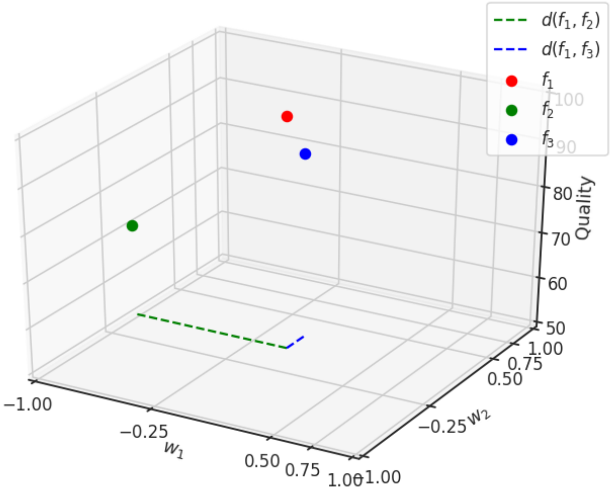
Projection to latent space

MIPT

2024

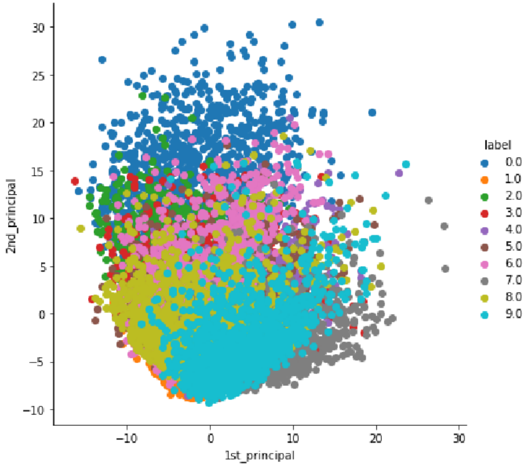
Motivation

Which model is closer to f_1?



Principal compnent analysis

$$\boldsymbol{W} = \arg\max Var(\boldsymbol{X}\boldsymbol{W})$$

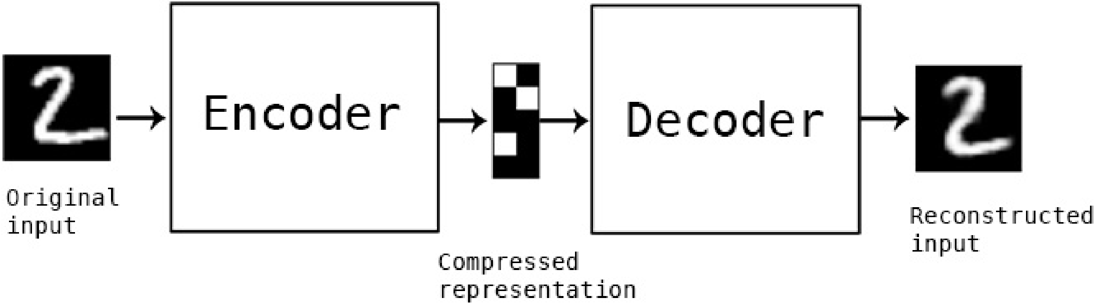


Autoencoder

Autoencoder is a model of dimension reduction:

$$\mathbf{H} = \sigma(\mathbf{W}_e \mathbf{X}),$$

$$||\sigma(\mathbf{W}_d \mathbf{H}) - \mathbf{X}||_2^2 \to \min .$$



Manifold

Manifold is space that can be locally approximated by Euclidian space.

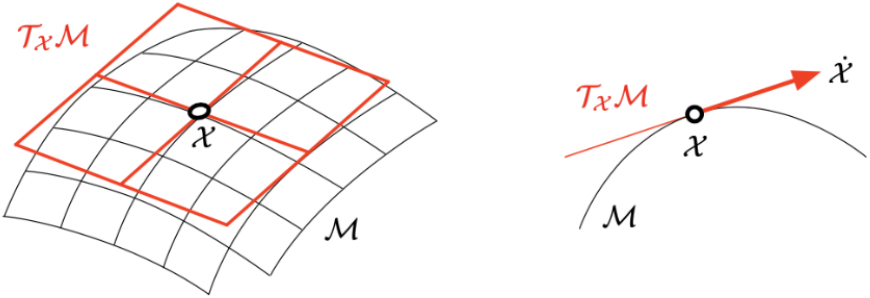
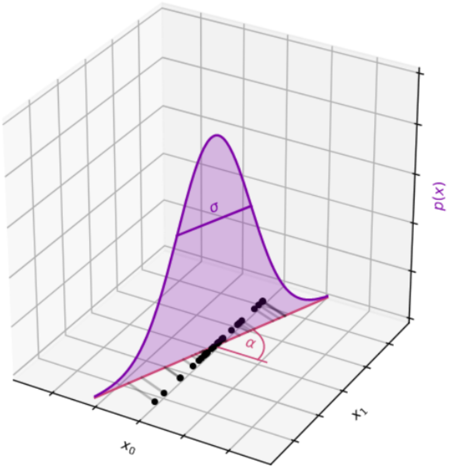
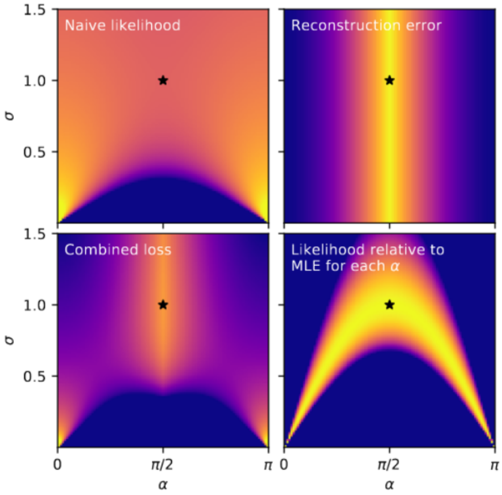


Figure 2. A manifold $\mathcal{M}$ and the vector space $T_\mathcal{X}\mathcal{M}$ (in this case $\cong \mathbb{R}^2$) tangent at the point $\mathcal{X}$, and a convenient side-cut. The velocity element, $\dot{\mathcal{X}} = \partial\mathcal{X}/\partial t$, does not belong to the manifold $\mathcal{M}$ but to the tangent space $T_\mathcal{X}\mathcal{M}$.

Manifold: do we need it?





Autoencoder: generative model?

(Alain, Bengio 2012): consider regularized autoencoder:

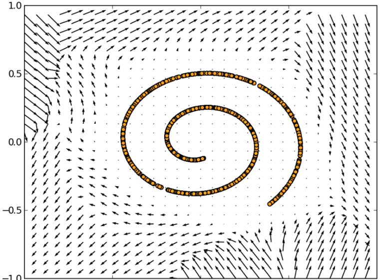
$$||\mathbf{f}(\mathbf{x}, \sigma) - \mathbf{x}||^2,$$

where σ is a noise level.

Then

$$\frac{\partial \log p(x)}{\partial x} = \frac{||\mathbf{f}(\mathbf{x}, \sigma) - \mathbf{x}||^2}{\sigma^2} + o(1) \text{ with } \sigma \to 0.$$

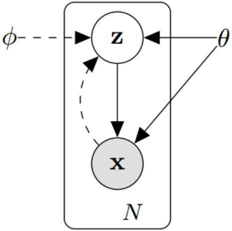
Vector field induced by reconstruction error



Variational autoencoder

Let the objects $\mathbf{X}$ be generated by latent variable $\mathbf{h} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$:

$$\mathbf{x} \sim p(\mathbf{x}|\mathbf{h}, \mathbf{w}).$$



$p(\mathbf{h}|\mathbf{x}, \mathbf{w})$ is unknown.
Maximize ELBO:

$$\log p(\mathbf{x}|\mathbf{w}) \geq \mathsf{E}_{q_\phi(\mathbf{h}|\mathbf{x})} \log p(\mathbf{x}|\mathbf{h}, \mathbf{w}) - D_{\mathrm{KL}}(q_\phi(\mathbf{h}|\mathbf{x}) || p(\mathbf{h})) \to \max .$$

Distributions $q_\phi(\mathbf{h}|\mathbf{x})$ and $p(\mathbf{x}|\mathbf{h}, \mathbf{w})$ are modeled by neural networks:

$$q_\phi(\mathbf{h}|\mathbf{x}) \sim \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}^2_\phi(\mathbf{x})),$$

$$p(\mathbf{x}|\mathbf{h}, \mathbf{w}) \sim \mathcal{N}(\boldsymbol{\mu}_w(\mathbf{h}), \boldsymbol{\sigma}^2_w(\mathbf{h})),$$

where $\boldsymbol{\mu}$,$\boldsymbol{\sigma}$ are neural network's outputs.

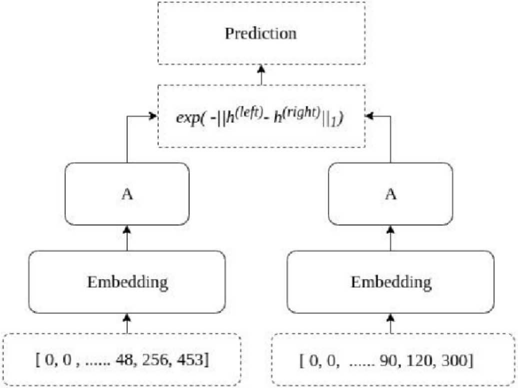
Multiple spaces

Given two spaces: $\mathbf{X}, \mathbf{Y}$.
Ho we can build a shared latent space between them?

Multiple spaces

Given two spaces: $\mathbf{X}, \mathbf{Y}$.
Ho we can build a shared latent space between them?
Naive method: $||\mathbf{f}(\mathbf{x}) - \mathbf{g}(\mathbf{y})||_2^2 \to \min$ does not work.

Siamese networks



Metric learning

$$D(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sqrt{(\boldsymbol{x}_1 - \boldsymbol{x}_2)^\top \boldsymbol{M} (\boldsymbol{x}_1 - \boldsymbol{x}_2)}$$

Triplet loss

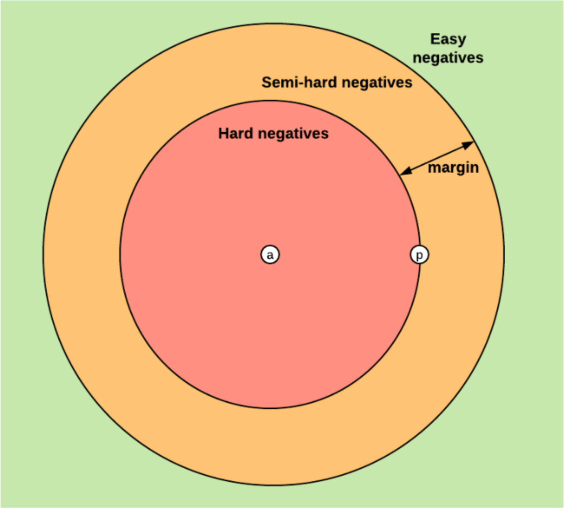
The loss function for each sample in the mini-batch is:

$$L(a, p, n) = \max\{d(a_i, p_i) - d(a_i, n_i) + \mathrm{margin}, 0\}$$

where

$$d(x_i, y_i) = \|\mathbf{x}_i - \mathbf{y}_i\|_p$$

Triplet loss



Bayesian representation learning with oracle constraints

$$p(t_{i,j,l}) = \int_z p(t_{i,j,l}|z_i, z_j, z_l)p(\boldsymbol{z}_i)p(\boldsymbol{z}_j)p(\boldsymbol{z}_k)dz_i dz_j dz_k,$$

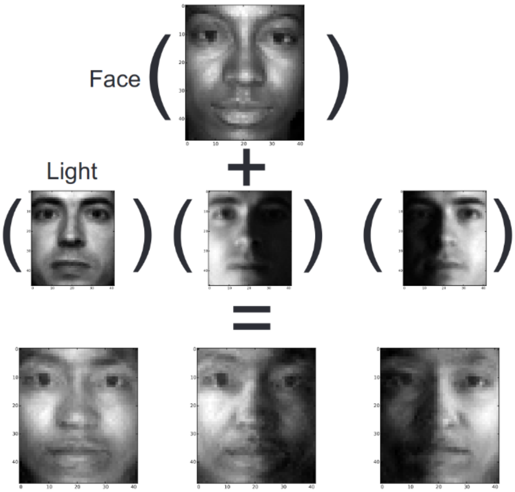
this gives the following likelihood:

$$p(t_{i,j,l}) = Ber(t_{i,j,l}) = \frac{\mathrm{e}^{-D_{i,j}}}{\mathrm{e}^{-D_{i,j}} + \mathrm{e}^{-D_{i,l}}}$$

with

$$D_{a,b} = \sum_{h=1}^{H} D_{a,b}^{h} = -\sum_{h=1}^{H} \left[\mathrm{JS}\Big(p(\boldsymbol{z}_a^h)||p(\boldsymbol{z}_b^h)\Big)\right].$$

Bayesian representation learning with oracle constraints



Variational learning across domains with triplet information

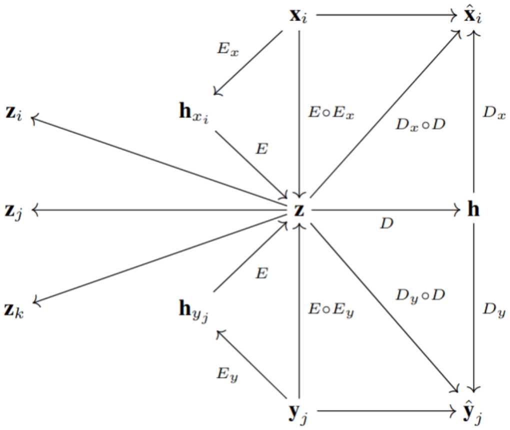


Figure 1: VBTA generative process

Variational learning across domains with triplet information

$$\mathcal{L}_{VBTA} = \mathbb{E}_{q_{\phi_x}(\mathbf{z}_x|\mathbf{x})} \log \frac{p_{\theta_x}(\mathbf{x}, \mathbf{y}, \mathbf{t}, \mathbf{z}_x)}{q_{\phi_x}(\mathbf{z}_x|\mathbf{x})} + \mathbb{E}_{q_{\phi_y}(\mathbf{z}_y|\mathbf{y})} \log \frac{p_{\theta_y}(\mathbf{x}, \mathbf{y}, \mathbf{t}, \mathbf{z}_y)}{q_{\phi_y}(\mathbf{z}_y|\mathbf{y})} =$$

$$= \underbrace{-\Big[KL\big(q_{\phi_{\mathbf{x}}(\mathbf{z}_x|\mathbf{x})}(\mathbf{z}_x|\mathbf{x}) \parallel p_{\theta_{\mathbf{x}}}(\mathbf{z}_x)\big) + KL\big(q_{\phi_{\mathbf{y}}(\mathbf{z}_y|\mathbf{y})}(\mathbf{z}_y|\mathbf{y}) \parallel p_{\theta_{\mathbf{y}}}(\mathbf{z}_y)\big)\Big]}_{\text{Penalty}} +$$

$$+ \underbrace{\Big[\mathbb{E}_{q_{\phi_{\mathbf{x}}}(\mathbf{z}_x|\mathbf{x})}\big[\log p_{\theta_{\mathbf{x}}}(\mathbf{x}|\mathbf{z}_x)\big] + \mathbb{E}_{q_{\phi_{\mathbf{y}}}(\mathbf{z}_y|\mathbf{y})}\big[\log p_{\theta_{\mathbf{y}}}(\mathbf{y}|\mathbf{z}_y)\big]\Big]}_{\text{Reconstruction}} +$$

$$+ \underbrace{\Big[\mathbb{E}_{q_{\phi_{\mathbf{x}}}(\mathbf{z}_x|\mathbf{x})}\big[\log p_{\theta_{\mathbf{x}}}(\mathbf{y}|\mathbf{z}_x)\big] + \mathbb{E}_{q_{\phi_{\mathbf{y}}}(\mathbf{z}_y|\mathbf{y})}\big[\log p_{\theta_{\mathbf{y}}}(\mathbf{x}|\mathbf{z}_y)\big]\Big]}_{\text{Cycle-consistency}} +$$

$$+ \underbrace{\mathbb{E}_{q_{\phi_{\mathbf{x}}}(\mathbf{z}_x|\mathbf{x})}\big[\log p(\mathbf{t}|\mathbf{z}_x)\big] + \mathbb{E}_{q_{\phi_{\mathbf{y}}}(\mathbf{z}_y|\mathbf{x})}\big[\log p(\mathbf{t}|\mathbf{z}_y)\big]}_{\text{Triplet likelihood}}$$

How we can embed models into (probabilistic) vector space?

What do we want from these embeddings?

Differentiable Neural Architecture Search in Equivalent Space with Exploration Enhancement

- Structure representation: graph supervised encoder
- Structure optimization: DARTS + exploration

Table 1: Comparison results with state-of-the-art NAS approaches on NAS-Bench-201.

Method	CIFAR-10		CIFAR-100		ImageNet-16-120	
	Valid(%)	Test(%)	Valid(%)	Test(%)	Valid(%)	Test(%)
ENAS	37.51±3.19	53.89±0.58	13.37±2.35	13.96±2.33	15.06±1.95	14.84±2.10
RandomNAS*	85.63±0.44	88.58±0.21	60.99±2.79	61.45±2.24	31.63±2.15	31.37±2.51
DARTS (1st)	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
DARTS (2nd)	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
SETN	84.04±0.28	87.64±0.00	58.86±0.06	59.05±0.24	33.06±0.02	32.52±0.21
NAO*	82.04±0.21	85.74±0.31	56.36±3.14	59.64±2.24	30.14±2.02	31.35±2.21
GDAS*	90.03±0.13	93.37±0.42	70.79±0.83	70.35±0.80	40.90±0.33	41.11±0.13
E^2NAS	**90.94±0.83**	**93.89±0.47**	**71.83±1.84**	**72.05±1.58**	**45.44±1.24**	**45.77±1.00**

Does Unsupervised Architecture Representation Learning Help Neural Architecture Search?

- Structure representation: graph VAE
- Optimization: unsupervised for encoding models, then RL+BO

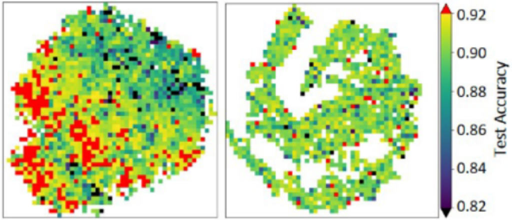
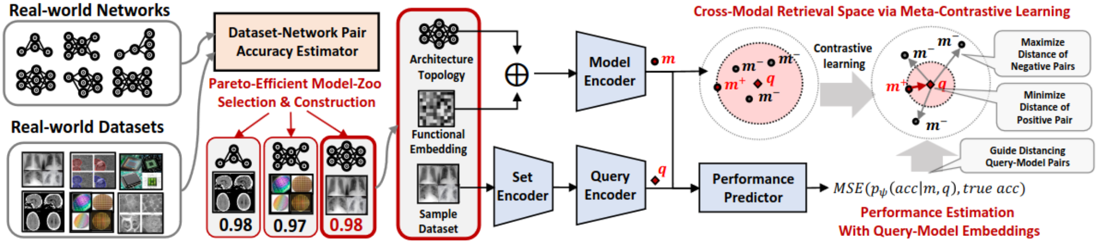


Figure 4: Latent space 2D visualization [65] comparison between *arch2vec* (left) and supervised architecture representation learning (right) on NAS-Bench-101. Color encodes test accuracy. We randomly sample 10,000 points and average the accuracy in each small area.

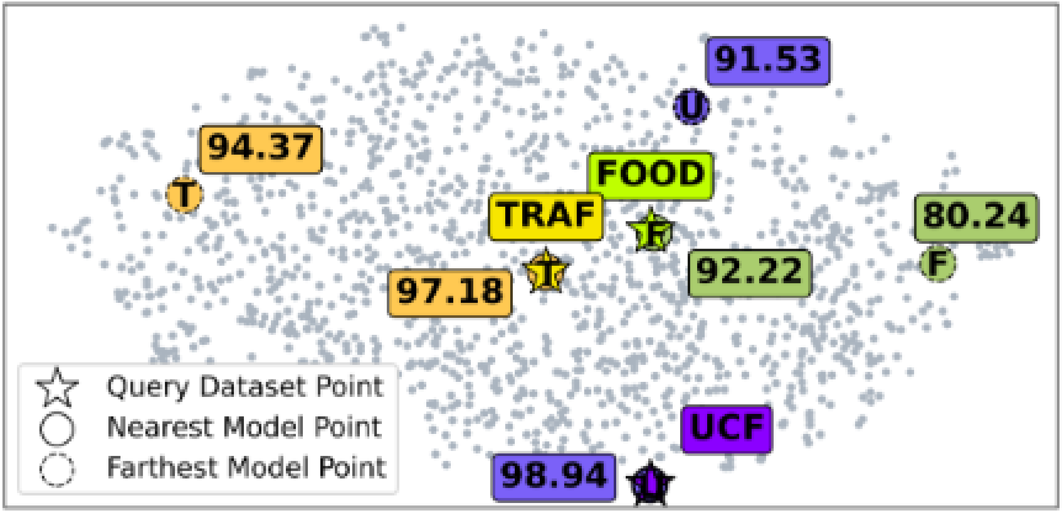
Task-Adaptive Neural Network Search with Meta-Contrastive Learning



Task-Adaptive Neural Network Search with Meta-Contrastive Learning

Target Dataset	Method	# Epochs	FLOPs (M)	Params (M)	Search Time (GPU sec)	Training Time (GPU sec)	Speed Up	Accuracy (%)
Averaged Performance	MobileNetV3 [26]	50	132.94	4.00	-	257.78±09.77	1.00×	94.20±0.70
	PC-DARTS [65]	500	566.55	**3.54**	1100.37±22.20	5721.13±793.71	0.04×	79.22±1.69
	DrNAS [10]	500	623.43	4.12	1501.75±43.92	5659.77±403.62	0.04×	84.06±0.97
	FBNet-A [60]	50	246.69	4.3	-	293.42±57.45	0.88×	93.00±1.95
	OFA [8]	50	148.76	6.74	121.90±0.00	226.58±03.13	0.74×	93.89±0.84
	MetaD2A [31]	50	512.67	6.56	2.59±0.13	345.39±28.36	0.74×	95.24±1.14
	TANS (Ours)	10	181.74	5.51	0.002±0.00	40.19±03.06	-	95.17±2.20
	TANS (Ours)	50	181.74	5.51	**0.002±0.00**	**200.93±11.01**	**1.28×**	**96.28±0.30**
Colorectal Histology Dataset (Easy)	MobileNetV3 [26]	50	132.94	**4.00**	-	577.18±04.15	1.00×	96.23±0.07
	PC-DARTS [65]	500	534.64	4.02	2062.42±49.14	12124.18±1051.16	0.04×	96.17±0.68
	DrNAS [10]	500	614.23	4.12	4183.20±188.60	11355.18±1352.62	0.04×	97.51±0.13
	FBNet-A [60]	50	215.45	4.3	-	696.00±295.19	0.83×	95.43±0.57
	OFA [8]	50	134.85	6.74	121.90±0.00	537.61±03.52	0.88×	96.40±0.52
	MetaD2A [31]	50	506.88	5.93	2.58±0.12	784.45±79.32	0.73×	96.57±0.56
	TANS (Ours)	10	171.74	4.95	0.001±0.00	98.56±04.24	-	96.87±0.21
	TANS (Ours)	50	171.74	4.95	**0.001±0.00**	**492.81±21.19**	**1.17×**	**97.67±0.05**
Food Classification Dataset (Hard)	MobileNetV3 [26]	50	132.94	4.00	-	235.57±07.57	1.00×	87.52±0.78
	PC-DARTS [65]	500	567.85	**3.62**	1018.49±6.31	6323.40±938.83	0.03×	55.42±2.46
	DrNAS [10]	500	632.67	4.12	1276.38±0.00	5079.89±161.05	0.04×	61.45±0.68
	FBNet-A [60]	50	251.29	4.3	-	251.24±3.31	0.94×	84.33±1.41
	OFA [8]	50	152.34	6.74	121.90±0.00	**190.86±03.48**	0.75×	87.43±0.59
	MetaD2A [31]	50	521.11	8.23	2.60±0.23	324.62±34.97	0.72×	89.72±1.53
	TANS (Ours)	10	179.83	5.07	0.002±0.00	40.59±04.84	-	93.11±0.24
	TANS (Ours)	50	179.83	5.07	**0.002±0.00**	202.93±24.21	**1.16×**	**93.71±0.24**

Task-Adaptive Neural Network Search with Meta-Contrastive Learning



Recap: PCA (1 component)

Given a dataset $\mathbf{X}$. We want to find a weight vector $\mathbf{p}, ||\mathbf{p}||^2 = 1$ that the linear combination

$$\sum_{\mathbf{x}} \sum_{j=1}^{d} p_j \mathbf{x}_j \to \max$$

Can we repeat the same technique for the functions?

PCA for functional spaces

Given a set of functions $\mathbf{X}(s)$ in a Banach space. We want to find a weight vector $\mathbf{p}, ||\mathbf{p}||^2 = 1$ that the linear combination

$$\sum_{\mathbf{x}} \sum_{j=1}^{d} \int (\mathbf{p}(s)\mathbf{x}(s)ds)^2 \to \max$$

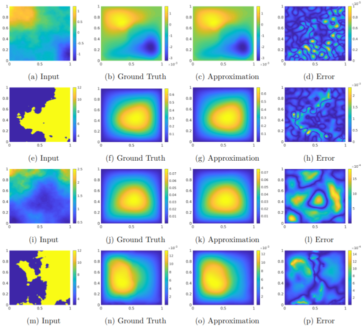
- How the $\mathbf{X}(s)$ and $\mathbf{X}$ are connected?
- What are «good» functions for use this method?

Operator learning

Example 1 We want to approximate a differential operator $\mathcal{P}_x$.

$$(\mathcal{P}_x y)(s) = 0 \forall s \in D,$$

where x and y are **functions** from Banach space, and $D \subset \mathbb{R}^d$.

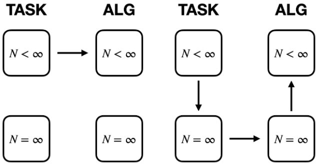


(a) Input (b) Ground Truth (c) Approximation (d) Error
(e) Input (f) Ground Truth (g) Approximation (h) Error
(i) Input (j) Ground Truth (k) Approximation (l) Error
(m) Input (n) Ground Truth (o) Approximation (p) Error

Operator learning

(A) Same images at different resolutions

(B) Different resolutions as vectors and (bottom right) as a function



(A) Directly design algorithm at fixed resolution N

(B) Design algorithm at limit of infinite resolution

Example 2

PCA-net

1. Learn PCA for input (the input functions are discretized, so we can use a simple Euclidean basis for learning PCA)
2. Learn PCA for output
3. Learn NN to transform input to output

$$
\begin{tikzcd}
\mathcal{U} \arrow[r, "F_{\mathcal{U}}"] \arrow[d, "\Psi^{\dagger}"'] & \mathbb{R}^{d_{\mathcal{U}}} \arrow[r, "G_{\mathcal{U}}"] \arrow[d, "\varphi"'] & \mathcal{U} \arrow[d, "\Psi^{\dagger}"'] \\
\mathcal{V} \arrow[r, "F_{\mathcal{V}}"] & \mathbb{R}^{d_{\mathcal{V}}} \arrow[r, "G_{\mathcal{V}}"] & \mathcal{V}
\end{tikzcd}
$$

References

- Brehmer, Johann, and Kyle Cranmer. "Flows for simultaneous manifold learning and density estimation."Advances in Neural Information Processing Systems 33 (2020): 442-453.
- Bishop C. M. Pattern recognition //Machine learning. – 2006. – T. 128. – No. 9.
- Bishop C. Bayesian pca //Advances in neural information processing systems. – 1998. – T. 11.
- Sola J., Deray J., Atchuthan D. A micro Lie theory for state estimation in robotics //arXiv preprint arXiv:1812.01537. – 2018.
- Alain G., Bengio Y. What regularized auto-encoders learn from the data-generating distribution //The Journal of Machine Learning Research. – 2014. – T. 15. – No. 1. – C. 3563-3593.
- Kingma D. P., Welling M. Auto-encoding variational bayes //arXiv preprint arXiv:1312.6114. – 2013.
- Ranasinghe T., Orˇasan C., Mitkov R. Semantic textual similarity with siamese neural networks //Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019). – 2019. – C. 1004-1011.
- https://russianblogs.com/article/5172713037/
- Karaletsos T., Belongie S., Ratsch G. Bayesian representation learning with oracle constraints //arXiv preprint arXiv:1506.05011. – 2015.
- Kuznetsova R., Bakhteev O., Ogaltsov A. Variational learning across domains with triplet information //arXiv preprint arXiv:1806.08672. – 2018.
- Zhang M. et al. Differentiable neural architecture search in equivalent space with exploration enhancement //Advances in Neural Information Processing Systems. – 2020. – T. 33. – C. 13341-13351.
- Luo R. et al. Neural architecture optimization //Advances in neural information processing systems. – 2018. – T. 31.
- Yan S. et al. Does unsupervised architecture representation learning help neural architecture search? //Advances in Neural InformationProcessing Systems. – 2020. – T. 33. – C. 12486-12498
- Jeong, Wonyong, et al. "Task-adaptive neural network search with meta-contrastive learning."Advances in Neural Information Processing Systems 34 (2021): 21310-21324.
- Ramsay, James O., and James B. Ramsey. "Functional data analysis of the dynamics of the monthly index of nondurable goods production."Journal of econometrics 107.1-2 (2002): 327-344.
- Kovachki, Nikola B., Samuel Lanthaler, and Andrew M. Stuart. "Operator Learning: Algorithms and Analysis."arXiv preprint arXiv:2402.15715 (2024).