Meta

Enhancing and Evaluating Probabilistic Circuits for High-Resolution Lossless Image Compression

Daniel Severo*, Jingtong Su†, Anji Liu§, Jeff Johnson*, Brian Karrer*
Guy Van den Broeck§, Matthew J. Muckley*, Karen Ullrich*

*Meta AI {dsevero,jhj,briankarrer,mmuckley,karenu}@meta.com
†NYU js12196@nyu.edu
§UCLA {liuanji,guyvdb}@cs.ucla.edu

AI at Meta

Lossless Neural Compression

Lossless Neural Compression

Lossless compression

- density/PMF estimation
- entropy coding (ANS, AC)

Lossless Neural Compression

Lossless compression

- density/PMF estimation
- entropy coding (ANS, AC)

Entropy coding $X_1, \ldots, X_n$ requires computing $P(X_i < x | X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$

Lossless Neural Compression

Lossless compression

- density/PMF estimation
- entropy coding (ANS, AC)

Entropy coding $X_1, \ldots, X_n$ requires computing $P(X_i < x | X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$

Lossless Neural Compression = Model P with a neural network

Lossless Neural Compression

At least 2 ways to define a model for $P_\theta(X_1, \ldots, X_n)$

Lossless Neural Compression

At least 2 ways to define a model for $P_\theta(X_1, \ldots, X_n)$

Bottom-up: $P_{\theta_i}(X_i | X_1, \ldots, X_{i-1})$

Lossless Neural Compression

At least 2 ways to define a model for $P_\theta(X_1, \ldots, X_n)$

Bottom-up: $P_{\theta_i}(X_i | X_1, \ldots, X_{i-1}) \Rightarrow \prod_{i=1}^{n} P_{\theta_i}(X_i | X_1, \ldots, X_{i-1}) = P_\theta(X_1, \ldots, X_n)$

Lossless Neural Compression

At least 2 ways to define a model for $P_\theta(X_1, \ldots, X_n)$

Bottom-up: $P_{\theta_i}(X_i | X_1, \ldots, X_{i-1}) \Rightarrow \prod_{i=1}^{n} P_{\theta_i}(X_i | X_1, \ldots, X_{i-1}) = P_\theta(X_1, \ldots, X_n)$

Top-down: $P_\theta(X_1, \ldots, X_n)$

Lossless Neural Compression

At least 2 ways to define a model for $P_\theta(X_1, \ldots, X_n)$

Bottom-up: $P_{\theta_i}(X_i | X_1, \ldots, X_{i-1}) \Rightarrow \prod_{i=1}^{n} P_{\theta_i}(X_i | X_1, \ldots, X_{i-1}) = P_\theta(X_1, \ldots, X_n)$

Top-down: $P_\theta(X_1, \ldots, X_n) \Rightarrow \text{marginalization} \Rightarrow P_{\theta_i}(X_i | X_1, \ldots, X_{i-1})$

Lossless Neural Compression

At least 2 ways to define a model for $P_\theta(X_1, \ldots, X_n)$

Bottom-up: $P_{\theta_i}(X_i | X_1, \ldots, X_{i-1}) \Rightarrow \prod_{i=1}^{n} P_{\theta_i}(X_i | X_1, \ldots, X_{i-1}) = P_\theta(X_1, \ldots, X_n)$

Top-down: $P_\theta(X_1, \ldots, X_n) \Rightarrow \text{marginalization} \Rightarrow P_{\theta_i}(X_i | X_1, \ldots, X_{i-1})$

Probabilistic Circuits are **top-down** models with **efficient marginalization** properties

Chow-Liu Trees (Chow & Liu, 1968)

Chow-Liu Trees (Chow & Liu, 1968)

Given a joint $P(X_1, \ldots, X_n)$, what's the best model we can construct from products of conditionals

$$Q(X_1, \ldots, X_n) = \prod_i P(X_i \mid X_{j(i)})$$?

Chow-Liu Trees (Chow & Liu, 1968)

Given a joint $P(X_1, \ldots, X_n)$, what's the best model we can construct from products of conditionals

$$Q(X_1, \ldots, X_n) = \prod_i P(X_i \mid X_{j(i)}) \ ?$$

Goodness is measured by $\mathrm{KL}\left[P || Q\right]$

Chow-Liu Trees (Chow & Liu, 1968)

Given a joint $P(X_1, \ldots, X_n)$, what's the best model we can construct from products of conditionals

$$Q(X_1, \ldots, X_n) = \prod_i P(X_i \mid X_{j(i)})$$?

Goodness is measured by $\mathrm{KL}\left[P || Q\right]$

The only optimization variable is the assignment function $j \colon \{1, \ldots, n\} \mapsto \{1, \ldots, n\}$

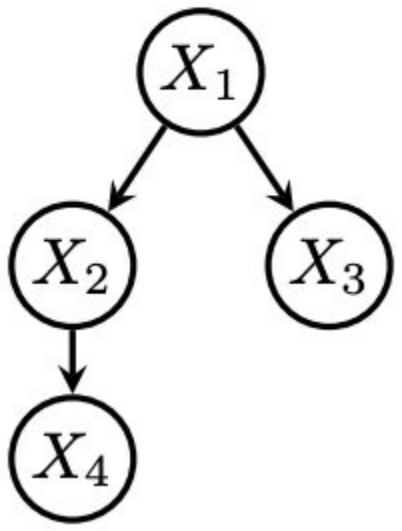
Chow-Liu Trees (Chow & Liu, 1968)

Given a joint $P(X_1, \dots, X_n)$, what's the best model we can construct from products of conditionals

$$Q(X_1, \dots, X_n) = \prod_i P(X_i \mid X_{j(i)})\ ?$$

Goodness is measured by $\mathrm{KL}\left[P||Q\right]$

The only optimization variable is the assignment function $j\colon \{1, \dots, n\} \mapsto \{1, \dots, n\}$



This is equivalent to asking which tree-shaped probabilistic graphical model minimizes the KL divergence?

Chow-Liu Trees (Chow & Liu, 1968)

If Q is a tree, then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = -\sum_i I(X_i; X_{j(i)}) + \text{const}$$

Chow-Liu Trees (Chow & Liu, 1968)

If Q is a tree, then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = -\sum_i I(X_i; X_{j(i)}) + \mathrm{const}$$

Chow-Liu Tree Algorithm:

1) Build a complete weighted graph with MI as edge weights

Chow-Liu Trees (Chow & Liu, 1968)

If Q is a tree, then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = -\sum_i I(X_i; X_{j(i)}) + \text{const}$$

Chow-Liu Tree Algorithm:

1) Build a complete weighted graph with MI as edge weights
2) Find a **maximum spanning tree (MST)**

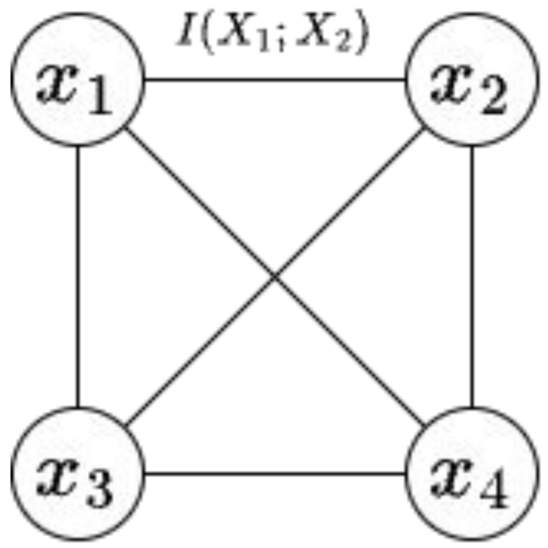
Chow-Liu Trees (Chow & Liu, 1968)

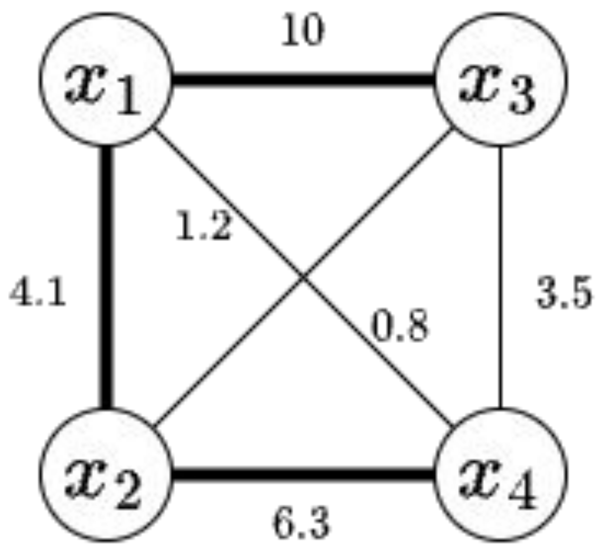
If Q is a tree, then it can be shown that

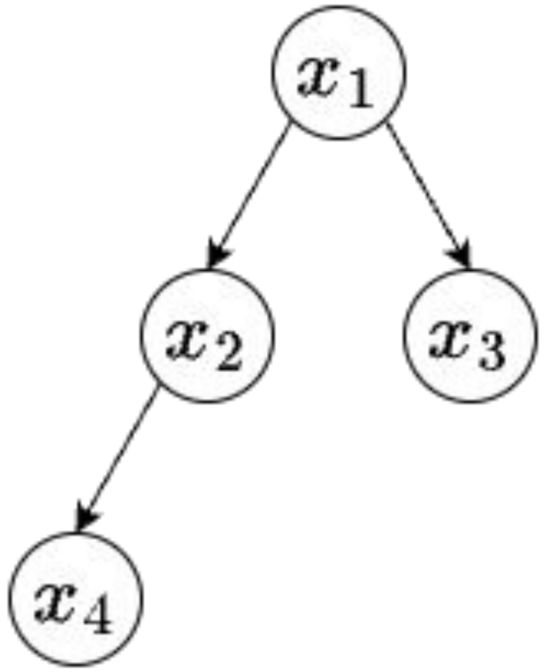
$$\mathrm{KL}\left[P \| Q\right] = -\sum_i I(X_i; X_{j(i)}) + \mathrm{const}$$

Chow-Liu Tree Algorithm:

1) Build a complete weighted graph with MI as edge weights
2) Find a **maximum spanning tree (MST)**







Chow-Liu Trees (this part isn't in the original paper)

If P is not known ....

Chow-Liu Trees (this part isn't in the original paper)

If P is not known ...., then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = \underbrace{-\sum_i I(P_{X_i, X_{J(i)}}) + \Delta H(P_X)}_{\text{Original CLT objective (function of } J \text{ only)}}$$

Chow-Liu Trees (this part isn't in the original paper)

If P is not known ...., then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = \underbrace{-\sum_i I(P_{X_i, X_{J(i)}}) + \Delta H(P_X)}_{\text{Original CLT objective (function of } J \text{ only)}} + \underbrace{\sum_i \mathbb{E}_{X_{J(i)}} \mathrm{KL}\left[P_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)}) \mid\mid Q^{\theta_i, J}_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)})\right]}_{\text{Estimation error (function of both } J, \theta)},$$

where $\Delta H(P_X) = \sum_i H(P_{X_i}) - H(P_X) \geq 0$ with equality when X_i are pair-wise independent.

Chow-Liu Trees (this part isn't in the original paper)

If P is not known …., then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = \underbrace{-\sum_i I(P_{X_i,X_{J(i)}}) + \Delta H(P_X)}_{\text{Original CLT objective (function of } J \text{ only)}} \underbrace{+\sum_i \mathbb{E}_{X_{J(i)}} \mathrm{KL}\left[P_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)}) \,||\, Q^{\theta_i,J}_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)})\right]}_{\text{Estimation error (function of both } J,\theta)},$$

where $\Delta H(P_X) = \sum_i H(P_{X_i}) - H(P_X) \geq 0$ with equality when X_i are pair-wise independent.

Joint optimization over (θ, J) is difficult

Chow-Liu Trees (this part isn't in the original paper)

If P is not known …., then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = \underbrace{-\sum_i I(P_{X_i, X_{J(i)}}) + \Delta H(P_X)}_{\text{Original CLT objective (function of } J \text{ only)}} \; \underbrace{+ \sum_i \mathbb{E}_{X_{J(i)}} \mathrm{KL}\left[P_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)}) \;||\; Q^{\theta_i, J}_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)})\right]}_{\text{Estimation error (function of both } J, \theta)},$$

where $\Delta H(P_X) = \sum_i H(P_{X_i}) - H(P_X) \geq 0$ with equality when X_i are pair-wise independent.

Joint optimization over (θ, J) is difficult

Pragmatic solution: two-step process

Chow-Liu Trees (this part isn't in the original paper)

If P is not known …., then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = \underbrace{-\sum_i I(P_{X_i, X_{J(i)}}) + \Delta H(P_X)}_{\text{Original CLT objective (function of } J \text{ only)}} + \underbrace{\sum_i \mathbb{E}_{X_{J(i)}} \mathrm{KL}\left[P_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)}) \,||\, Q^{\theta_i, J}_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)})\right]}_{\text{Estimation error (function of both } J, \theta)},$$

where $\Delta H(P_X) = \sum_i H(P_{X_i}) - H(P_X) \geq 0$ with equality when X_i are pair-wise independent.

Joint optimization over (θ, J) is difficult

Pragmatic solution: two-step process

- Estimate the MI of every pair $I(X_i; X_j)$, then find the MST (equivalently, J*)

Chow-Liu Trees (this part isn't in the original paper)

If P is not known …., then it can be shown that

$$\mathrm{KL}\left[P \| Q\right] = \underbrace{-\sum_i I(P_{X_i, X_{J(i)}}) + \Delta H(P_X)}_{\text{Original CLT objective (function of } J \text{ only)}} \; + \underbrace{\sum_i \mathbb{E}_{X_{J(i)}} \mathrm{KL}\left[P_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)}) \;||\; Q^{\theta_i, J}_{X_i \mid X_{J(i)}}(\cdot \mid X_{J(i)})\right]}_{\text{Estimation error (function of both } J, \theta)},$$

where $\Delta H(P_X) = \sum_i H(P_{X_i}) - H(P_X) \geq 0$ with equality when X_i are pair-wise independent.

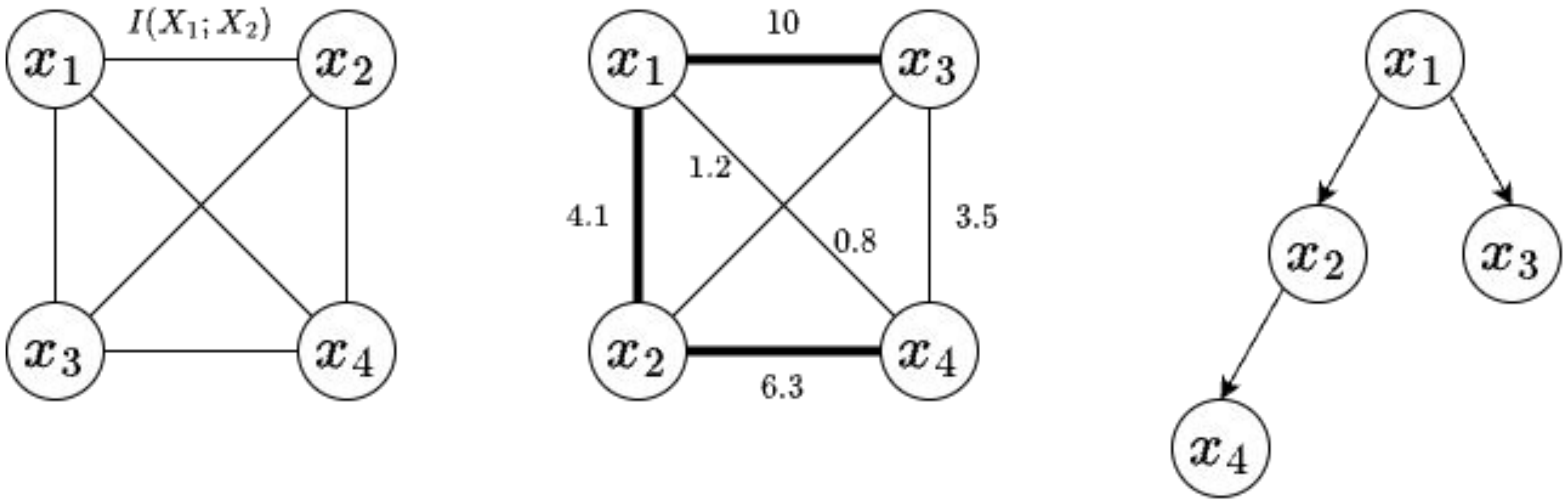
Joint optimization over (θ, J) is difficult

Pragmatic solution: two-step process

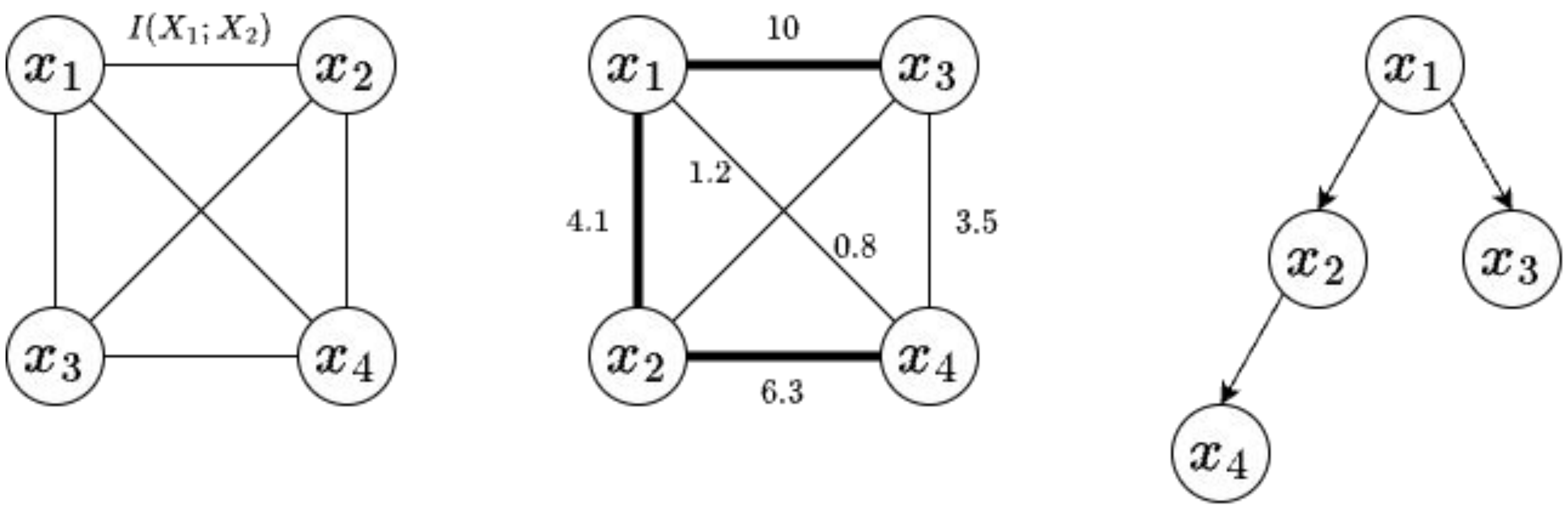
- Estimate the MI of every pair $I(X_i; X_j)$, then find the MST (equivalently, J*)
- Minimize estimation error by optimizing $Q^{\theta_i, J}_{X_i | X_{J(i)}}$

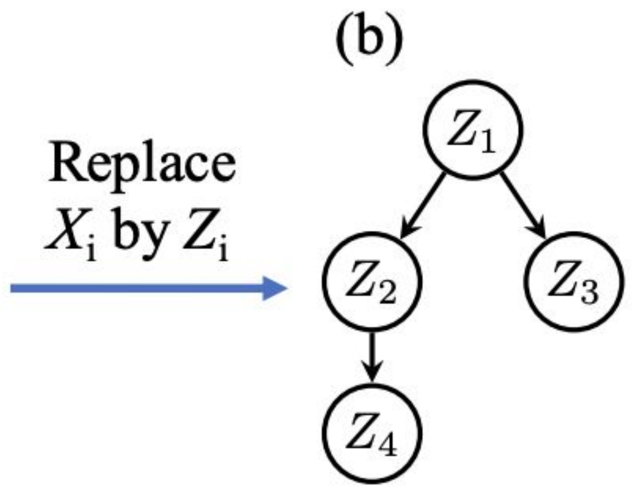
Hidden CLTs (Liu & Van den Broeck, 2021)

Hidden CLTs (Liu & Van den Broeck, 2021)

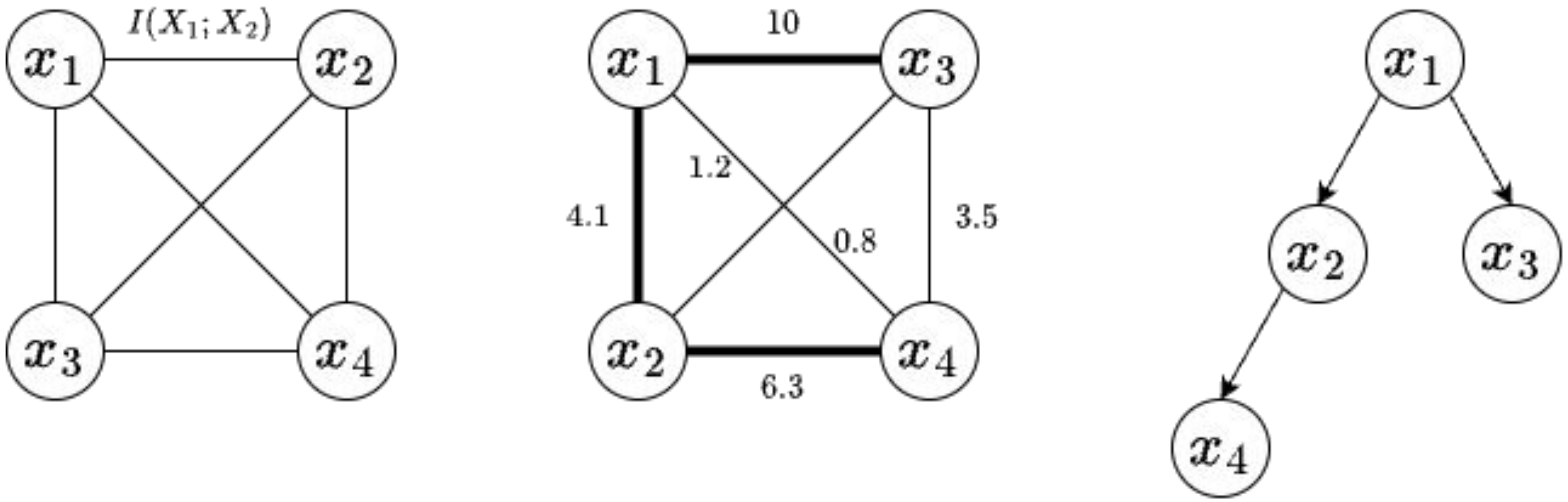


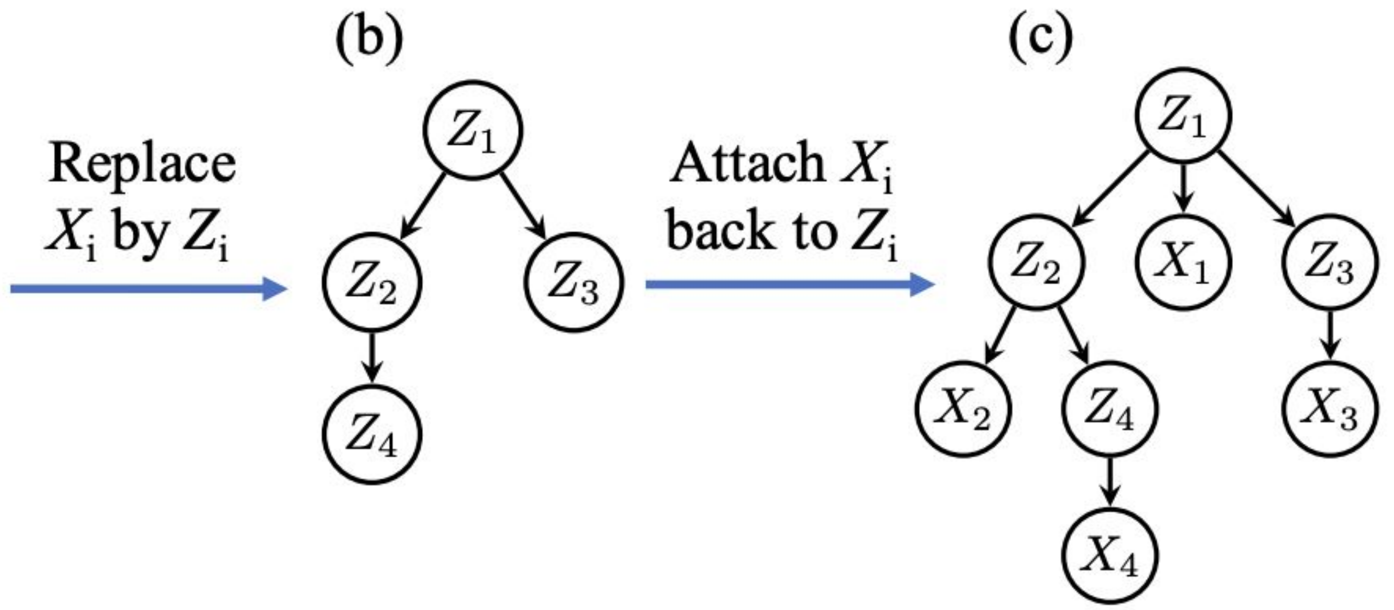
Hidden CLTs (Liu & Van den Broeck, 2021)



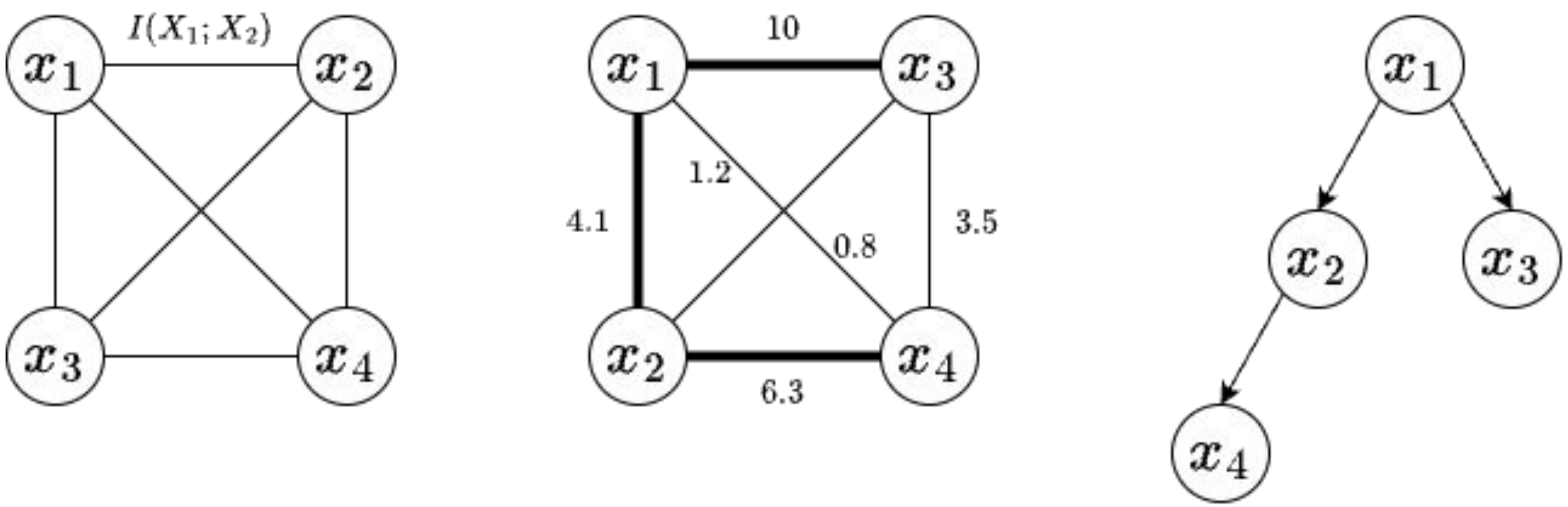


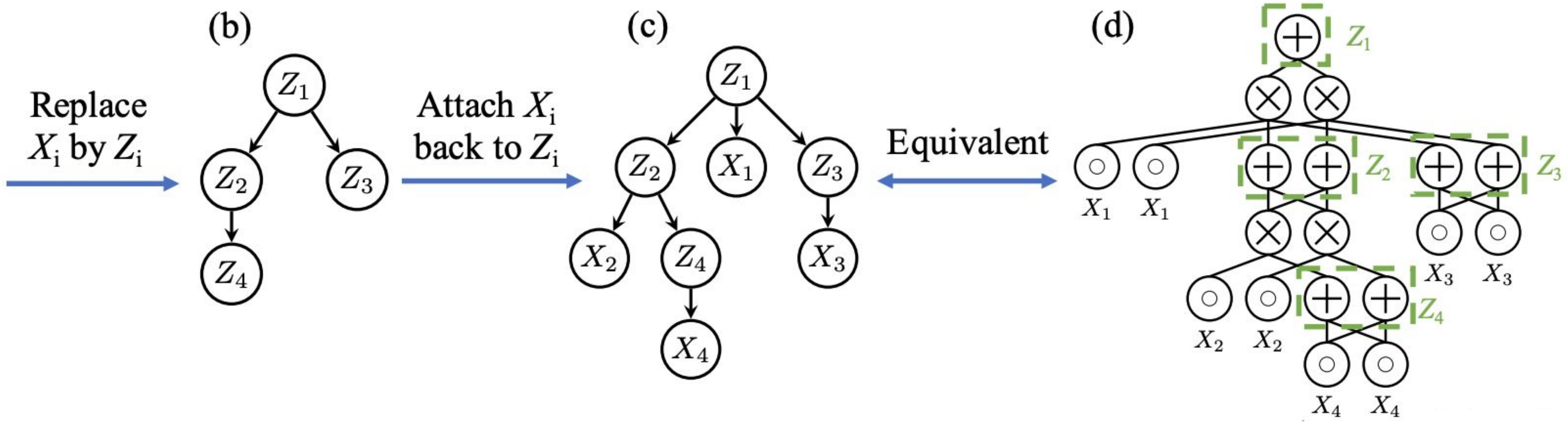
Hidden CLTs (Liu & Van den Broeck, 2021)





Hidden CLTs (Liu & Van den Broeck, 2021)





Our contributions

Sparse Mutual Information Estimation

Estimating MI scales quadratically with n

Sparse Mutual Information Estimation

Estimating MI scales quadratically with n

For images, MI will correlate with pixel distance

Sparse Mutual Information Estimation

Estimating MI scales quadratically with n

For images, MI will correlate with pixel distance

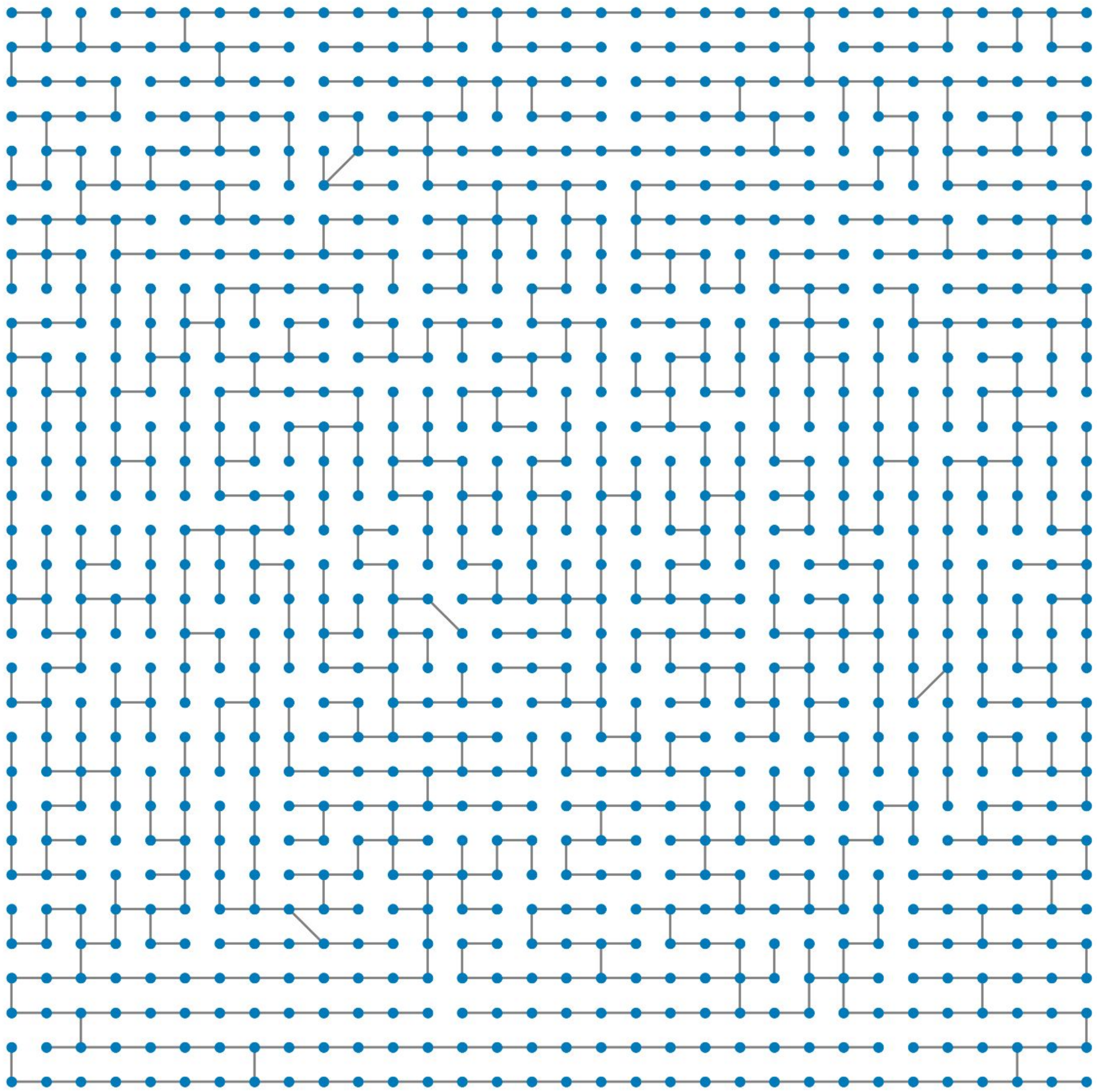
Solution: only compute MI between neighbouring pixels, set others to 0

Sparse Mutual Information Estimation

Estimating MI scales quadratically with n

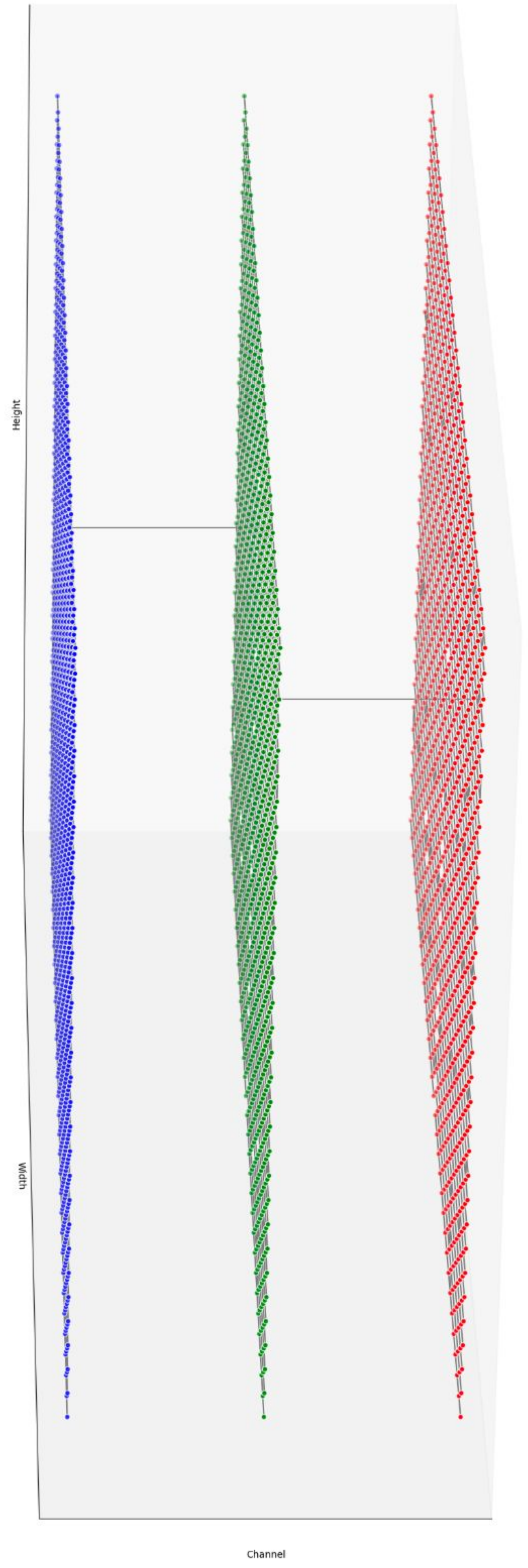
For images, MI will correlate with pixel distance

Solution: only compute MI between neighbouring pixels, set others to 0

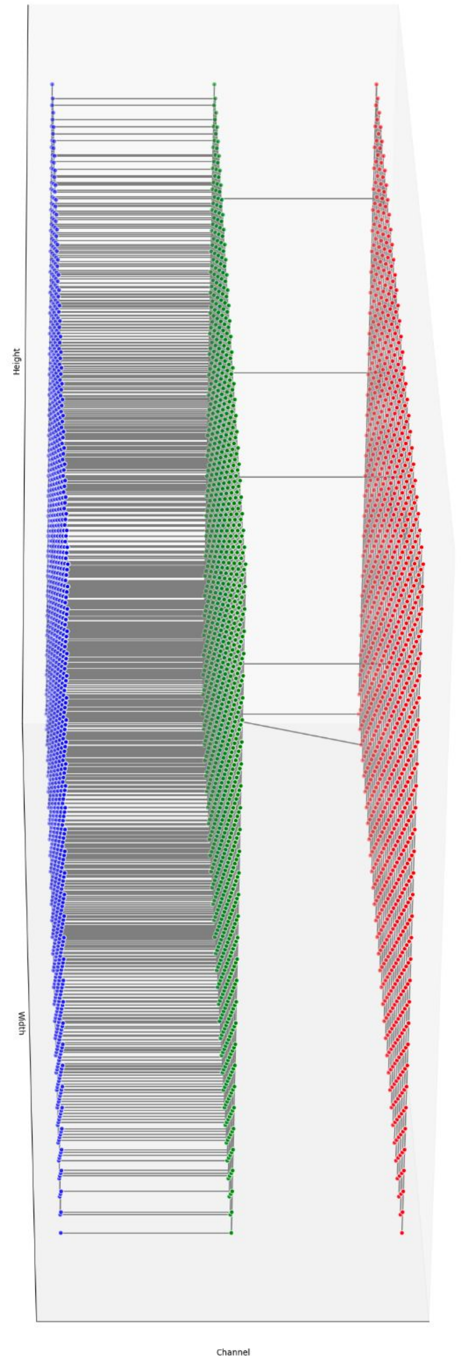


Repurpose Invertible Transforms from WebP

ConvertColorSpace (RGB or YCoCg)



PredictTransformThenConvertColorspacePipeline (YCoCg)

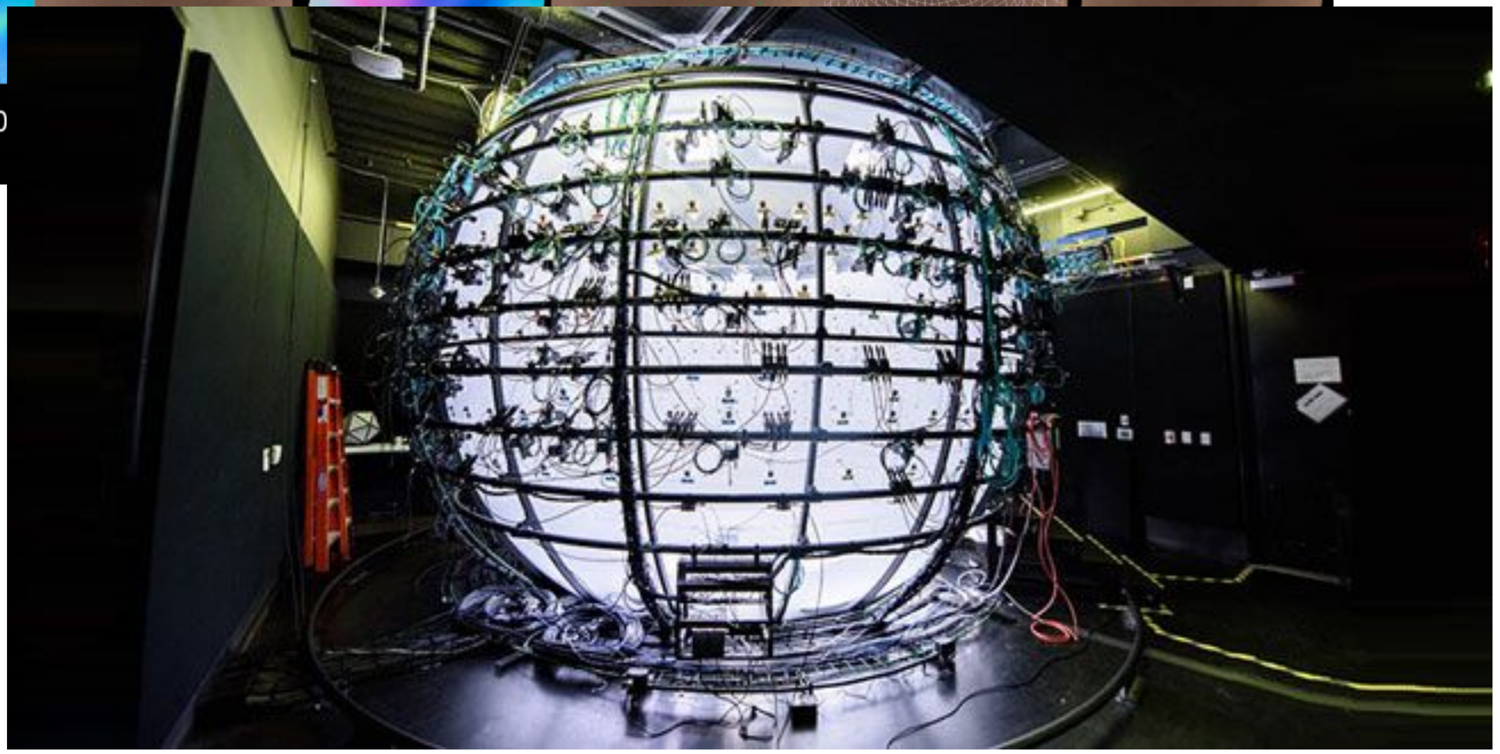


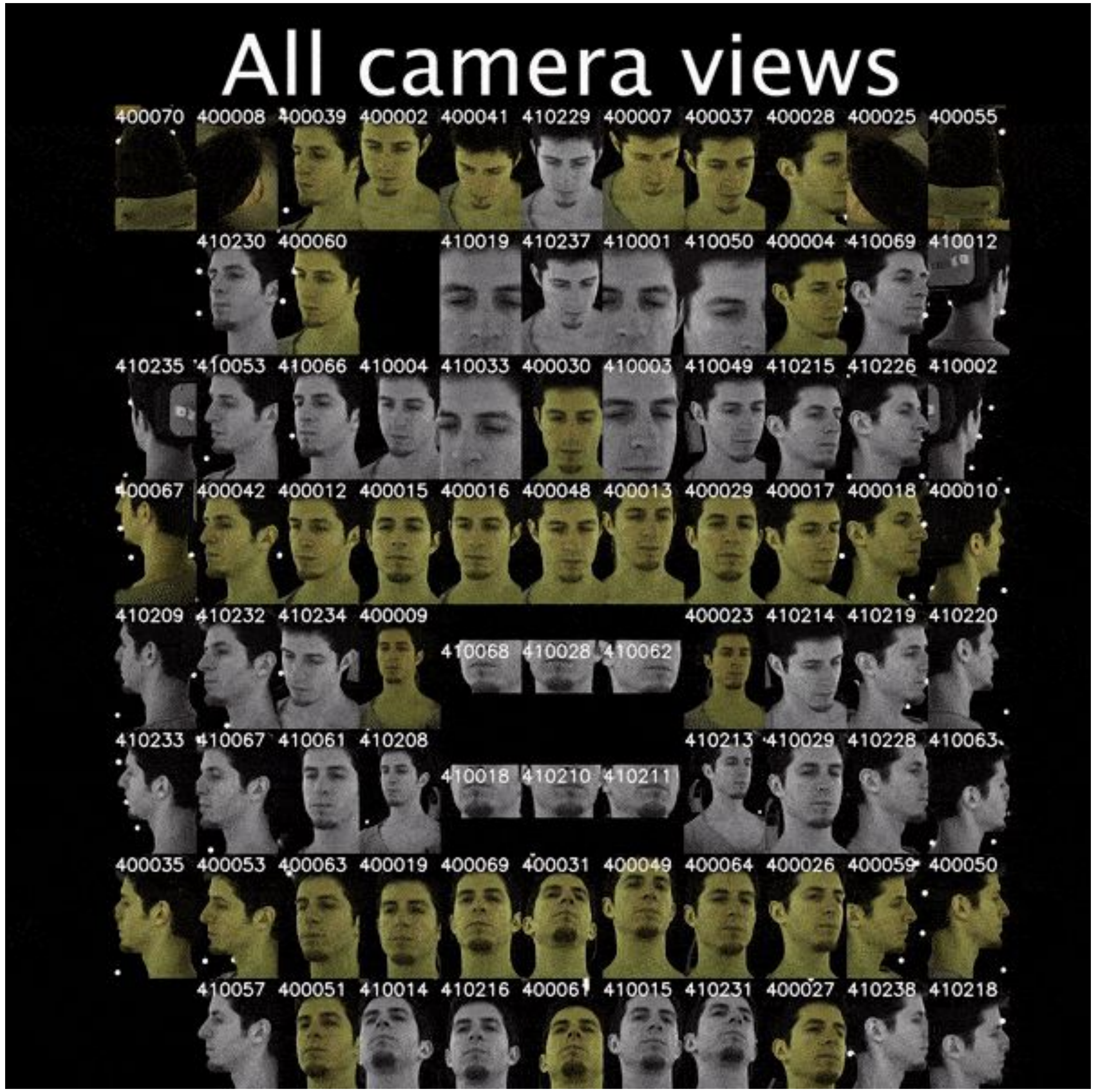
Only improved performance of PCs, not other neural compressors we tried!

Scaling up to AR/VR avatar datasets

Dataset to train 3D renderings of people created by Codec Avatar team at Meta

- publicly released on github.
- 50TB uncompressed images
- resolution of 2048 × 1334 pixels





Experiments

Table 1: PC results compared to standard codecs and neural compressors. PCs provide a gain in compression performance relative to standard codecs, while not requiring as much compute as neural compressors. The advantage of both PCs and neural codecs diminishes as the average image size of the dataset increases (left-to-right). All units are bits-per-dimension.

		CIFAR	IM32	IM64	CLIC	MF
Standard Codecs	PNG(RGB)	5.87	6.05	5.34	3.49	2.87
	PNG(YCoCg)	5.23	5.54	4.88	3.13	3.01
	WebP(YCoCg)	4.87	5.20	4.51	2.68	2.66
	WebP(RGB)	**4.61**	**4.98**	**4.30**	**2.59**	**2.61**
PCs	HCLT	6.04	6.16	5.92	4.10	3.12
	HCLT++	**4.13**	**4.72**	**4.29**	**2.48**	**2.75**
Neural Codecs	HiLLoC	3.56	4.20	3.90	2.63	-
	IDF	3.32	3.95	3.66	2.43	2.57
	IDF++	**3.26**	**3.94**	**3.62**	**2.44**	**2.54**
Adv. of best Prob. Circuit over WebP		10.8%	5.8%	1.0%	4.0%	-5.4%
Adv. of best Neural Codec over WebP		29.2%	20.8%	15.9%	6.1%	2.7%

Obrigado! / Thanks!

Slides will be available at dsevero.com

References

[1] (2020) Probabilistic Circuits: A Unifying Framework for Tractable Probabilistic Models

https://yoojungchoi.github.io/files/ProbCirc20.pdf

[2] (2022) Lossless Compression with Probabilistic Circuits

https://arxiv.org/pdf/2111.11632.pdf

[3] (2020) Group Fairness by Probabilistic Modeling with Latent Fair Decisions

https://arxiv.org/pdf/2009.09031.pdf

This paper outlines the full EM procedure they adopt in [2]

[4] (2022) Sparse Probabilistic Circuits via Pruning and Growing

https://arxiv.org/pdf/2211.12551.pdf

[5] (2020) Random Sum-Product Networks: A Simple and Effective Approach to Probabilistic Deep Learning

http://proceedings.mlr.press/v115/peharz20a/peharz20a.pdf

[6] (2020) Einsum Networks: Fast and Scalable Learning of Tractable Probabilistic Circuits

https://arxiv.org/abs/2004.06231

https://github.com/cambridge-mlg/EinsumNetworks

[7] (2022) Scaling Up Probabilistic Circuits by Latent Variable Distillation

https://arxiv.org/abs/2210.04398

[8] (2023) Understanding the Distillation Process from Deep Generative Models to Tractable Probabilistic Circuits

https://arxiv.org/abs/2302.08086

Papers highlighted in bold are good starting points.