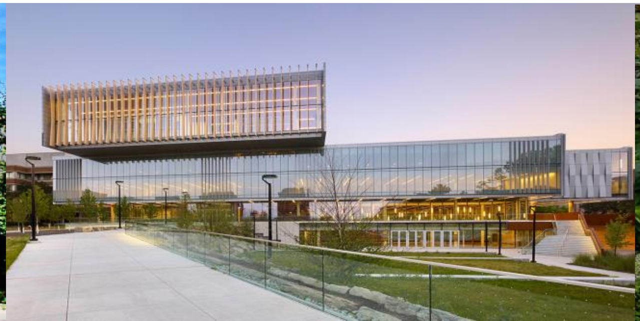
Interpretable Lightweight Transformer via Unrolling of Learned Graph Smoothness Priors

Tam Thuc Do, Parham Eftekhar, Seyed Alireza Hosseini, **Gene Cheung**, Philip Chou*

York University, Canada

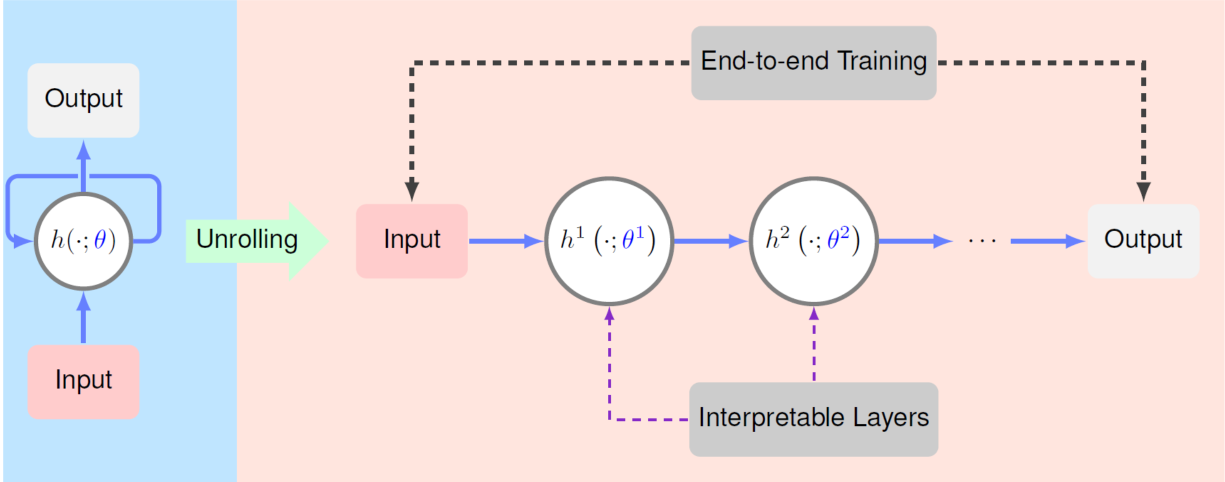
*packet.media, USA

NeurIPS 2024



Unrolling of Graph-based Algorithms

- **Algorithm Unrolling:** implements each iteration of an iterative algorithm as neural layer + parameter tuning [2].
  - e.g., ISTA to LISTA [1].
  - *100% mathematically interpretable*.
  - Train *fewer* parameters.
  - Robust to **covariate shift**.



- **"White-box" transformer** by unrolling ***sparse rate reduction*** algorithm [3].

Our approach:
1. design graph-based algorithms,
2. unroll + parameter tuning.

[1] J. K. Gregor and Y. LeCun, "**Learning fast approximations of sparse coding**," *ICML*, Madison, WI, USA, 2010, ICML'10, p. 399–406.
[2] V. Monga, Y. Li, and Yonina C. Eldar, "**Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing**," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.
[3] Y. Yu, S. Buchanan, D. Pai, T. Chu, Z. Wu, S. Tong, B.D. Haeffele, Y. Ma, "**White-box transformers via sparse rate reduction**," *NeurIPS*, 2023.

GTV for Image Interpolation

- Formulate interpolation problem using **graph total variation** (**GTV)** as objective:

GTV

sampling matrix

$$\min_{\mathbf{x}} \|\mathbf{C}\mathbf{x}\|_1, \quad \text{s.t.} \quad \mathbf{H}\mathbf{x} = \mathbf{y}$$

observation

$$\|\mathbf{Cx}\|_1 = \sum_{(i,j)\in E} w_{i,j}|x_i - x_j|$$

- Rewrite as standard form of ***linear programming*** (LP):

$$\min_{\mathbf{z},\mathbf{x},\mathbf{q}} \mathbf{1}_M^\top \mathbf{z}, \quad \text{s.t.} \quad \underbrace{\begin{bmatrix} \mathbf{I}_M & -\mathbf{C} & -(\mathbf{I}_M \;\; \mathbf{0}_{M,M}) \\ \mathbf{I}_M & \mathbf{C} & -(\mathbf{0}_{M,M} \;\; \mathbf{I}_M) \\ \mathbf{0}_{K,M} & \mathbf{H} & \mathbf{0}_{K,2M} \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \mathbf{z} \\ \mathbf{x} \\ \mathbf{q} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{0}_M \\ \mathbf{0}_M \\ \mathbf{y} \end{bmatrix}}_{\mathbf{b}}, \quad \mathbf{q} \geq \mathbf{0}_{2M}$$

- Solve ***sparse LP*** (SLP) via **ADMM** algorithm in **linear time** [1].
- Can interpret output as **LP filter** of up-sampled input:

upsampling operator

LP filter

$$(\mathbf{C}^\top\mathbf{C} + \mathbf{H}^\top\mathbf{H})\mathbf{x}^{t+1} = \frac{1}{2\gamma}\mathbf{C}^\top\left(\boldsymbol{\mu}_a^t - \boldsymbol{\mu}_b^t + \boldsymbol{\mu}_d^t - \boldsymbol{\mu}_e^t\right) - \frac{1}{\gamma}\mathbf{H}^\top\boldsymbol{\mu}_c^t - \frac{1}{2}\mathbf{C}^\top(\tilde{\mathbf{q}}_1^t - \tilde{\mathbf{q}}_2^t) + \mathbf{H}^\top\mathbf{y}$$

[1] Sinong Wang and Ness Shroff, "**A new alternating direction method for linear programming**," *NeurIPS'17*.

Similarity Graph Learning from Data

- **Graph Edge Definition**: exponential of feature distance.

PSD metric matrix

$$w_{i,j} = \exp\left(-d(i,j)\right), \qquad d(i,j) = (\mathbf{f}_i - \mathbf{f}_j)^\top \mathbf{M} (\mathbf{f}_i - \mathbf{f}_j)$$

edge weight

Mahalanobis distance

feature vector
(computed via ***feature function*** from input embedding)

- With random walk **normalization**, then

$$\bar{w}_{i,j} = \frac{\exp(-d(i,j))}{\sum_{l|(i,l)\in\mathcal{E}} \exp(-d(i,l))}.$$

edge set stemming from node *i*

[1] Do, Tam Thuc, et al. "**Interpretable Lightweight Transformer via Unrolling of Learned Graph Smoothness Priors**." accepted to *NeurIPS'24*.

Self-Attention in Transformer

- **Graph Edge Definition**: exponential of feature distance.

$$w_{i,j} = \exp\left(-d(i,j)\right), \qquad d(i,j) = (\mathbf{f}_i - \mathbf{f}_j)^\top \mathbf{M} (\mathbf{f}_i - \mathbf{f}_j)$$

  - With normalization, then

$$\bar{w}_{i,j} = \frac{\exp(-d(i,j))}{\sum_{l|(i,l)\in\mathcal{E}} \exp(-d(i,l))}.$$

- **Self-Attention Mechanism** in Transformer: dot product of linear transformed embeddings, using ***key*** and ***query*** matrices, **K** and **Q**:

transformed dot product

$$a_{i,j} = \frac{\exp(e(i,j))}{\sum_{l=1}^{N} \exp(e(i,l))}, \qquad e(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{Q}\mathbf{x}_j)^\top (\mathbf{K}\mathbf{x}_i).$$

[1] Do, Tam Thuc, et al. "**Interpretable Lightweight Transformer via Unrolling of Learned Graph Smoothness Priors**." accepted to *NeurIPS'24*.

Self-Attention in Transformer

- **Graph Edge Definition**: exponential of feature distance.

$$w_{i,j} = \exp\left(-d(i,j)\right), \qquad d(i,j) = (\mathbf{f}_i - \mathbf{f}_j)^\top \mathbf{M} (\mathbf{f}_i - \mathbf{f}_j)$$

  - With normalization, then

$$\bar{w}_{i,j} = \frac{\exp(-d(i,j))}{\sum_{l \mid (i,l) \in \mathcal{E}} \exp(-d(i,l))}.$$

- **Self-Attention Mechanism** in Transformer: dot product of linear transformed embeddings, using ***key*** and ***query*** matrices, **K** and **Q**:

transformed dot product

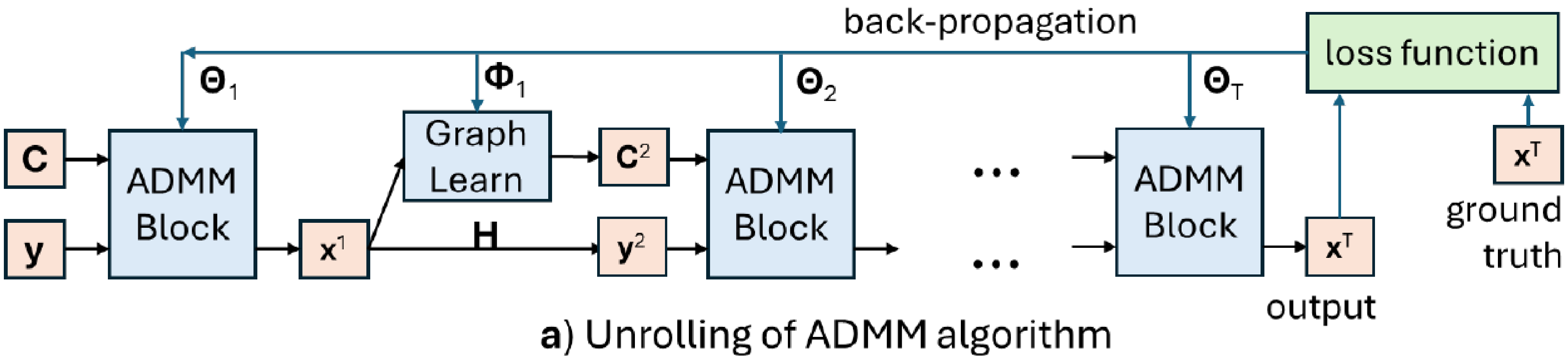
1. Graph learning with normalization from data is a self-attention mechanism!

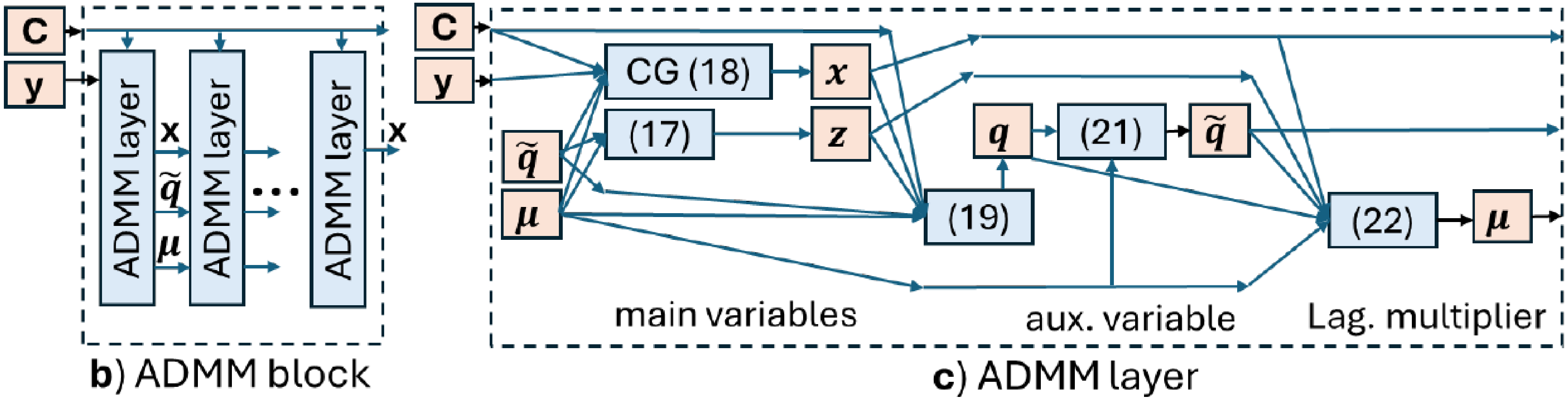
$$\mathbf{x}_j)^\top (\mathbf{K}\mathbf{x}_i).$$

$$\sum_{l=1} \exp(e(i,l))$$

[1] Do, Tam Thuc, et al. "**Interpretable Lightweight Transformer via Unrolling of Learned Graph Smoothness Priors**." accepted to *NeurIPS'24*.

Unrolling GTV-based ADMM Alg. for Image Interpolation





Gene Cheung (genec@yorku.ca)

Experiments: Unrolled GLR/GTV for Demosaicking

Method	Params#	McM		Kodak		Urban100	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Bilinear	-	29.71	0.9304	28.22	0.8898	24.18	0.8727
RST-B [46]	931763	34.85	0.9543	38.75	0.9857	32.82	0.973
RST-S [46]	3162211	35.84	0.961	**39.81**	**0.9876**	33.87	0.9776
Menon [47]	-	32.68	0.9305	38.00	0.9819	31.87	0.966
Malvar [48]	-	32.79	0.9357	34.17	0.9684	29.00	0.9482
iGLR	-	29.39	0.8954	27.50	0.8487	23.13	0.8406
iGTV	-	30.43	0.8902	28.66	0.8422	24.91	0.8114
uGLR	323410	36.09	0.9650	37.88	0.9821	33.60	0.9772
uGTV	323435	**36.59**	**0.9665**	39.11	0.9855	**34.01**	**0.9792**

Table 2: Demosaicking performance for different models, trained on 10k sample dataset.

- **Demosaicking**: fill in missing color pixels.
- Compared with two variants of RSTCANet employing Swin Transformer [1].
- Models trained on subset of DIV2K with 10K of 64x64 patches and same number of epochs (30).
- uGTV *employed 10% parameters of **RSTCANet** w/ comparable demosaicking performance*.

[1] Wenzhu Xing and Karen Egiazarian, "**Residual swin transformer channel attention network for image demosaicing**," *10th EUVIP. IEEE*, 2022, pp. 1–6.