11–14, 2020, Proceedings, Part II 24, pp. 396–408. Springer, 2020.

Yuwei Cao, Hao Peng, Jia Wu, Yingtong Dou, Jianxin Li, and Philip S Yu. Knowledge-preserving incremental social event detection via heterogeneous gnns. In Proceedings of the Web Conference 2021, pp. 3383–3395, 2021.

John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. Journal of computational and applied mathematics, 6(1):19–26, 1980.

Andrew J McMinn, Yashar Moshfeghi, and Joemon M Jose. Building a large-scale corpus for evaluating event detection on twitter. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pp. 409–418, 2013.

Hao Peng, Jianxin Li, Qiran Gong, Yangqiu Song, Yuanxing Ning, Kunfeng Lai, and Philip S Yu. Fine-grained event categorization with heterogeneous graph convolutional networks. arXiv preprint arXiv:1906.04580, 2019.

Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. Graph neural ordinary differential equations. arXiv preprint arXiv:1911.07532, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In International conference on machine learning, pp. 478–487. PMLR, 2016.