Modele generatywne dla grafów

Jakub Binkowski
jakub.binkowski@pwr.edu.pl

Szkoła Letnia AITech 2023

Katedra Sztucznej Inteligencji

Politechnika Wrocławska

Cele warsztatów

1. Wprowadzenie do zagadnienia generowania grafów
2. Wprowadzenie do Grafowych Sieci Neuronowych
3. **Analiza działania w teorii i praktyce** wybranych metod, zadania praktyczne z zakresu 3 modeli:
    a. VGAE
    b. GraphVAE
    c. DGMG
4. Omówienie teoretyczne wybranych modeli spośród:
        a. GAN
        b. Normalizing Flows
        c. Diffusion Models

Część praktyczna warsztatów

https://tinyurl.com/graphgen-aitech

Wprowadzenie

Do czego potrzebne nam generowanie grafów?

- Przetwarzanie języka naturalnego
- Generowanie kodu oprogramowania
- **W chemii / biologii (AI4Science):**
  - generowanie nowych molekuł (o zadanych parametrach), generowanie potencjalnych kandydatów na leki!
  - generowanie protein
- Otrzymanie bogatszej reprezentacji dla zadań docelowych (*representation learning / pre-training*)

Podejścia klasyczne

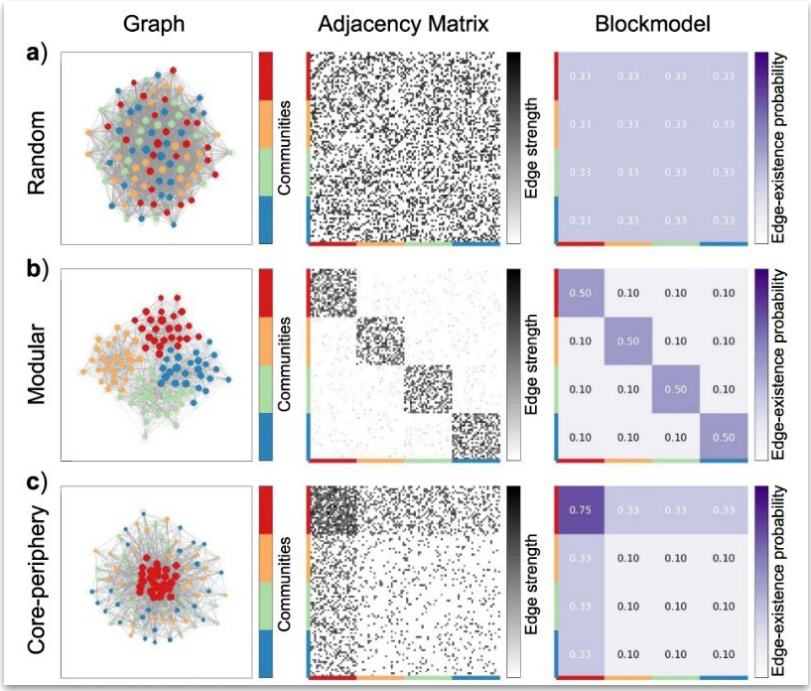
Zakładają pewne statystyczne własności grafów, na podstawie których definiują metodę ich generowania. Przykładowe modele:

- Erdős–Rényi

$$P(\mathbf{A}[u, v] = 1) = r \quad \forall u, v \in \mathcal{V}$$

- Stochastic Block Models
  - różne prawdopodobieństwa dla krawędzi wewnątrz i na zewnątrz klastra
- Barabási–Albert Model (*preferential attachment*)
  - mała liczba wierzchołków o wysokim stopniu, duża liczba wierzchołków o małym stopniu (*power law distribution)*

Stochastic Block Model



Przykładowe reprezentacje SBM grafu [(Faskowitz et al., 2018)]

W czym tkwi problem z metodami klasycznymi?

- Z góry ustalony, ręcznie opracowany proces generowania grafów
- W rzeczywistości rozkład prawdopodobieństwa grafów jest bardzo skomplikowany
- Klasyczne metody stosują duże uproszczenie względem rzeczywistego, złożonego, rozkładu generującego
- Dlatego dziś uwaga skierowana jest na modelach głębokich…

W czym tkwi problem z metodami klasycznymi?

- Z góry ustalony, ręcznie opracowany proces generowania grafów
- W rzeczywistości rozkład prawdopodobieństwa grafów jest bardzo skomplikowany
- Klasyczne metody stosują duże uproszczenie względem rzeczywistego, złożonego, rozkładu generującego
- Dlatego dziś uwaga skierowana jest na modelach głębokich…
- **Ale jak użyć głębokich modeli do grafów?**

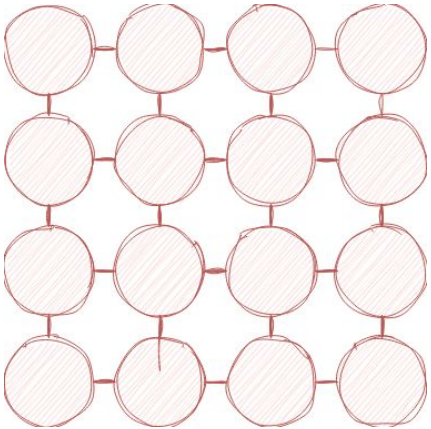
W czym tkwi problem z metodami klasycznymi?

- Z góry ustalony, ręcznie opracowany proces generowania grafów
- W rzeczywistości rozkład prawdopodobieństwa grafów jest bardzo skomplikowany
- Klasyczne metody stosują duże uproszczenie względem rzeczywistego, złożonego, rozkładu generującego
- Dlatego dziś uwaga skierowana jest na modelach głębokich…
- **Ale jak użyć głębokich modeli do grafów?**

Grafowe Sieci Neuronowe

Grafowe Sieci Neuronowe

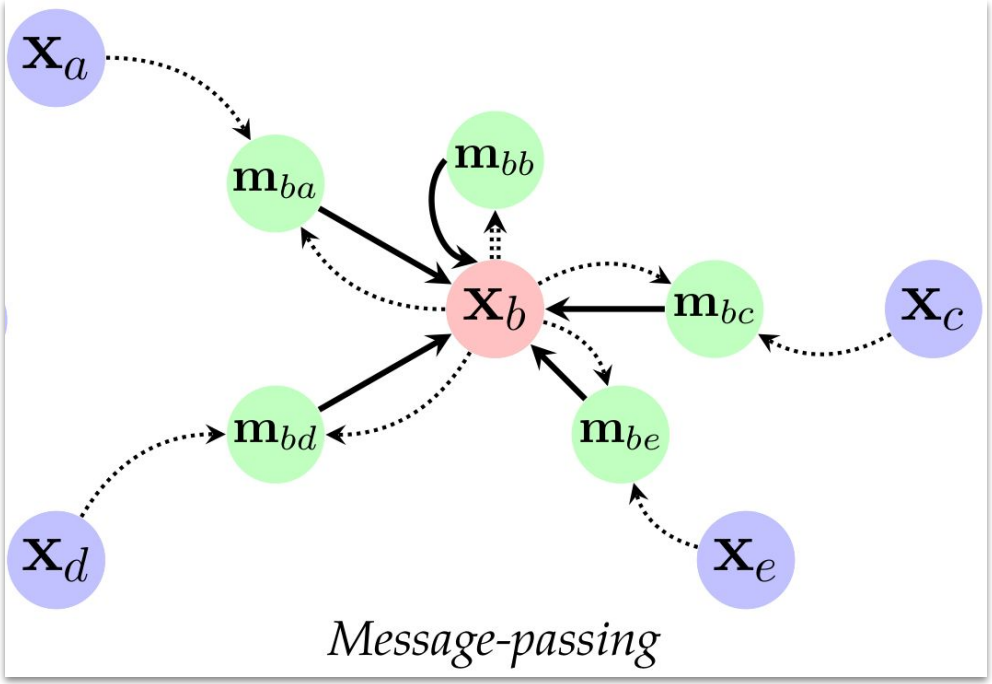
- Przetwarzanie obrazów oraz tekstu jest relatywnie prostsze niż grafów - **są to struktury regularne**



- grafy są nieregularne, struktura (dziedzina) zmienia się z każdym kolejnym grafem - jak zatem na nich operować?

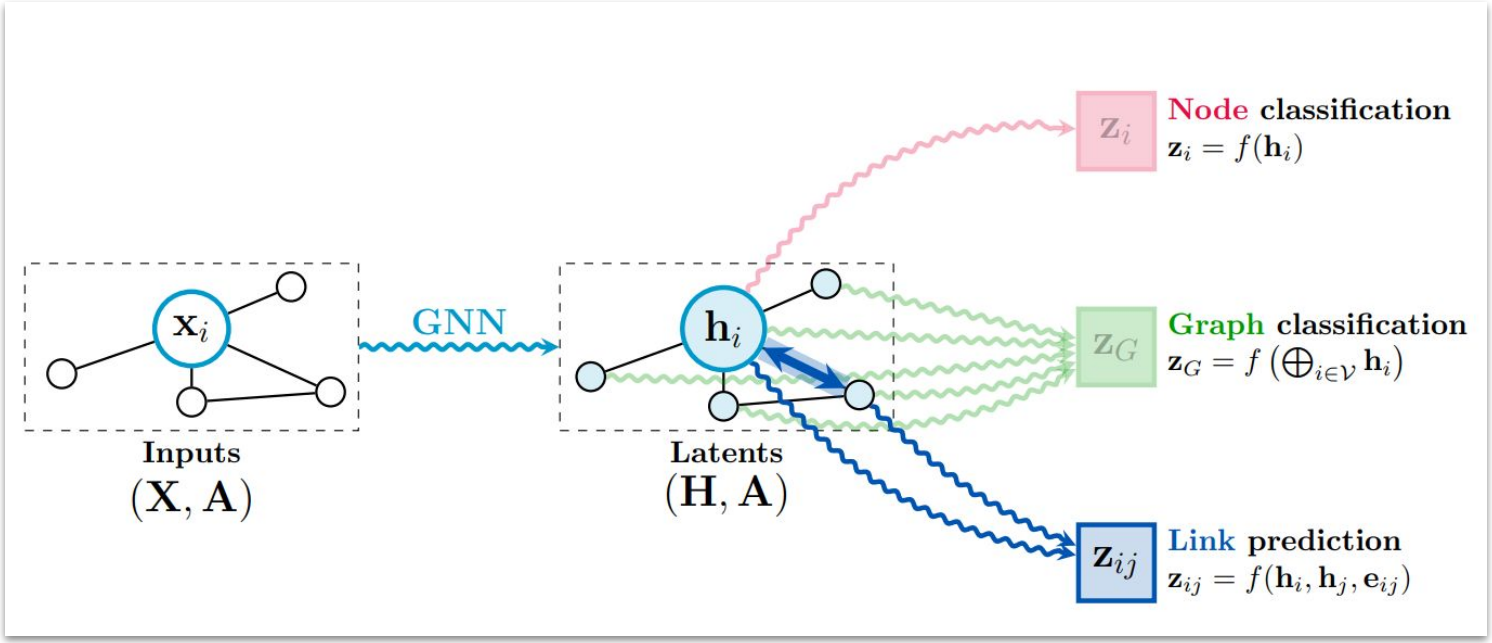
Grafowe Sieci Neuronowe (GNN)

$$\mathbf{h}_u = \mathrm{UPDATE}\left(\mathbf{x}_u, \mathrm{AGGREGATE}\left(\{\psi(\mathbf{x}_u, \mathbf{x}_v); \forall v \in \mathcal{N}(u)\}\right)\right)$$



(Bronstein et al., 2021)

Grafowe Sieci Neuronowe (GNN)



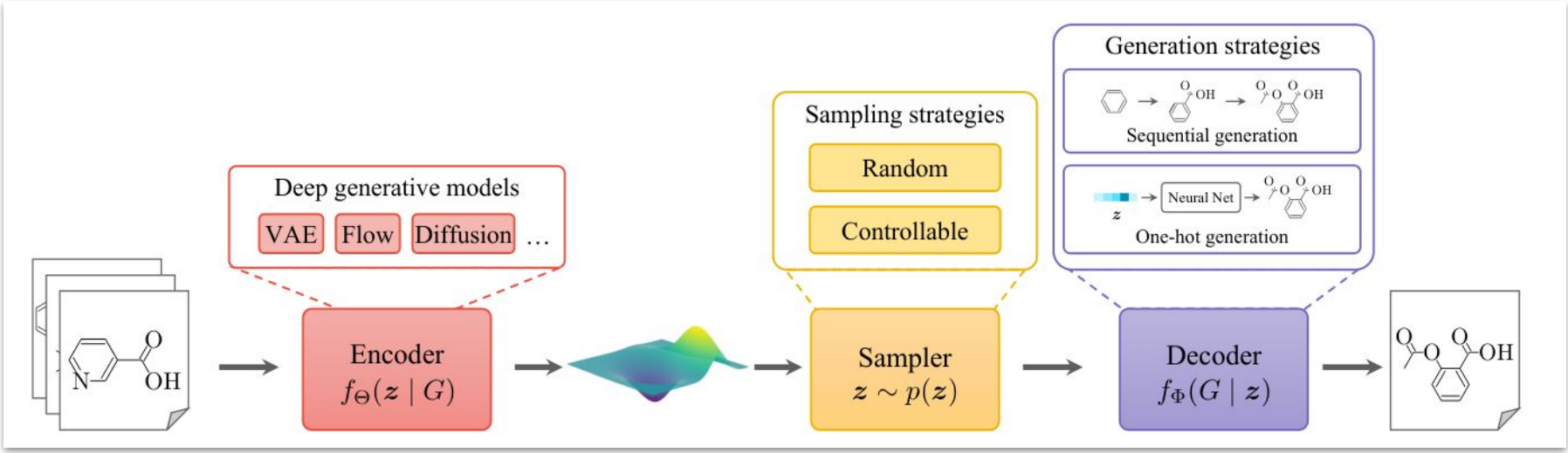
Wykorzystanie GNN w różnych zadaniach (Veličković, 2023)

Jakie są problemy? (Guo et al., 2020)

- Nieunikalne reprezentacje - graf o *n* wierzchołkach może być reprezentowany na *n!* sposobów
- Skomplikowane zależności węzłów i krawędzi - istnienie węzłów i krawędzi jest uzależnione od licznych powiązań w grafie
- Duże przestrzenie wyjściowe - najprostsze podejścia wymagają sprawdzenia wszystkich możliwych krawędzi (w rzeczywistości grafy są rzadkie)
- Dyskretność grafów - wiele metod działa w przestrzeniach ciągłych, co w przypadku grafów nie ma zastosowania
- Ewaluacja skuteczności - nie ma jednoznacznej metody oceny generowanych grafów

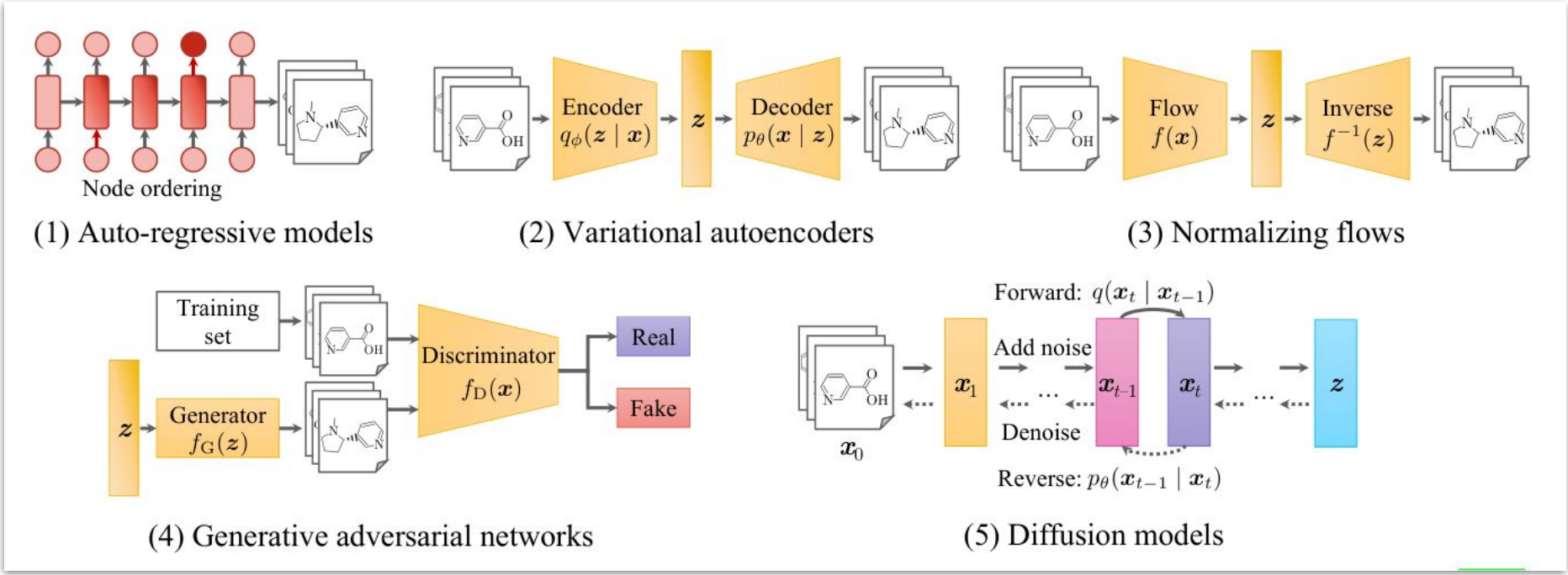
Modele generatywne dla grafów

Struktura modeli generatywnych dla grafów



(Zhu et al., 2022)

Rodzaje modeli

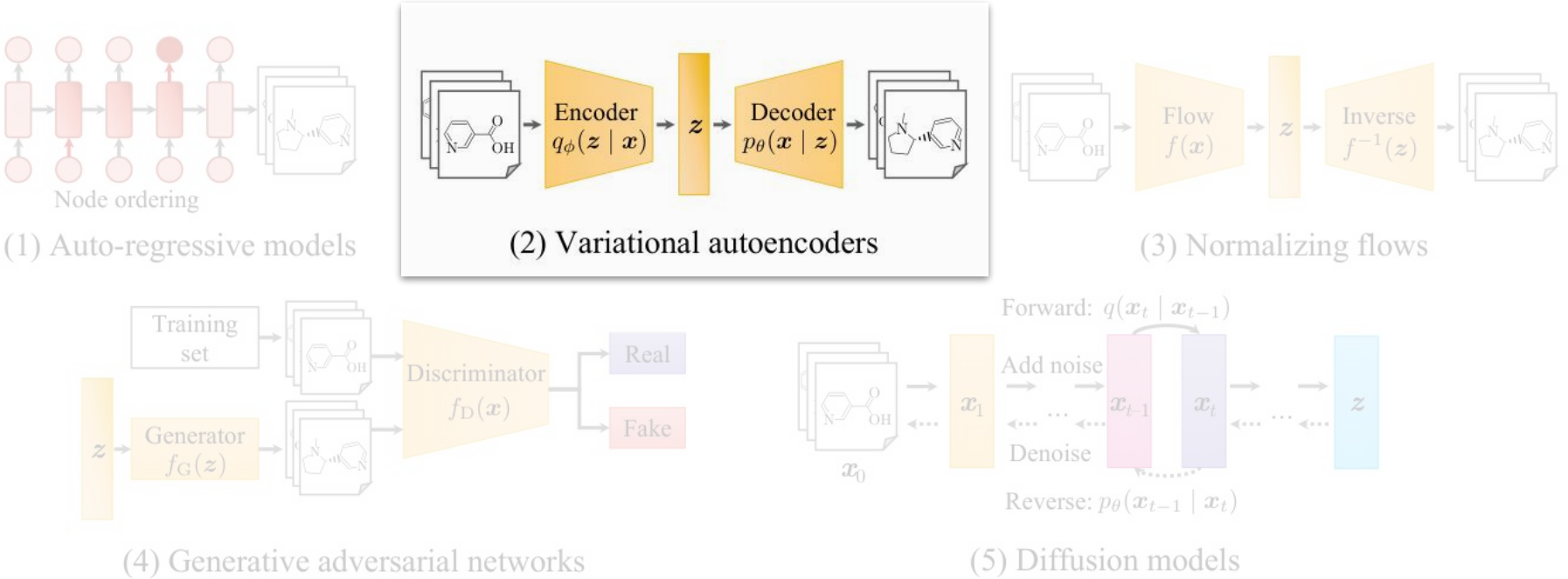


(Zhu et al., 2022)

VGAE

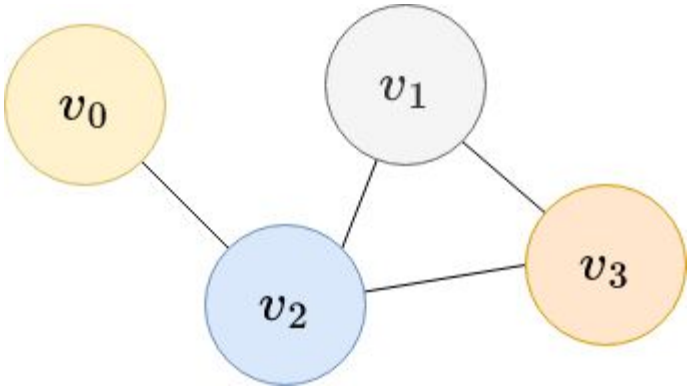
(Kipf & Welling, 2016)

Modele



(Zhu et al., 2022)

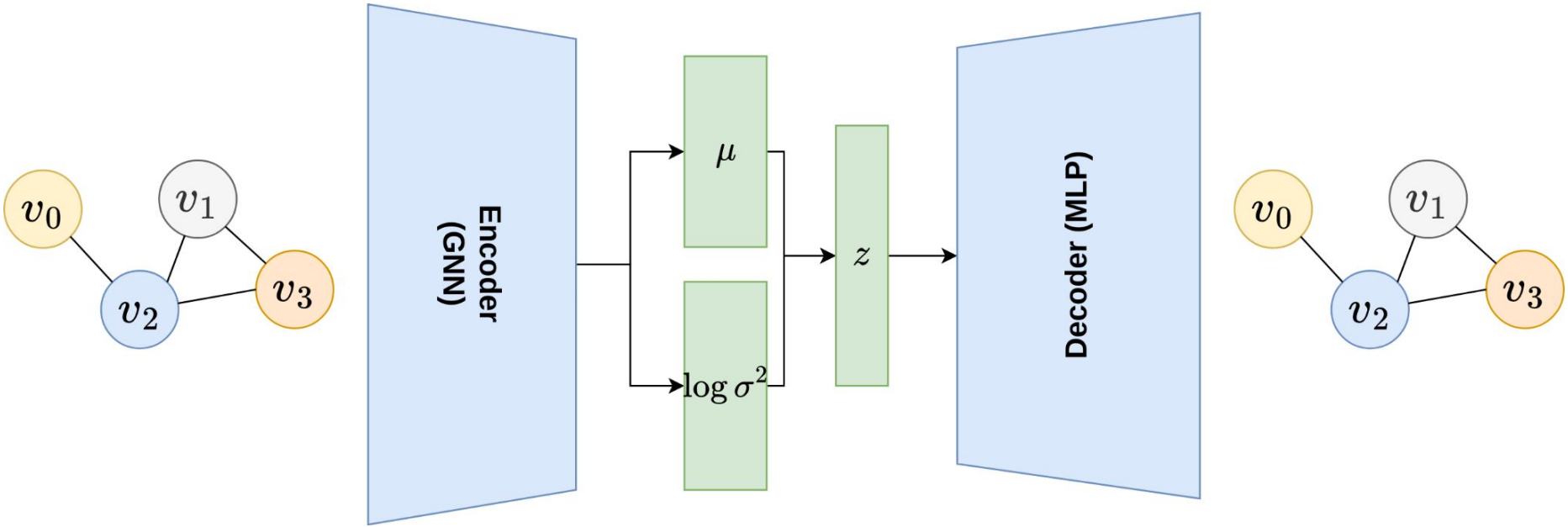
Jak przedstawić graf?



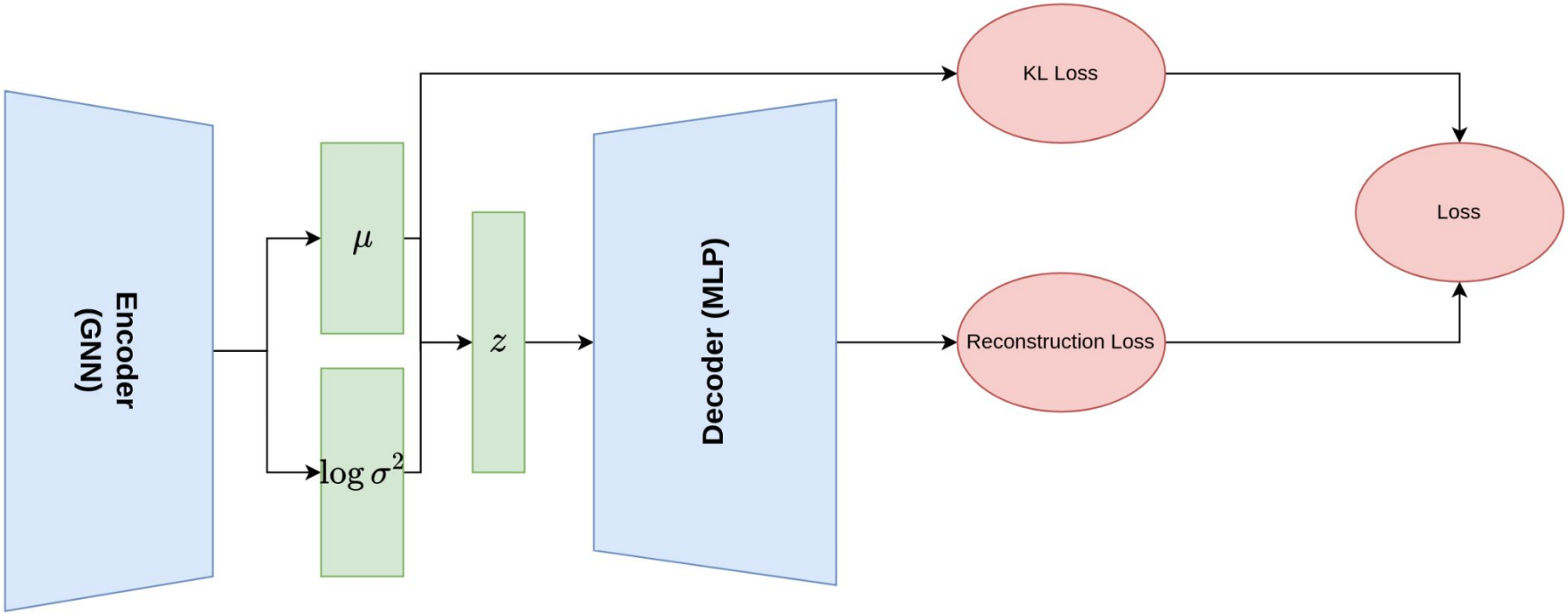
$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

$$X = \begin{bmatrix} 0.1 & 0.5 & 2.0 \\ 0.9 & 4.5 & 1.0 \\ 0.3 & -0.5 & 3.0 \\ 0.1 & 1.5 & 7.0 \end{bmatrix}$$

Variational Graph Autoencoder



Variational Graph Autoencoder



Variational Graph Autoencoder

1. **Koder** (*encoder)* GNN transformuje graf do postaci reprezentacji wierzchołków:
   a. wektor średnich dla każdego wierzchołka
   b. wektor odchyleń dla każdego wierzchołka
2. **Dekoder** MLP* przewiduje istnienie krawędzi na podstawie (wszystkich) par wierzchołków
3. Model uczymy jakby był klasyfikatorem - dekoder klasyfikuje istnienie krawędzi (zadanie *link prediction*)
4. Po zakończeniu uczenia możemy próbkować w przestrzeni ukrytej (z rozkładu normalnego)

* w oryginalnej pracy był to iloczyn skalarny

Variational Graph Autoencoder - funkcja straty

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

Błąd rekonstrukcji

Regularyzacja do *N(0, 1)*

Ale jak ocenić jakość modelu?

- Jakość rekonstrukcji jest stosunkowo prosta do oceny - możemy ocenić dekoder tak samo jak klasyfikator, np. miarą AUC
- Jednak nie jest oczywistym jak ewaluować wygenerowane grafy:

Type		*Evaluation feature*
General	Statistics-based	Average KLD
		MMD
	Classifier-based	Accuracy-based
		FID-based
	Intrinsic-quality-based	Validity
		Uniqueness
		Novelty
Condition-specialized	Graph property-based	
	Mapping-relationship-based	

(Guo et al., 2022)

Maximum Mean Discrepancy

- MMD pozwala nam porównać dwa rozkłady prawdopodobieństwa - jest to odległość pomiędzy średnimi z cech obiektów

$$MMD^2(P, Q) = \|\mu_P - \mu_Q\|_{\mathcal{F}}^2$$

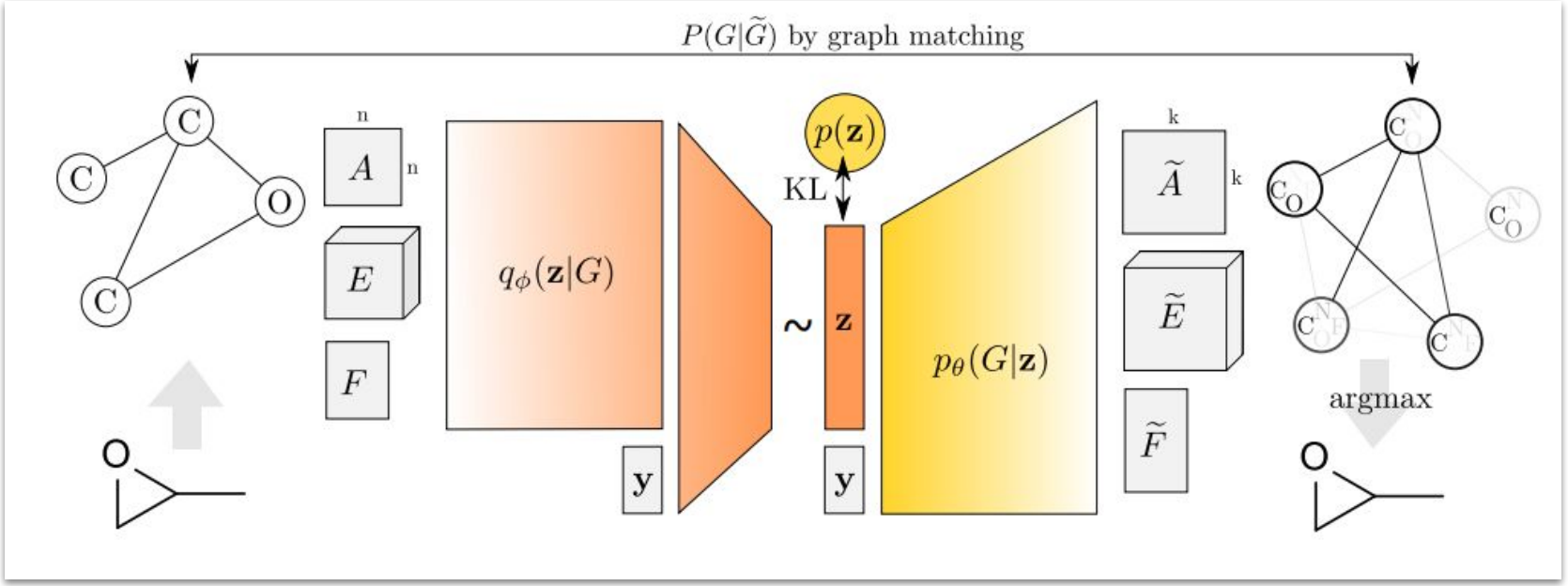
- Odległość liczymy pomiędzy cechami zbioru testowego a cechami wygenerowanych grafów
- W części praktycznej skorzystamy z następujących cech:
  - stopień wierzchołków
  - współczynnik klasteryzacji
  - cechy spektralne

Część praktyczna I

Variational Graph Autoencoder (VGAE)

- Model nie dostarcza nam zadowalających rezultatów, nawet na prostym zbiorze
- VGAE jest metodą, która opracowana była głównie do uczenia reprezentacji - dzięki podejściu *unsupervised* możemy otrzymać embeddingi wierzchołków do zadań docelowych (ang. *downstream tasks*)
- **Jednak VGAE jest punktem wyjścia dla późniejszych udoskonaleń, które osiągały wysoką jakość w generowaniu grafów, np. GraphVAE (Simonovsky & Komodakis, 2018), JT-VAE (Jin et al., 2019)**

GraphVAE



Architektura modelu [(Simonovsky & Komodakis, 2018)]

Część praktyczna II

GraphVAE

- Używanie zagregowanej reprezentacji całego grafu poprawiło jakość modelu generatywnego
- Wiąże się to jednak z dodatkowym narzutem obliczeniowym
- Ale VAE to nie jedyny paradygmat z jakiego możemy skorzystać…

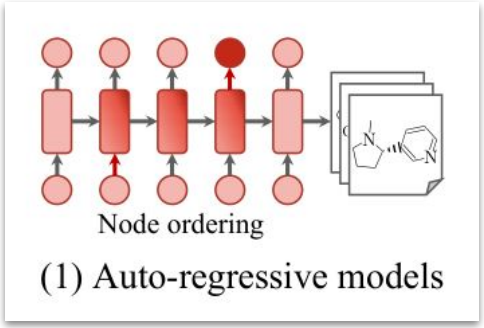
Wady generowania *one-shot*

- Należy założyć z góry maksymalną liczbę wierzchołków
- Trudności w generowaniu dużych (i rzadkich) grafów - duża wymiarowość wyjściowa modelu
- Często wymagane jest dopasowywanie grafów (*graph matching*) - dodatkowy narzut obliczeniowy!
- Założenie o niezależności krawędzi w grafie nie jest w rzeczywistości spełnione

Learning Deep Generative Models of Graphs (DGMG)

(Li et al., 2018)

Modele

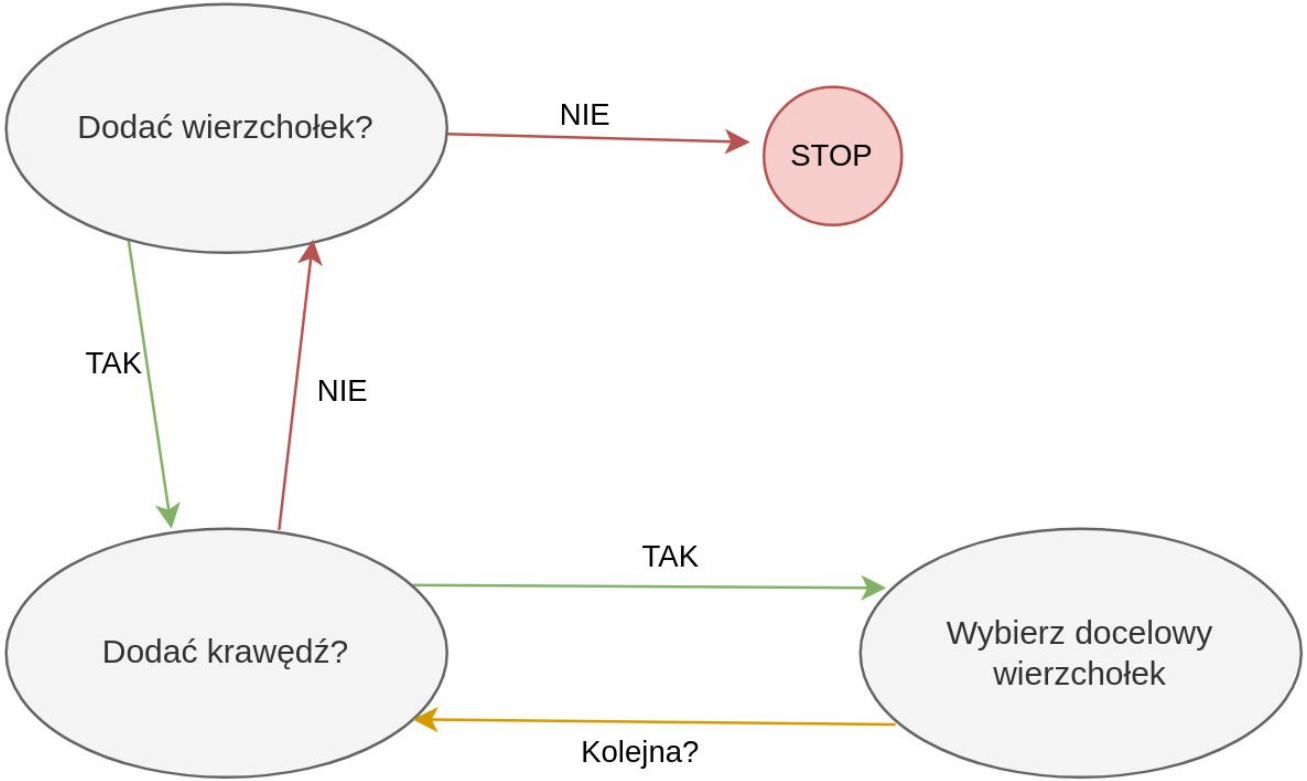


(Zhu et al., 2022)

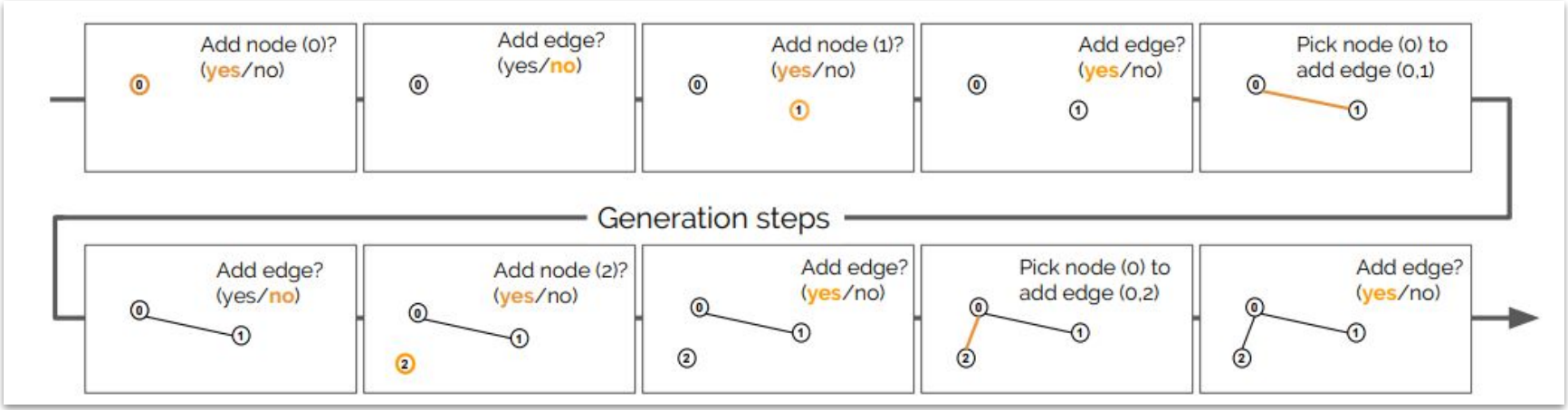
A co gdyby nie generować wszystkiego na raz?

- Generowanie grafu możemy rozbić na sekwencje kroków
- Modele sekwencyjne mają kilka pożądanych własności:
  - możemy generować sekwencje o dowolnej długości
  - nie zakładamy niezależności generowanych elementów sekwencji!
- Mamy dostęp do licznych pozwalających na generowanie sekwencji, np. RNN
- Zatem spróbujmy przekształcić graf w sekwencje

Proces generowania grafu (Li et al., 2018)



Proces generowania grafu - przykład



Proces generowania grafów [(Li et al., 2018)]

Algorytm

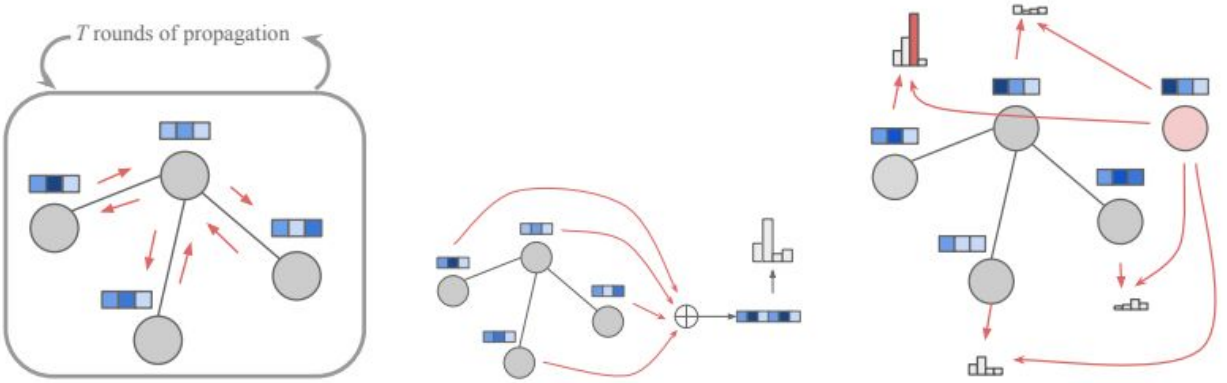


Figure 2. Illustration of the graph propagation process (left), graph level predictions using $f_{addnode}$ and $f_{addedge}$ (center), and node selection f_{nodes} modules (right).

Algorytm generowania grafu (Li et al., 2018)

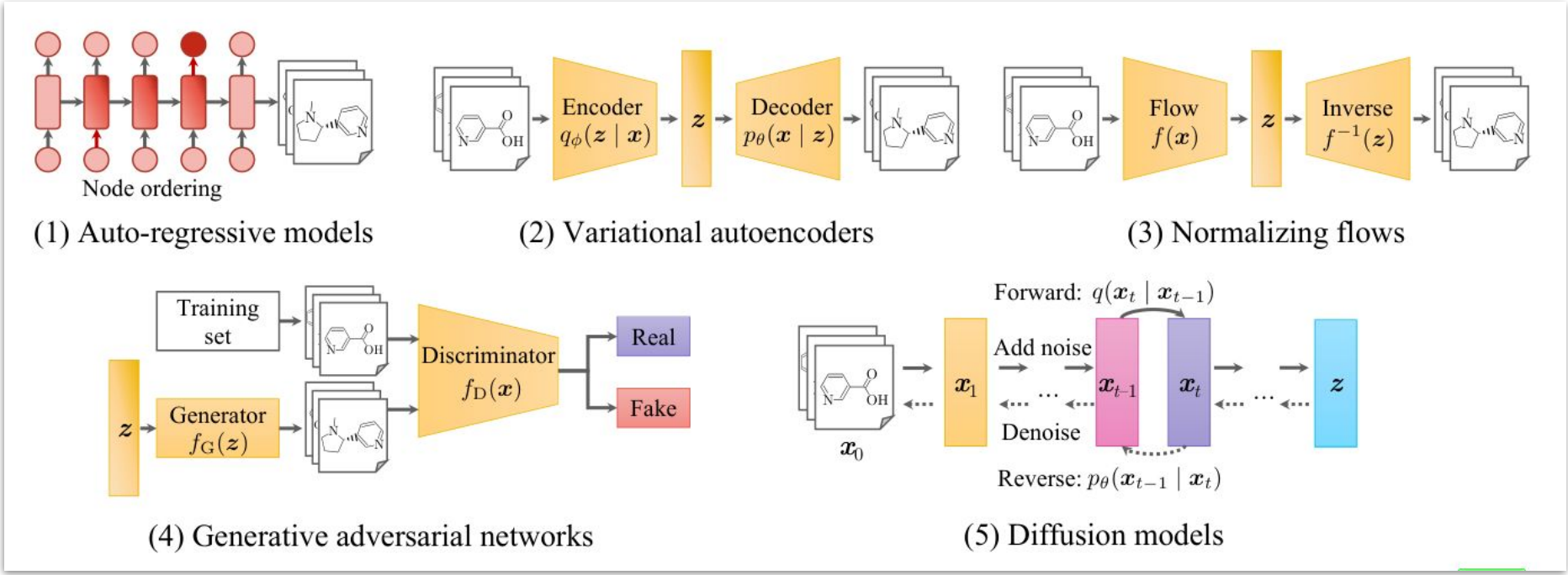
Część praktyczna III

DGMG

- Oprócz samej struktury grafu możemy przewidywać także cechy wierzchołków i krawędzi, oraz warunkować generowanie - wymaga to pewnej modyfikacji modelu
- Jednak DGMG (oraz inne modele autoregresyjne) mają kilka wad:
  - konieczność uszeregowania wierzchołków
  - trudności w uczeniu, szczególnie długich sekwencji
  - propagacja błędu
- Możemy również użyć innych podejść autoregresyjnych: (Liao et al., 2020), (You et al., 2018)

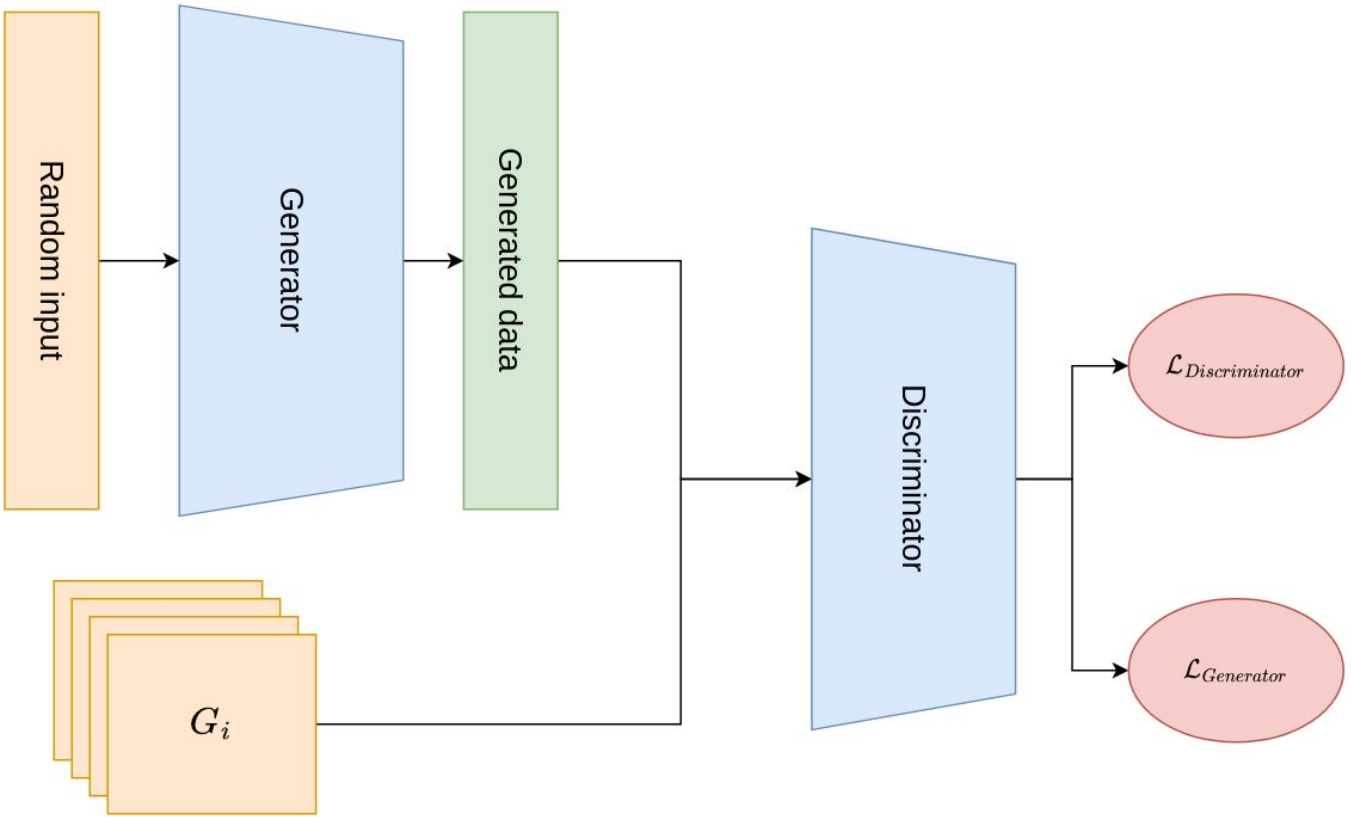
To nie koniec…

Modele

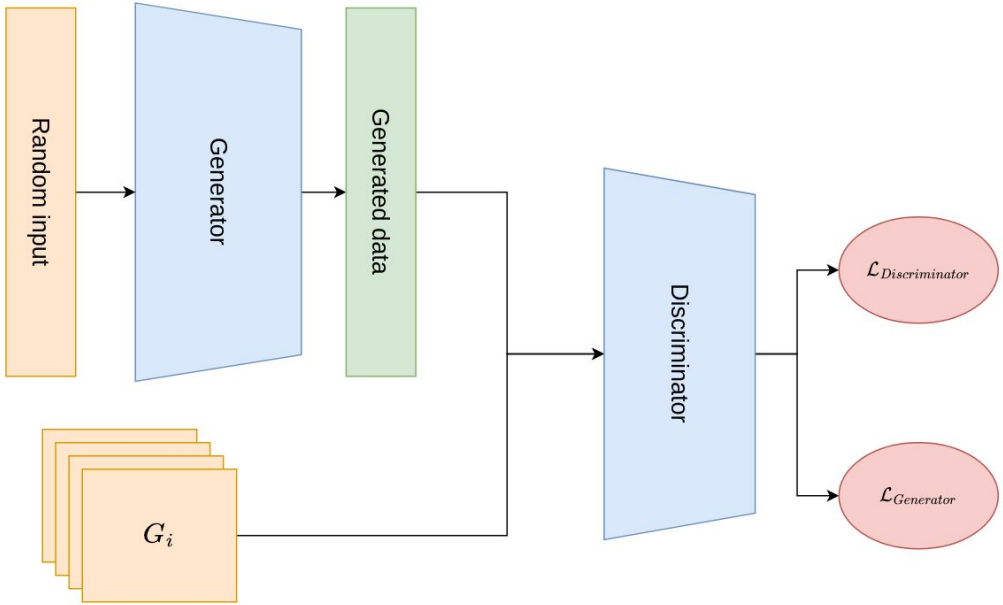


(Zhu et al., 2022)

Sieci typu GAN

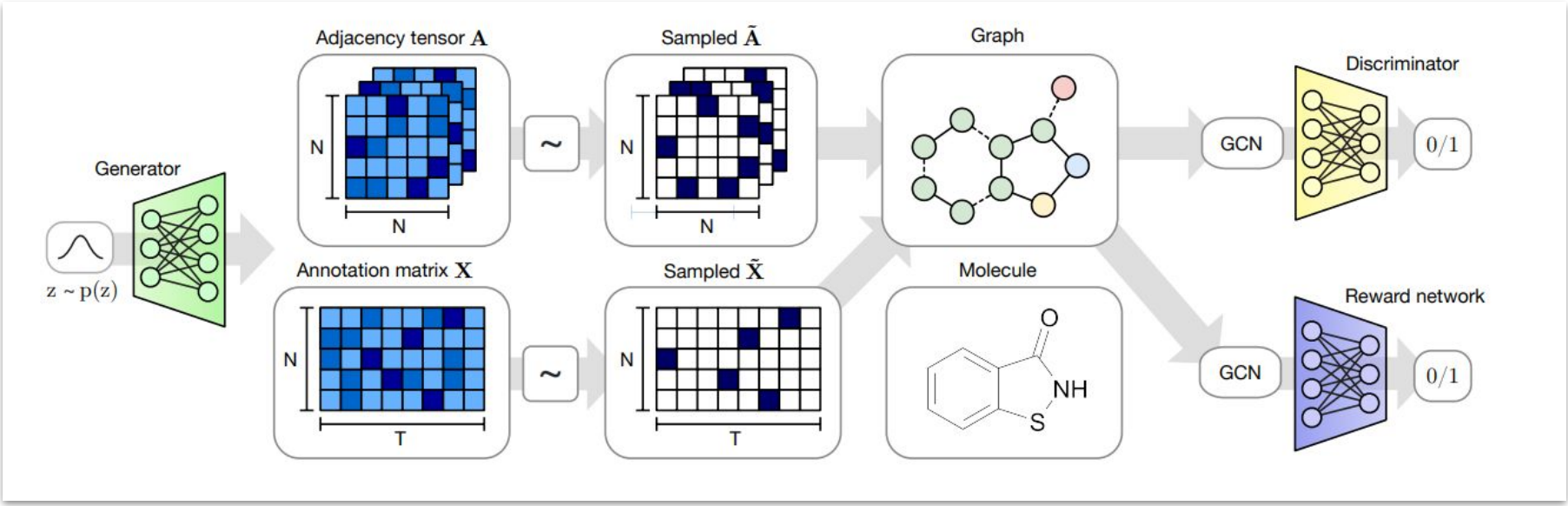


Sieci typu GAN



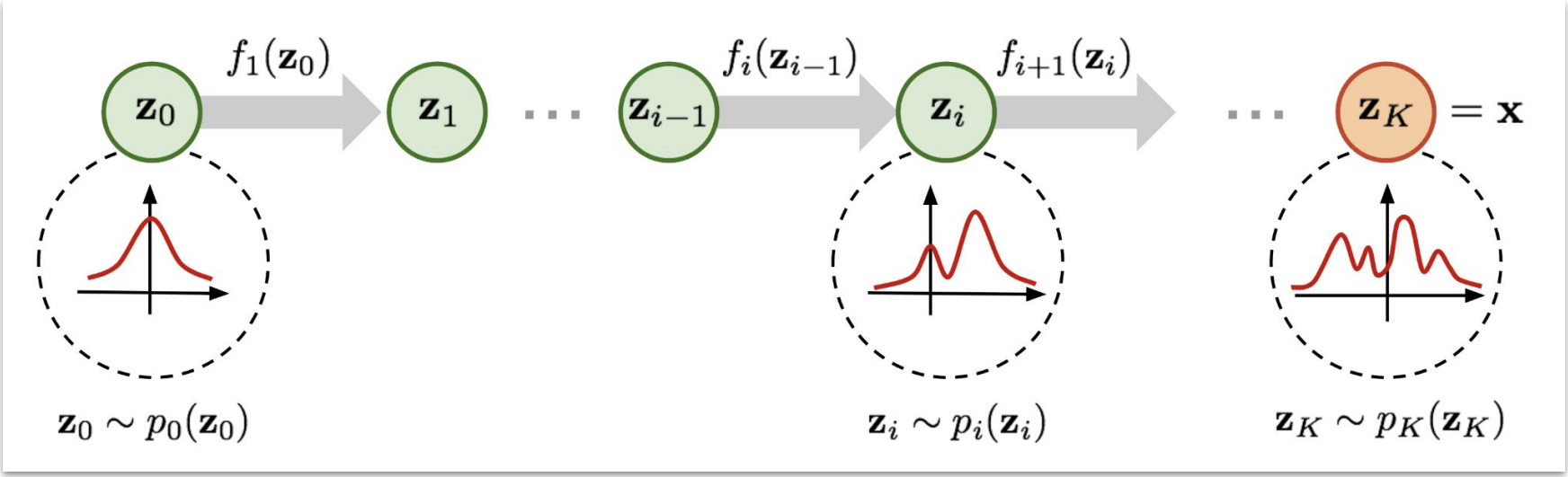
$$\min_{\theta} \max_{\phi} \mathbb{E}_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[\log D_{\phi}(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D_{\phi}(G_{\theta}(\boldsymbol{z}))]$$

MolGAN (Cao and Kipf, 2018)



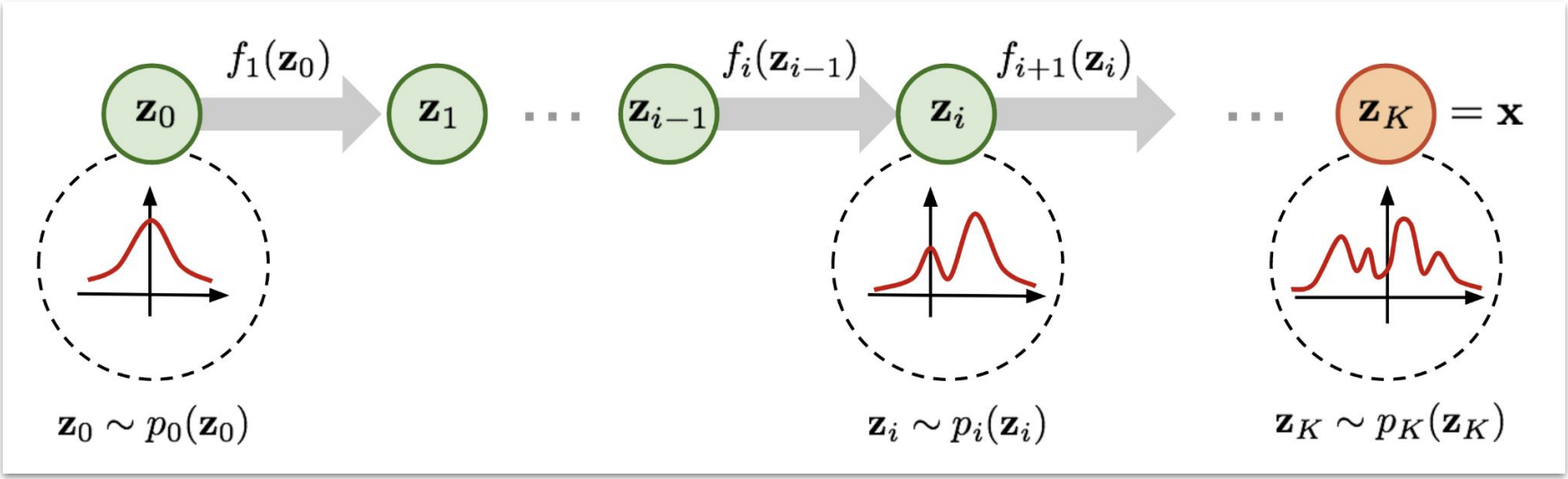
Architektura sieci MolGAN (Cao and Kipf, 2018)

Modele przepływowe (Normalizing Flows)



Działanie modelu opartego o przepływy (Weng, 2018)

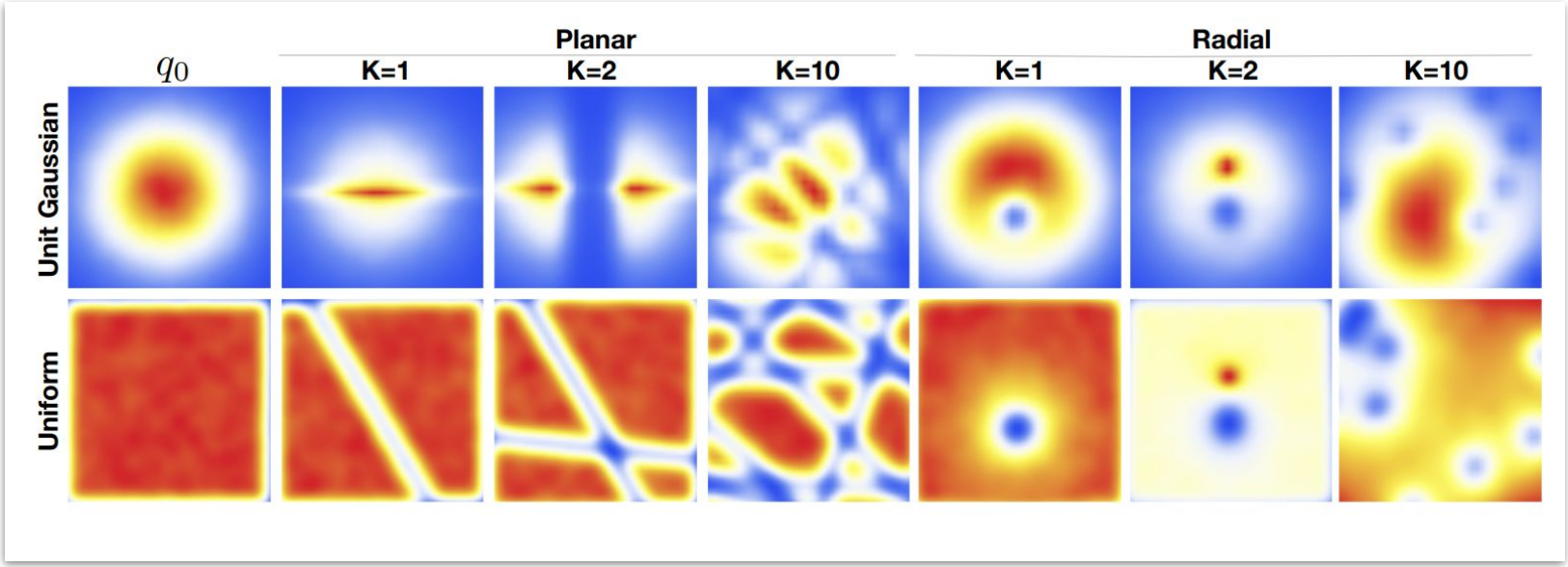
Modele przepływowe (Normalizing Flows)



Działanie modelu opartego o przepływy (Weng, 2018)

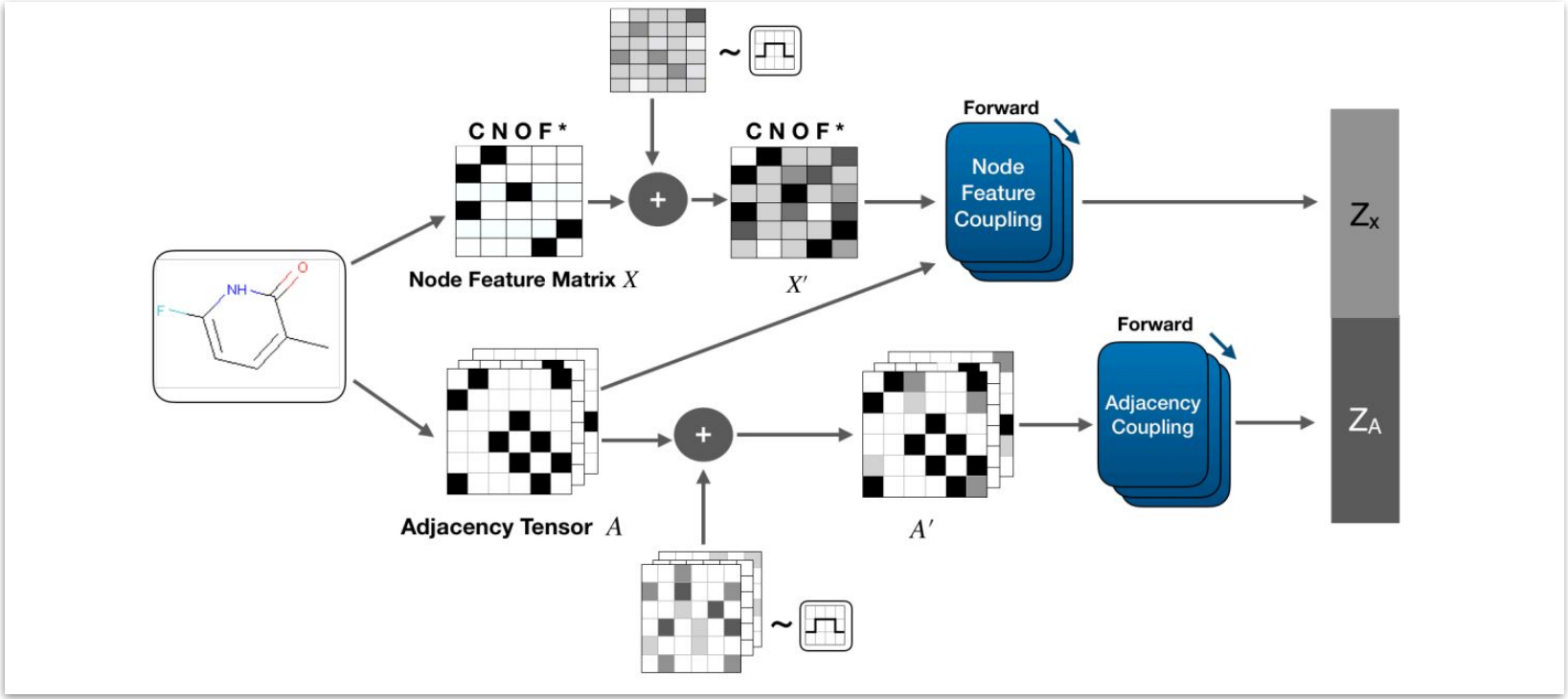
$$p(G) = p(\boldsymbol{z}) \left| \det \left(\frac{\partial f^{-1}(G)}{\partial G} \right) \right|$$

Modele przepływowe (Normalizing Flows)



Działanie modelu opartego na przykładzie dwóch rozkładów (Rezende & Mohamed, 2015)

GraphNVP (Madhawa et al., 2019)



Architektura modelu GraphNVP (Madhawa et al., 2019)

Equivariant Diffusion Model (EDM) (Hoogeboom et al., 2022)

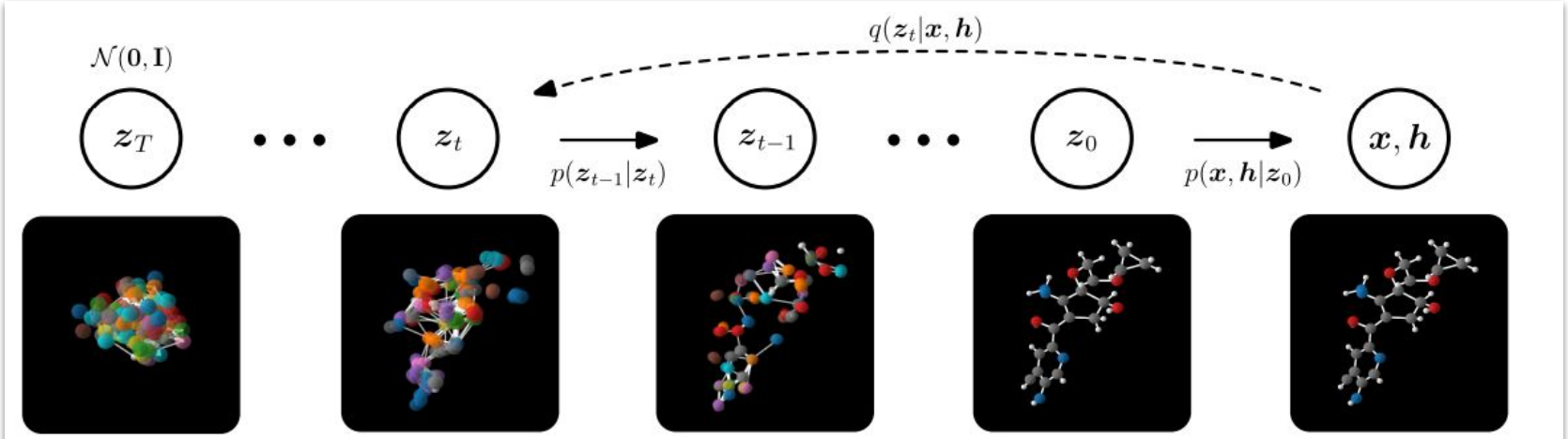


Figure 2. Overview of the Equivariant Diffusion Model. To generate molecules, coordinates $\boldsymbol{x}$ and features $\boldsymbol{h}$ are generated by denoising variables $\boldsymbol{z}_t$ starting from standard normal noise $\boldsymbol{z}_T$. This is achieved by sampling from the distributions $p(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$ iteratively. To train the model, noise is added to a datapoint $\boldsymbol{x},\boldsymbol{h}$ using $q(\boldsymbol{z}_t|\boldsymbol{x},\boldsymbol{h})$ for the step t of interest, which the network then learns to denoise.

Generowanie molekuł modelem EDM (Hoogeboom et al.,2022)

Equivariant Diffusion Model (EDM) (Hoogeboom et al., 2022)

- **Forward diffusion** (zaszumianie) - w danej chwili *t* dodajemy szum do cech, tak że na końcu procesu zaszumiania uzyskujemy szum Gaussowski
- **Reverse diffusion** (odszumianie) - w danej chwili *t* chcemy, aby model przewidział aktualny szum, po odjęciu którego otrzymujemy wygenerowany graf
- Dodatkowo możemy warunkować model, tak aby generował molekuły o zadanych własnościach

EDM - wizualizacja

Generowanie molekuł modelem EDM (Hoogeboom, Twitter)

Podsumowanie

- Modele generatywne dla grafów adaptują metody dotychczas znane z generowania obrazów czy tekstu
- Generowanie grafów jest złożonym procesem, w którym napotykamy szereg problemów powodowanych złożoną naturą tych obiektów
- Obecnie wiodącym podejściem jest wykorzystywanie modeli dyfuzyjnych
- Pomimo wielu spektakularnych wyników, generowanie grafów nadal pozostaje dziedziną z wieloma otwartymi problemami…

Przydatne linki

1. yuanqidu/awesome-graph-generation - zestawienie publikacji z dziedziny generowania grafów
2. JiaxuanYou/graph-generation: GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models - implementacja kilku modeli autoregresyjnych
3. Denoising Diffusion Generative Models in Graph ML | by Michael Galkin | Towards Data Science - omówienie modeli dyfuzyjnych dla grafów
4. https://github.com/ehoogeboom/e3_diffusion_for_molecules - implementacja modelu dyfuzyjnego EDM
5. GitHub - cvignac/DiGress: code for the paper "DiGress: Discrete Denoising diffusion for graph generation" - implementacja modelu dyfuzyjnego DiGress

Bibliografia

1. Zhu, Yanqiao, et al. "**A survey on deep graph generation: Methods and applications.**" arXiv preprint arXiv:2203.06714 (2022).
2. Li, Y., Vinyals, O., Dyer, C., Pascanu, R., & Battaglia, P. (2018). **Learning deep generative models of graphs.** arXiv preprint arXiv:1803.03324.
3. You, J., Ying, R., Ren, X., Hamilton, W., & Leskovec, J. (2018, July). **GraphRNN: Generating realistic graphs with deep auto-regressive models.** In International conference on machine learning (pp. 5708-5717). PMLR.
4. Liao, R., Li, Y., Song, Y., Wang, S., Hamilton, W., Duvenaud, D. K., ... & Zemel, R. (2019). **Efficient graph generation with graph recurrent attention networks.** Advances in neural information processing systems, 32.
5. Bronstein, M. M., Bruna, J., Cohen, T., & Veličković, P. (2021). **Geometric deep learning: Grids, groups, graphs, geodesics, and gauges.** arXiv preprint arXiv:2104.13478.
6. Veličković, P. (2023). **Everything is connected: Graph neural networks.** Current Opinion in Structural Biology, 79, 102538.
7. Guo, X., & Zhao, L. (2022). **A systematic survey on deep generative models for graph generation.** IEEE Transactions on Pattern Analysis and Machine Intelligence.
8. Madhawa, K., Ishiguro, K., Nakago, K., & Abe, M. (2019). **GraphNVP: An invertible flow model for generating molecular graphs.** arXiv preprint arXiv:1905.11600.
9. Rezende, D., & Mohamed, S. (2015, June). **Variational inference with normalizing flows.** In International conference on machine learning (pp. 1530-1538). PMLR.
10. Hoogeboom, E., Satorras, V. G., Vignac, C., & Welling, M. (2022, June). **Equivariant diffusion for molecule generation in 3d**. In International Conference on Machine Learning (pp. 8867-8887). PMLR.
11. Faskowitz, J., Yan, X., Zuo, X. N., & Sporns, O. (2018). **Weighted stochastic block models of the human connectome across the life span.** Scientific reports, 8(1), 1-16.

Dziękuję za uwagę

Kontakt: jakub.binkowski@pwr.edu.pl