University of Amsterdam
Faculty of Science

Categorical Normalizing Flows via Continuous Transformations

Master Thesis Defense, 25. August 2020

Student: Phillip Lippe

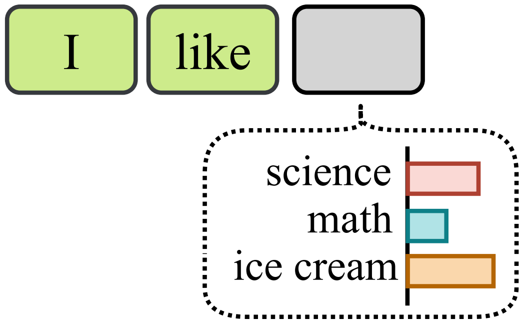
Supervisor: Dr. Efstratios Gavves

Second reader: Dr. Max Welling

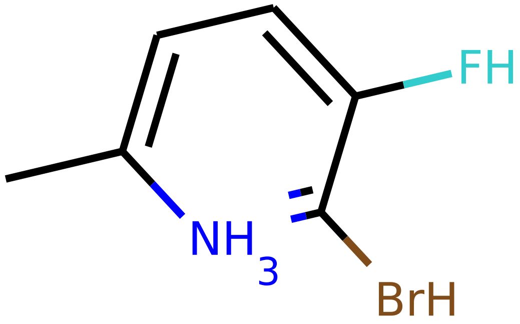
Introduction
Motivation

Categorical Data

Language



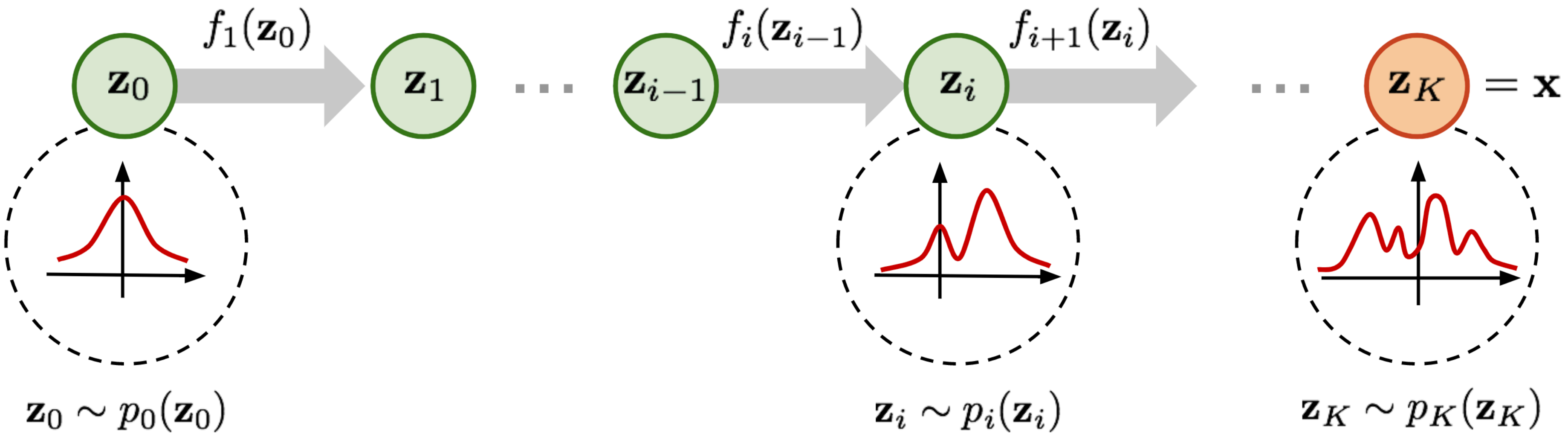
Graphs



Introduction

Preliminaries

Normalizing Flows



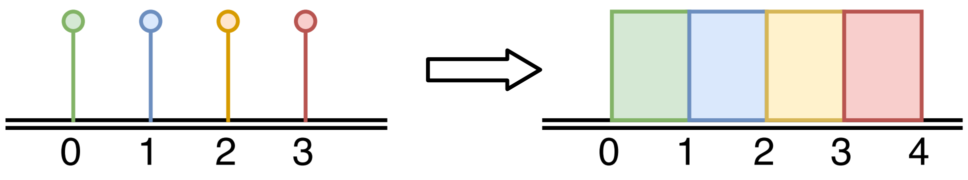
+ Universality
+ Exact likelihood estimate
+ Efficient density evaluation and (parallel) sampling

Figure credit: Weng, Lilian. "Flow-based Deep Generative Models", 2018.

Introduction

Related Work

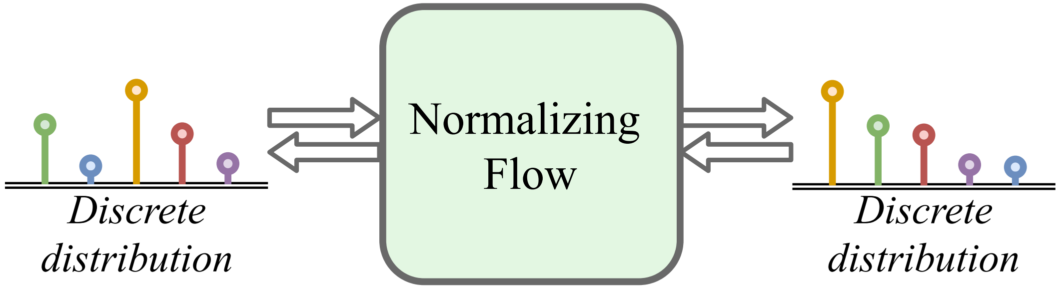
Applying (Variational) Dequantization



Designed for image modeling

– Categories are not “quantized” real values

Discrete Normalizing Flows



– Is limited to permutations
– Not universal with factorized prior
– (Biased) gradient approximations and difficult optimization

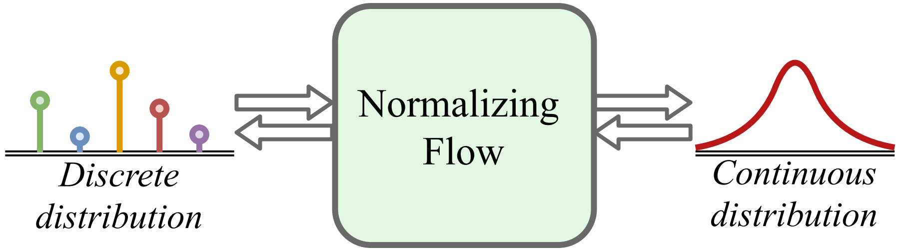
References
Tran, D. et al.: “Discrete Flows: Invertible Generative Models of Discrete Data”. NeurIPS, 2019.
Hoogeboom, E. et al.: “Integer Discrete Flows and Lossless Compression”. NeurIPS, 2019.

Introduction

Contributions

Categorical Normalizing Flow

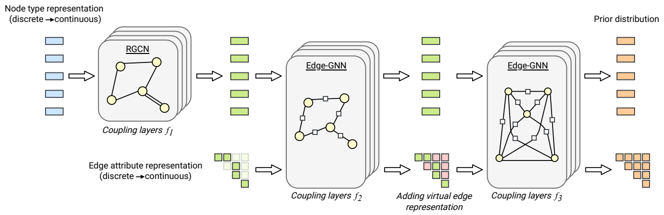
Modeling categorical distribution by a continuous normalizing flow

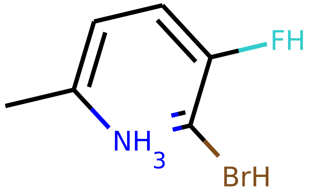


+ Universality
+ Stable optimization without biased gradients
+ Efficient density evaluation and (parallel) sampling

GraphCNF

Powerful graph generation model based on Categorical Normalizing Flows





+ One-shot generation
+ Permutation-invariant to node order
+ Support of categorical node and edge attributes

Categorical Normalizing Flows
Encoding

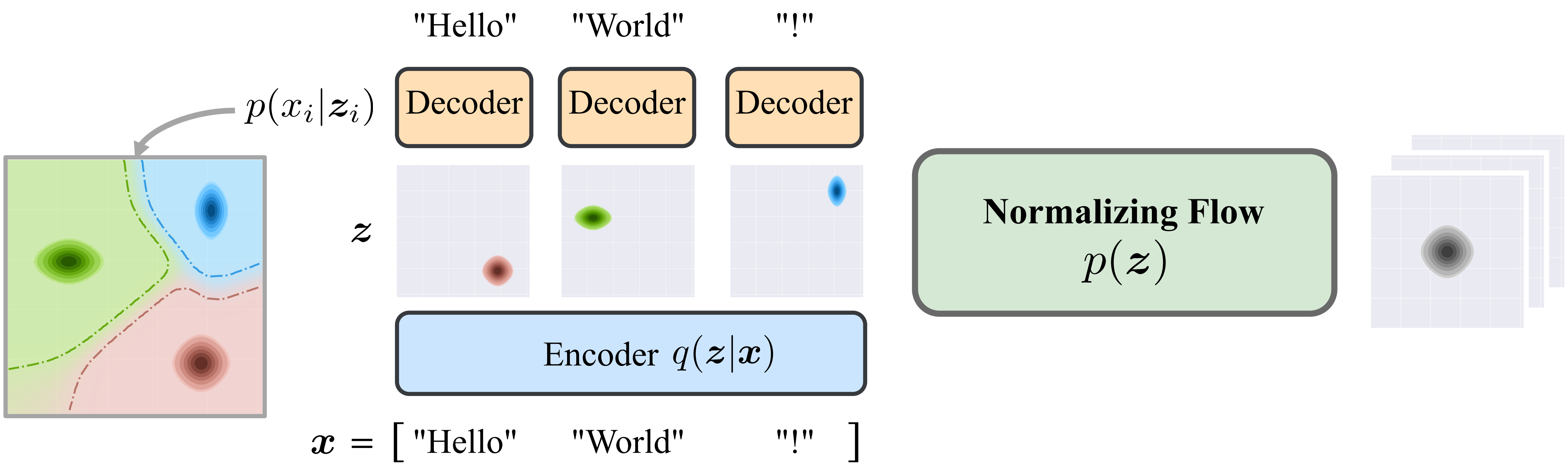
- First step: represent categorical data in continuous space
- Desired properties of an encoding function
  → No loss of information (non-overlapping volumes)
  → Learnable
  → Smooth
  → Support for higher dimensions

⇒ Variational inference with factorized posterior: $p(\boldsymbol{x}) \geq \mathbb{E}_{\boldsymbol{z}\sim q(\cdot|\boldsymbol{x})}\left[\frac{\prod_i p(x_i|\boldsymbol{z}_i)}{q(\boldsymbol{z}|\boldsymbol{x})}p(\boldsymbol{z})\right]$

- Ensures that continuous form z contains the exact same information as discrete x
- Small variational gap ⇒ close-to exact likelihood estimate

Categorical Normalizing Flows

Overview

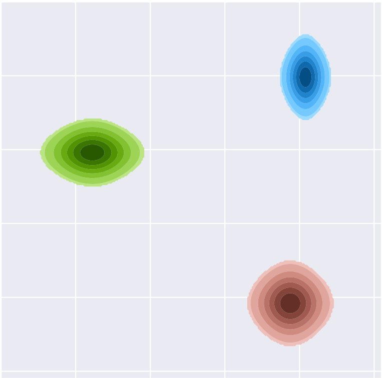


Objective function: $p(\boldsymbol{x}) \geq \mathbb{E}_{\boldsymbol{z}\sim q(\cdot|\boldsymbol{x})}\left[\frac{\prod_i p(x_i|\boldsymbol{z}_i)}{q(\boldsymbol{z}|\boldsymbol{x})}p(\boldsymbol{z})\right]$

Categorical Normalizing Flows

Encoding function

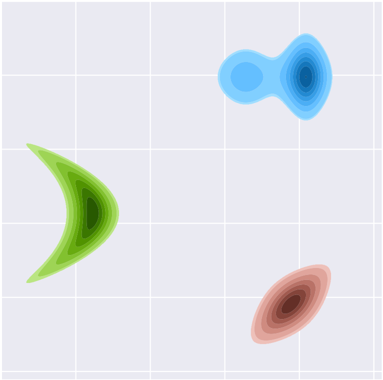
Mixture model



$$q(\boldsymbol{z}|\boldsymbol{x}) = \prod_{i=1}^{N} g(\boldsymbol{z}_i|\mu(x_i), \sigma(x_i))$$

- Exact posterior can be found
  ⇒ no "variational" gap

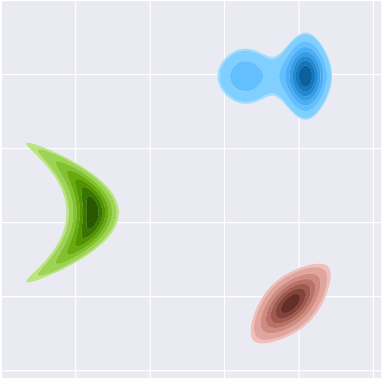
Linear flows



$$q(\boldsymbol{z}|\boldsymbol{x}) = \prod_{i=1}^{N} q(\boldsymbol{z}_i|x_i)$$

- One flow per category
- Shared across input

Variational encoding



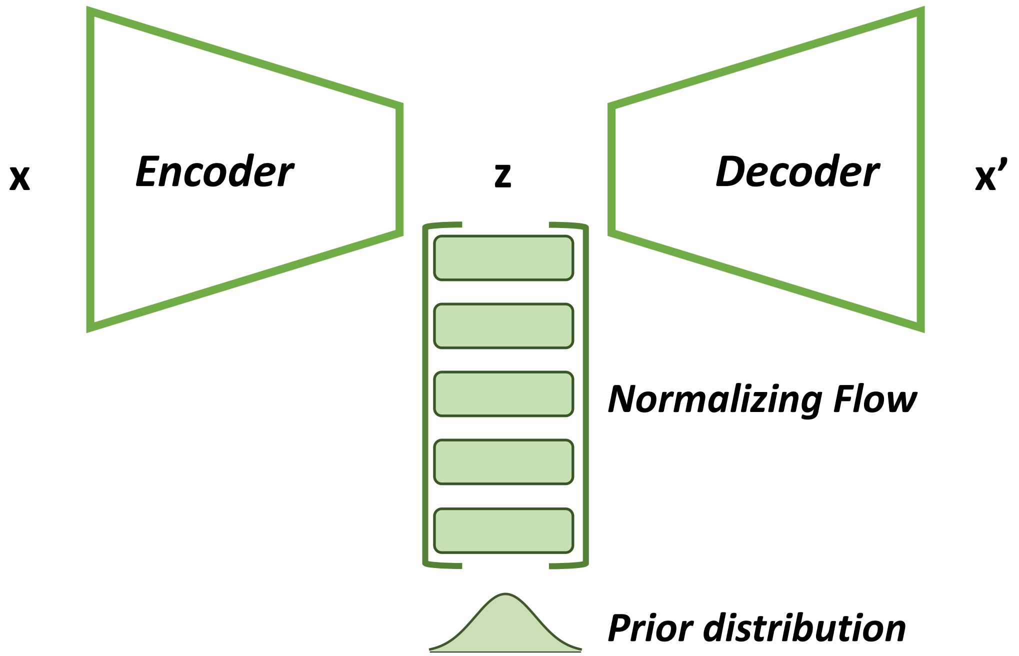
$$q(\boldsymbol{z}|\boldsymbol{x})$$

- One flow on the whole discrete input
- Similar to variational dequantization

Categorical Normalizing Flows

Related Work – Flow-based Variational Autoencoders

Latent Normalizing Flows



- Non-factorized posterior
- Complexity split between decoder and normalizing flow
- Instable training
- Performance worse than non-latent baseline

References
Ziegler, Z.M. and Rush, A.M.: "Latent Normalizing Flows for Discrete Sequences". ICML, 2019.

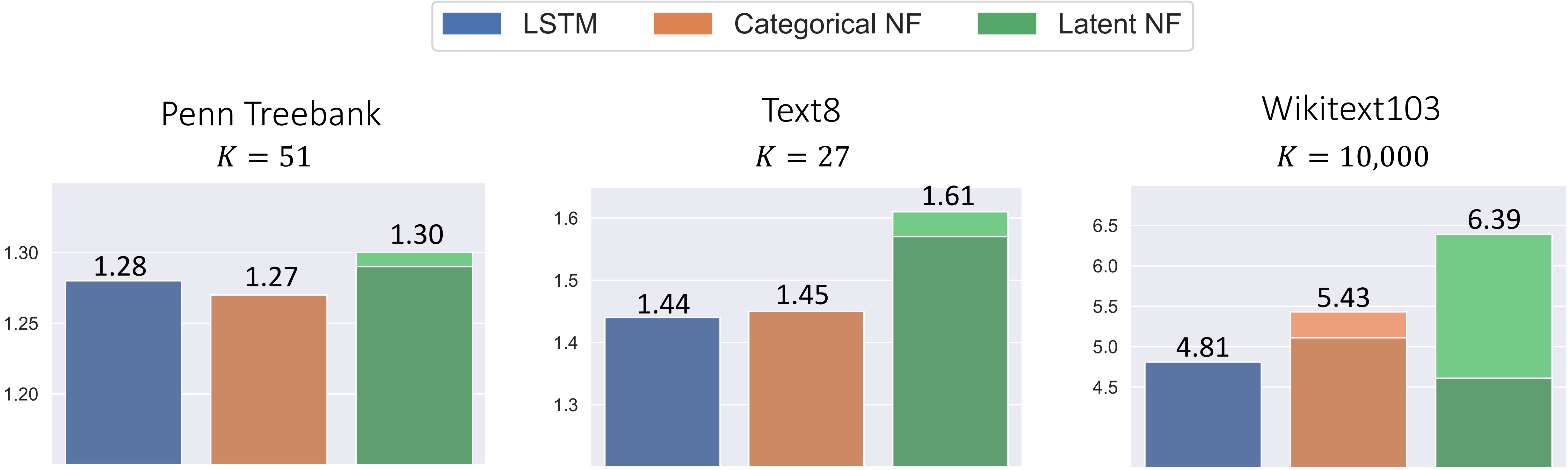
Categorical Normalizing Flows
Experiments – Set Modeling

- Toy datasets on sets with known dataset likelihood
- **Metric**: test likelihood in bits per categorical variable (lower = better)

Model	**Set shuffling**	**Set summation**
Discrete NF	3.87 $\pm$0.04	2.51 $\pm$0.00
Variational Dequantization	3.01 $\pm$0.02	2.29 $\pm$0.01
Latent NF	**2.78** $\pm$0.00	2.26 $\pm$0.01
CNF + Mixture model	**2.78** $\pm$0.00	**2.24** $\pm$0.00
CNF + Linear flows	**2.78** $\pm$0.00	2.25 $\pm$0.00
CNF + Variational Encoding	2.79 $\pm$0.01	2.25 $\pm$0.01
Optimal	2.77	2.24

Categorical Normalizing Flows

Experiments – Language Modeling



Metric: bits per character/word

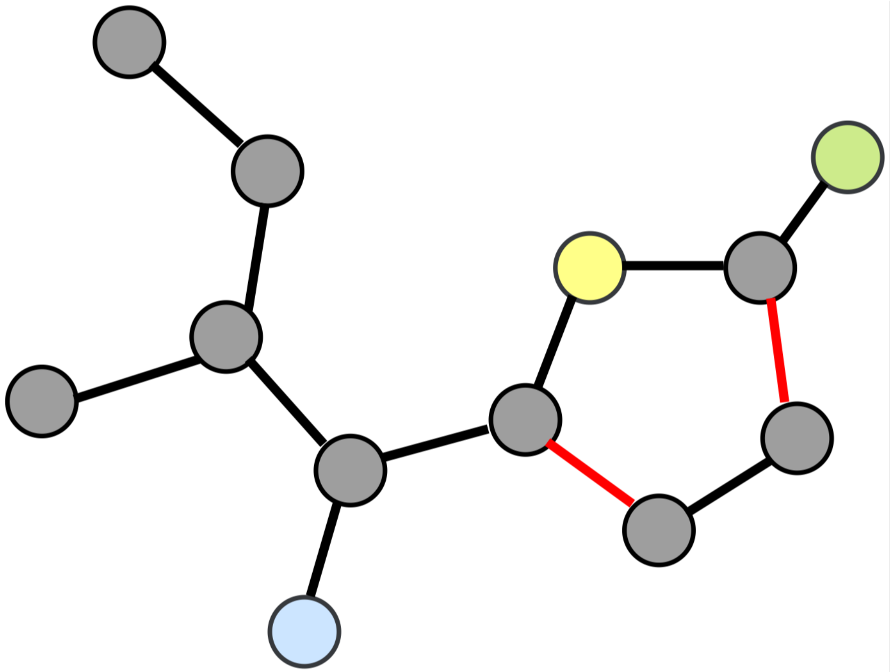
Categorical Normalizing Flows

Conclusion

- Categorical Normalizing Flow is a powerful framework to accurately model categorical distributions
- Encoding function: Mixture model efficient with exact posterior
- Factorized posterior pushes all complexity into the prior

Graph Generation with CNF

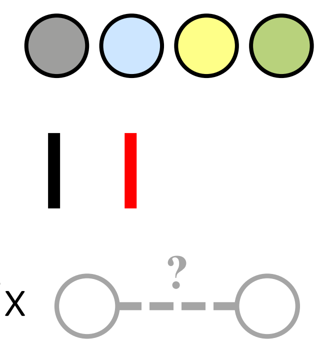
Introduction



(1) Node attributes

(2) Edge attributes

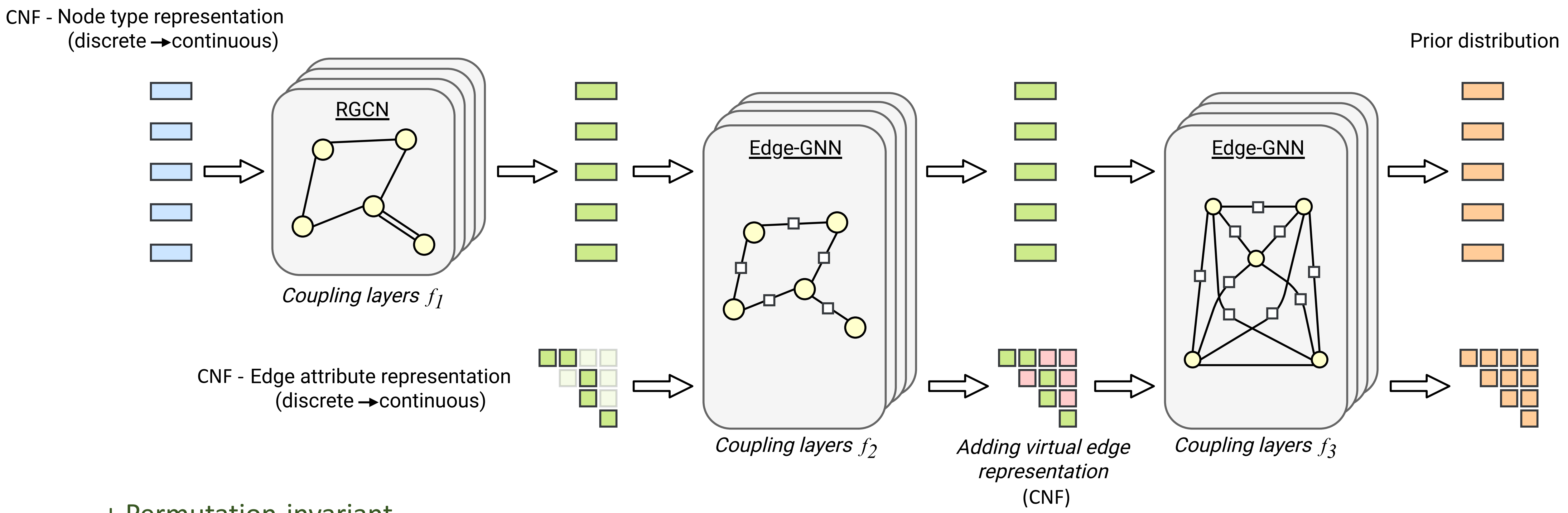
(3) Adjacency matrix



Challenge: nodes are unordered, i.e. a set

⇒ Maintain permutation-invariance of nodes

Graph Generation with CNF
GraphCNF



+ Permutation-invariant
+ Efficient three-step approach

Graph Generation with CNF

Experiments – Molecule Generation

- **Task**: given a set of molecules, learn to model the space of valid molecules
- **Metrics**: calculated on 10k generated graphs,

  (1) Validity: percentage of graphs being valid molecules

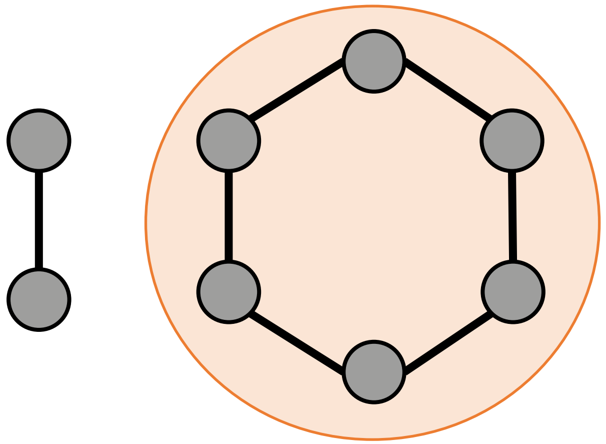
  (2) Uniqueness: percentage of unique molecules

  (3) Novelty: percentage of molecules that are not equal to any training molecule

  (4) Reconstruction: reconstruction accuracy of test molecules from latent space

Graph Generation with CNF

Experiments – Molecule Generation

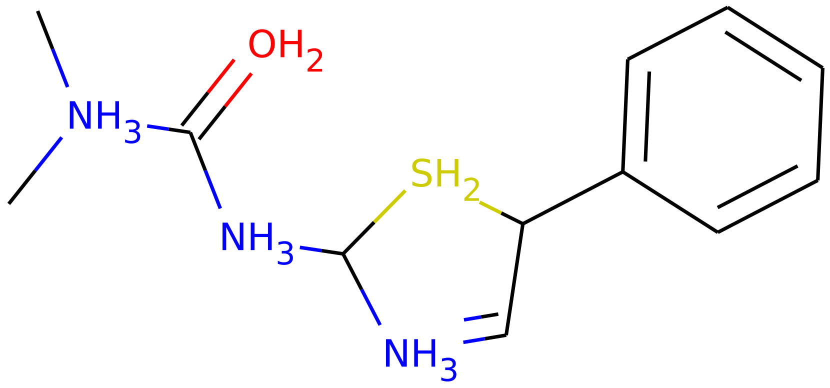


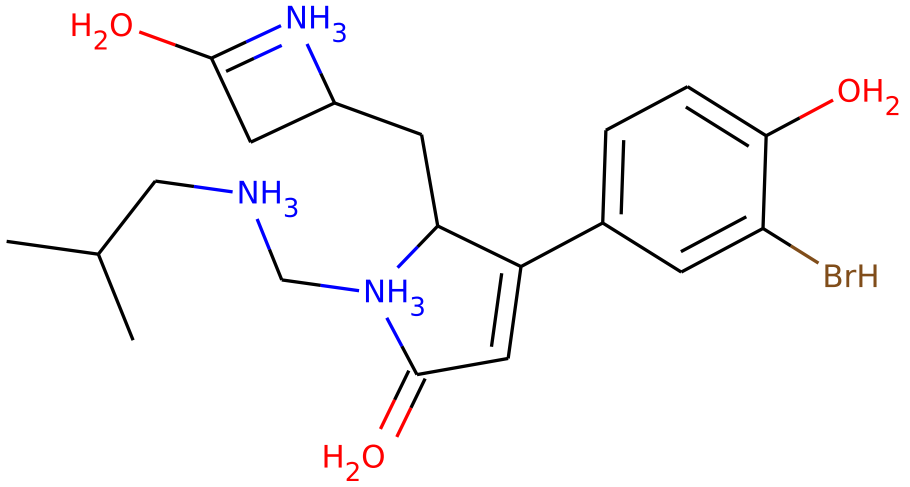
Results on the Zinc250k dataset
(224k examples)

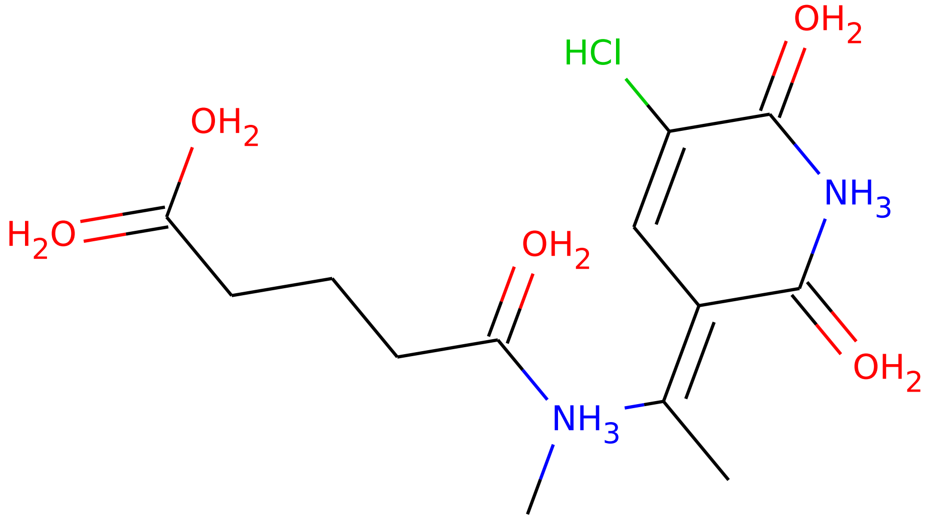
Method	Validity	Uniqueness	Novelty	Reconstruction	Parallel	General
JT-VAE	100%	100%	100%	71%	✗	✗
GraphAF	68%	99.10%	100%	100%	✗	✓
R-VAE	34.9%	100%	–	54.7%	✓	✓
GraphNVP	42.60%	94.80%	100%	100%	✓	✓
GraphCNF	83.41% ($\pm$2.88)	99.99% ($\pm$0.01)	100% ($\pm$0.00)	100% ($\pm$0.00)	✓	✓
+ Sub-graphs	96.35% ($\pm$2.21)	99.98% ($\pm$0.01)	99.98% ($\pm$0.02)	100% ($\pm$0.00)	✓	✓

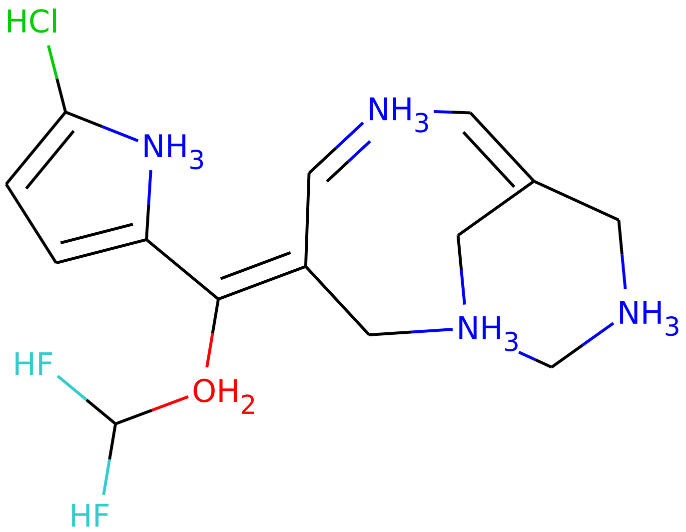
Graph Generation with CNF

Experiments – Molecule Generation









Conclusion

- Mixture model encoding is the “dequantization” for categorical data
  - Simple, efficient and exact in likelihood (continuous)
- CNFs enable strong, latent-based generative models on domains like graphs
  - GraphCNF significantly outperforms previous flow-based approach on molecule generation
- Possible future direction:
  - Combining continuous and discrete normalizing flows
  - GraphCNF on large graphs ($|V| > 100$)

Thank You