

Generating Graphs with Specified Properties

Yassine OJ,
Advanced AI for text and graphs

January 16, 2025

Overview

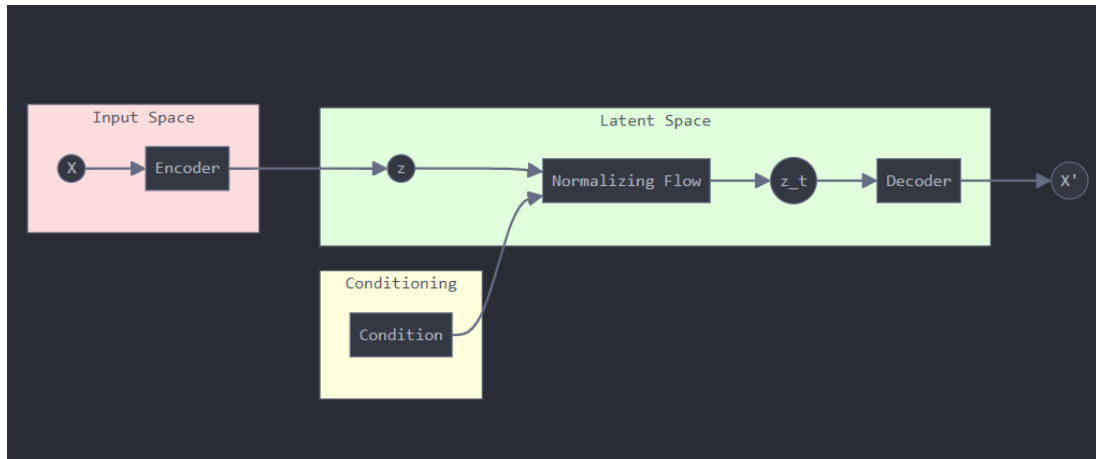
1. Presentation of the project
2. Normalizing flow
3. LLMs
4. LLMs
5. Second Section

Presentation of the project

Given 7 properties of a graph written in a text, generate a graph that respects those properties.

Example : "This graph comprises 49 nodes and 686 edges. The average degree is equal to 28.0 and there are 3647 triangles in the graph. The global clustering coefficient and the graph's maximum k-core are 0.5830535571542765 and 23 respectively. The graph consists of 5 communities."

Architecture of the model



Definition

Normalizing flows are a class of generative models that define a complex probability distribution $p_X(x)$ by transforming a simple base distribution $p_Z(z)$ (e.g., Gaussian) through a series of invertible and differentiable transformations $f = f_1 \circ f_2 \circ \dots \circ f_K$. Using the change of variables formula, the resulting density is computed as:

$$p_X(x) = p_Z(z) \left| \det \left(\frac{\partial z}{\partial x} \right) \right|,$$

where $z = f^{-1}(x)$, and $\det \left(\frac{\partial z}{\partial x} \right)$ is the determinant of the Jacobian matrix of the transformation f .

Conditional normalizing flow

Definition

Conditional Normalizing Flows are an extension to normalizing flows to model conditional probability distributions $p_X(x | c)$, where c represents additional conditioning information. The transformation f becomes condition-dependent, $f_c = f_{c,1} \circ f_{c,2} \circ \dots \circ f_{c,K}$, and the base distribution $p_Z(z)$ may also depend on c . Using the change of variables formula, the conditional density is expressed as:

$$p_X(x | c) = p_Z(z | c) \left| \det \left(\frac{\partial z}{\partial x} \right) \right|,$$

where $z = f_c^{-1}(x)$, and $\det \left(\frac{\partial z}{\partial x} \right)$ is the determinant of the Jacobian matrix of the conditional transformation f_c .

Definition [C.Winkler and al. (2019)]

Affine coupling layers are a type of coupling layer used in normalizing flows, which transforms data in a way that guarantees the bijectiveness of the transformation while maintaining computational efficiency. In an affine coupling layer, the transformation $f(x)$ is split into two parts:

$$f(x) = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ \exp(s(x_1)) \end{pmatrix} + \begin{pmatrix} 0 \\ t(x_1) \end{pmatrix}$$

Definition

Affine coupling layers are one family of those functions. They are a type of coupling layers used in normalizing flows, which transforms data in a way that guarantees the bijectiveness of the transformation while maintaining computational efficiency. In an affine coupling layer, the transformation $f(x)$ is split into two parts:

$$f(x) = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ \exp(s(x_1)) \end{pmatrix} + \begin{pmatrix} 0 \\ t(x_1) \end{pmatrix}$$

Affine coupling layers

The inverse of the affine coupling layer is computed as:

$$\mathbf{x}_1 = \mathbf{y}_1,$$

$$\mathbf{x}_2 = (\mathbf{y}_2 - t(\mathbf{y}_1)) \odot \exp(-s(\mathbf{y}_1)).$$

The Jacobian of the affine coupling layer is a triangular matrix due to the structure of the transformation. Therefore, the determinant of the Jacobian is:

$$\det \left(\frac{\partial(\mathbf{y}_1, \mathbf{y}_2)}{\partial(\mathbf{x}_1, \mathbf{x}_2)} \right) = \prod_i \exp(s_i(\mathbf{x}_1)),$$

where $s_i(\mathbf{x}_1)$ are the elements of the scale function $s(\mathbf{x}_1)$.

The log-determinant of the Jacobian is:

$$\log \det \left(\frac{\partial(\mathbf{y}_1, \mathbf{y}_2)}{\partial(\mathbf{x}_1, \mathbf{x}_2)} \right) = \sum_i s_i(\mathbf{x}_1).$$

Conditional Prior: $p(z|x) = \mathcal{N}(z; \mu(x), \sigma^2(x))$

Conditional Coupling: $y_0 = s(z_1, x) \cdot z_0 + t(z_1, x); \quad y_1 = z_1$

In practice :

$$\mu(x), \sigma^2(x) = NN(condition)$$

Stabilizing the normalizing flow

Idea [Daniel Andrade (2024)] : Threshold high and low scaling factors to ensure that $\alpha_{\text{neg}} \leq s(z_1, x) \leq \alpha_{\text{pos}}$, where the thresholds α_{pos} and α_{neg} is set to a fixed small value.

$$c(s) = \begin{cases} \frac{2}{\pi} \alpha_{\text{pos}} \arctan \left(\frac{s}{\alpha_{\text{pos}}} \right), & \text{if } s \geq 0, \\ \frac{2}{\pi} \alpha_{\text{neg}} \arctan \left(\frac{s}{\alpha_{\text{neg}}} \right), & \text{if } s < 0, \end{cases}$$

Definition [M. Belghazi and al. (2018)]

Mutual Information Neural Estimator (MINE) is a method for estimating the mutual information $I(X; Y)$ between two random variables X and Y . MINE relies on the Donsker-Varadhan representation of the Kullback-Leibler divergence:

$$I(X; Y) = \sup_{T_\theta} \mathbb{E}_{p(X, Y)}[T_\theta(x, y)] - \log \mathbb{E}_{p(X)p(Y)} \left[e^{T_\theta(x, y)} \right],$$

where T_θ is a neural network parameterized by θ , which acts as a critic function to distinguish between joint samples $(x, y) \sim p(X, Y)$ and marginal samples $(x, y) \sim p(X)p(Y)$.

$$total_loss = normalizing_loss + \beta \times (-I(X; Y))$$

- The model's performance was suboptimal.
- Base case result on the test dataset: **0.98**.
- Likely explanation:
 - Latent embeddings produced by the normalizing flow were not sufficiently representative of the condition.
 - This issue persisted despite employing various techniques to improve the embedding quality.

- **T5 (Text-to-Text Transfer Transformer)**

- Powerful at generating meaningful and coherent textual embeddings.
- Captures contextual information from input text.
- Effective for diverse text-based tasks, including graph link prediction.

- **BERT (Bidirectional Encoder Representations from Transformers))**

- Captures nuanced contextual information from text.
- Considers both the left and right context of a word,
- Excels in understanding sentence-level semantics, which aids in generating detailed node representations in graph models.

- **Variational Graph Autoencoder (VGAE) with standard F1 loss.**
- Baseline MAE: 0.8
- Improved Performance
 - T5 embeddings: $\text{MAE} = 0.41$
 - BERT embeddings: $\text{MAE} = 0.50$

Modified Loss Function

- Original: Standard F1 loss.
- Modifications:
 - Replaced with L1 loss to minimize absolute errors.
 - Added L2 regularization for better generalization.
- Results:
 - T5 embeddings: MAE remains at 0.41 (optimized).
 - BERT embeddings: MAE improved to 0.50.

Modified Loss Function

- Combines GNNs and transformers for graph-based learning.
- Key Components:
 - Self-attention for global dependencies in the graph.
 - Global pooling for fixed-size graph representation.
 - Decoder to reconstruct graph structure.
- Predicts graph relationships effectively:

Using T5 BERT with Graph Transformer

- Integration: T5 and BERT embeddings as node features.
- Enhancements:
 - Enriched node representations with semantic information.
 - Captures both textual context and graph structure.
- Results: $MAE = 0.70$.

- T5 and BERT significantly enhance link prediction tasks.
- Modified loss function (L1 + L2 regularization) improves accuracy.
- Graph Transformers effectively combine semantic and structural data.

Table

| Treatments | Response 1 | Response 2 |
|-------------------|-------------------|-------------------|
| Treatment 1 | 0.0003262 | 0.562 |
| Treatment 2 | 0.0015681 | 0.910 |
| Treatment 3 | 0.0009271 | 0.296 |

Table: Table caption

Theorem

Theorem (Mass–energy equivalence)

$$E = mc^2$$




Figure

Uncomment the code on this slide to include your own image from the same directory as the template .TeX file.

An example of the `\cite` command to cite within the presentation:

This statement requires citation [?].

References

-  Christina Winkler, Daniel Worrall, Emiel Hoogetboom, Max Welling (2019). Learning Likelihoods with conditional normalizing flows, [arXiv:1912.00042](#)
-  Daniel Andrade (2024). Stabilizing training of affine coupling layers for high-dimensional variational inference. *Machine Learning: Science and Technology*, Volume 5, Number 4
-  Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, R Devon Hjelm (2018). MINE: Mutual Information Neural Estimation, [arXiv:1801.04062](#)
MINE: Mutual Information Neural Estimation.
arXiv:1801.04062, 2018.
-  Jenny Liu, Aviral Kumar, Jimmy Ba, Jamie Kiros, Kevin Swersky (2020). Graph Normalizing Flows.
arXiv preprint arXiv:2006.00951.

References



Christos.X and Iakovos Evdaimon.
Generating Graphs with Specified Properties.
Kaggle Competition, 2024.



Iakovos Evdaimon, Giannis Nikolentzos, Christos Xypolopoulos, Ahmed Kammoun, Michail Chatzianastasis, Hadi Abdine, and Michalis Vazirgiannis.
Neural Graph Generator: Feature-Conditioned Graph Generation using Latent Diffusion Models.
arXiv:2403.01535, 2024.







Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. A., Kaiser, Ł., Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.



Kipf, T. N., Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

References

-  Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y. (2018). Graph attention networks. *arXiv preprint arXiv:1710.10903*.
-  Raffel, C., Shinn, C., Roberts, A., Lee, S., Narang, S., Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140), 1-67.
-  Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT 2019*.
-  Kingma, D. P., Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

The End