D. Representation Learning

D.1 Representation Learning: Learning useful coordinates for data

D.2 Representation Learning: Evaluating and probing Representations

D.3 Representation Learning: How to learn a good Representation

Recap: What makes a representation good?

A good representation makes a downstream task easier. (Goodfellow et al. 2016)

Broadly speaking, it is:

1. **Sufficient**: keeps task-relevant information

2. **Compact**: removes irrelevant variation

3. **Structured**: similar examples are nearby

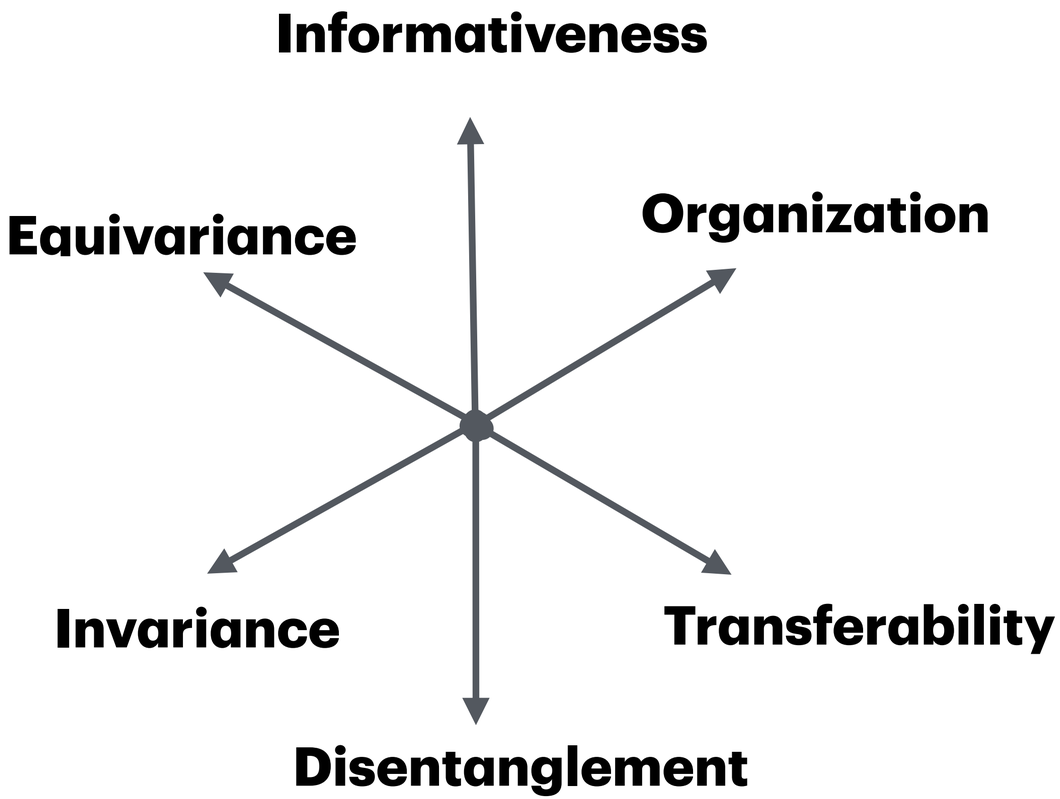
4. **Separated**: task-relevant groups are easier to distinguish

5. **Robust**: ignores irrelevant perturbations, but preserves meaningful differences

Look at:
Eastwood, Cian, and Christopher KI Williams. "A framework for the quantitative evaluation of disentangled representations." *International conference on learning representations*. 2018.

Representation Quality Has Multiple Axes (and is ongoing area of research)

- A representation is not “good” by only one universal metric. It is good when it preserves, removes, tracks, and organizes the right information for the task.
- Some possible aspects that we should investigate in order to have a better representation could be:



Plachouras, Christos, et al. "Towards a unified representation evaluation framework beyond downstream tasks." *arXiv preprint arXiv:2505.06224* (2025).

Representation Quality Has Multiple Axes (and is ongoing area of research)

a-Informativeness: Does it keep useful information for downstream tasks?

b-Organization: Is the latent space organized well?

c-Disentanglement: Does it separate factors of variation?

d-Invariance: Does it ignore irrelevant variation?

e-Equivariance: Does it track meaningful transformations?

f-Transferability: Does it generalize to new tasks or domains?

- We will explain and expand each of these in this lecture. We will also see how we can implement these investigation in a toy example with a toy model.

Plachouras, Christos, et al. "Towards a unified representation evaluation framework beyond downstream tasks." *arXiv preprint arXiv:2505.06224* (2025).

a-Informativeness: Does it keep useful information for downstream tasks?

- **Factor of variation (FV)**: is a property of the input that can change across examples:

$$FV(x) = \text{a measurable property of } x$$

Examples: class label, cell type, disease state, hue of image , brightness of image, rotation angle, batch, perturbation

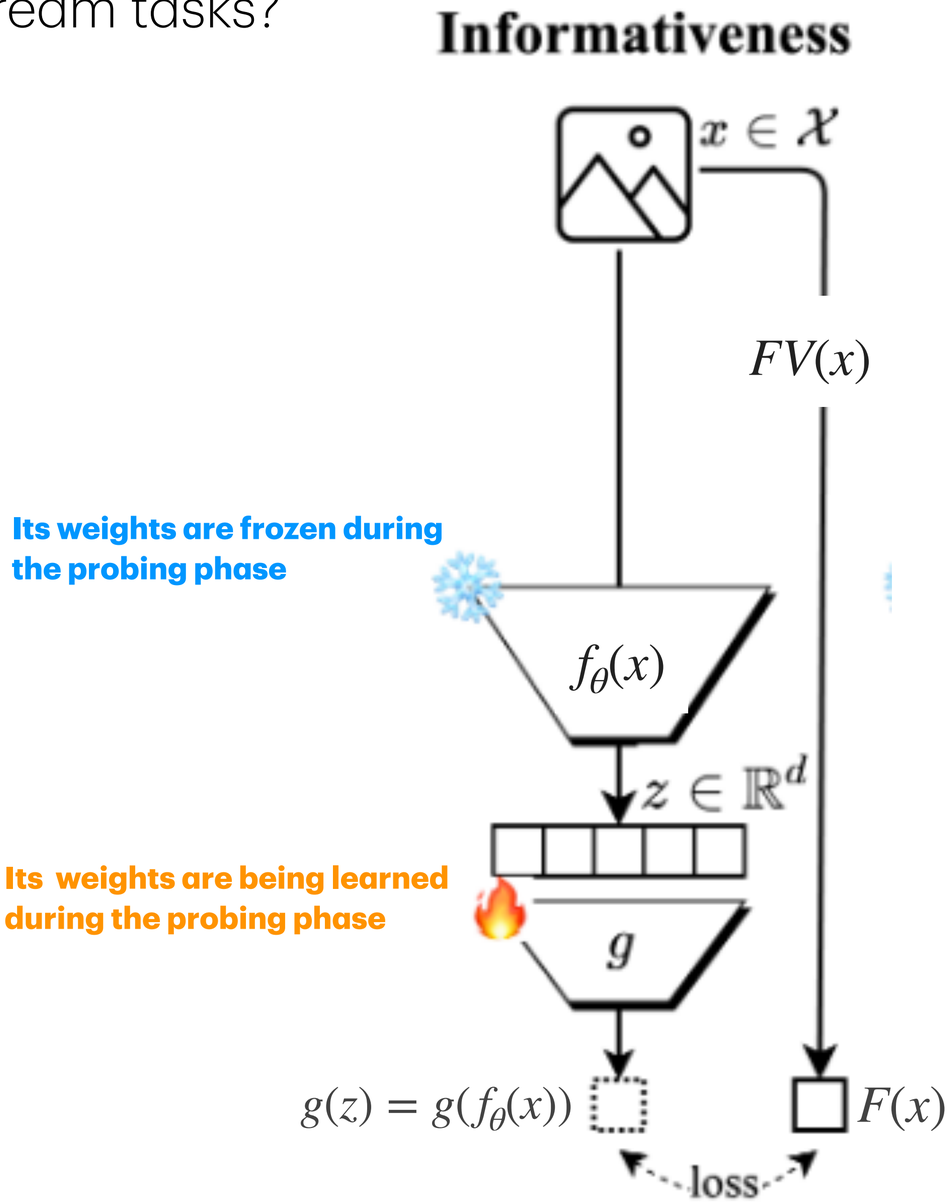
- **Informativeness** asks whether useful properties of the input can be recovered from the representation. We have an already trained encoder function f_θ, it does:

$$z = f_\theta(x)$$

- To investigate informativeness, we train a small probe function (g):

$$g(z) = g(f_\theta(x))$$

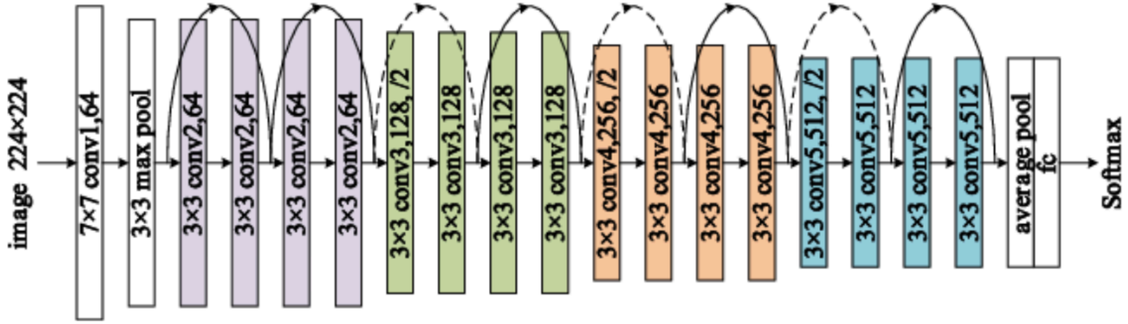
- and compare it to the true factor, $FV(x)$.
- If $g(z)$ predicts $FV(x)$ well, then z contains information about that factor.



a-Informativeness: Example setup

1. We load 2 pretrained visual encoders: **ResNet18** and **ResNet34**
   These models were already trained on ImageNet-1K for supervised image classification.

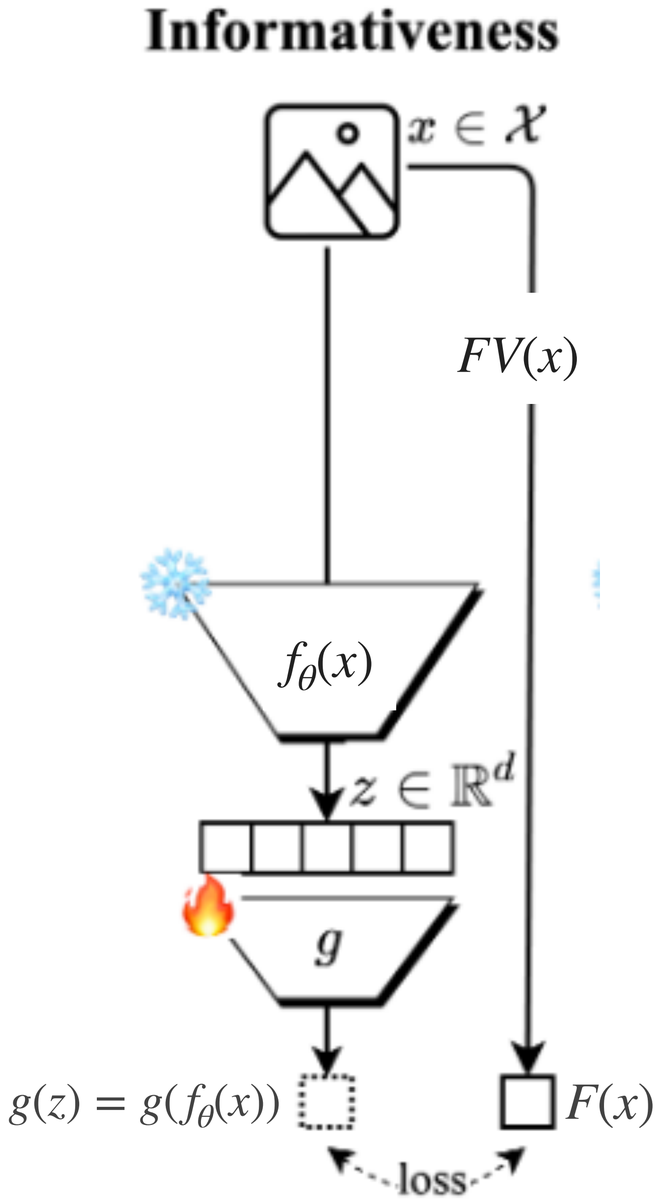
   We remove the final classifier and keep the encoder: $x \rightarrow z = f_\theta(x)$
   We will freeze the weights of the Encoder.



3. We apply the frozen encoder to a new dataset: STL-10

   We do not fine-tune the encoder. We train simple probes on top of z to ask: "What information is accessible from the representation?"

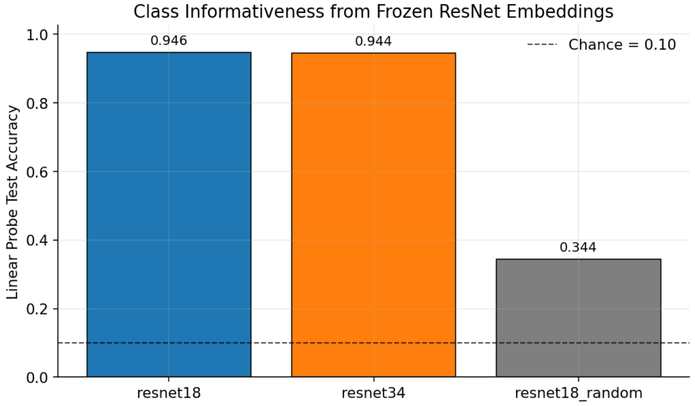
4. Factors of variation (FVs) we choose: Class label, Original image brightness, Original image contrast.

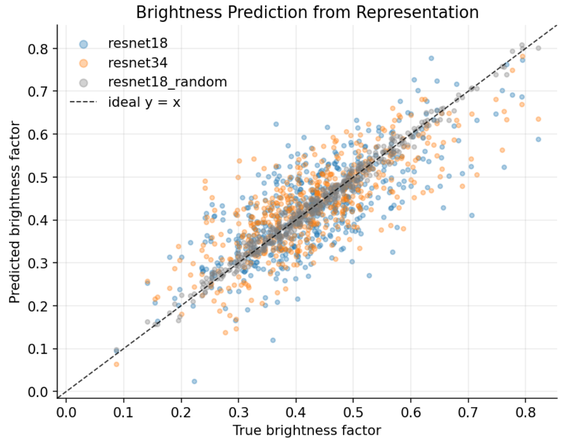


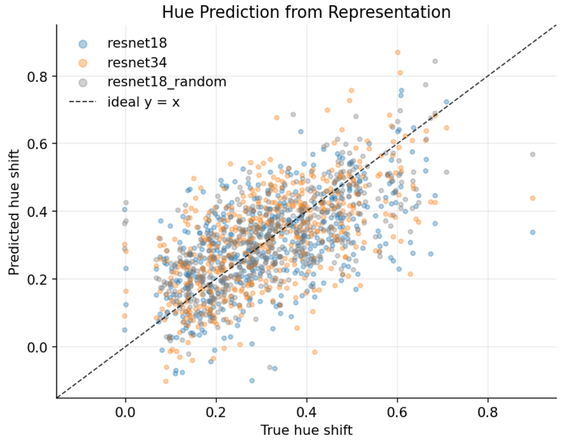
Plachouras, Christos, et al. "Towards a unified representation evaluation framework beyond downstream tasks." *arXiv preprint arXiv:2505.06224* (2025).

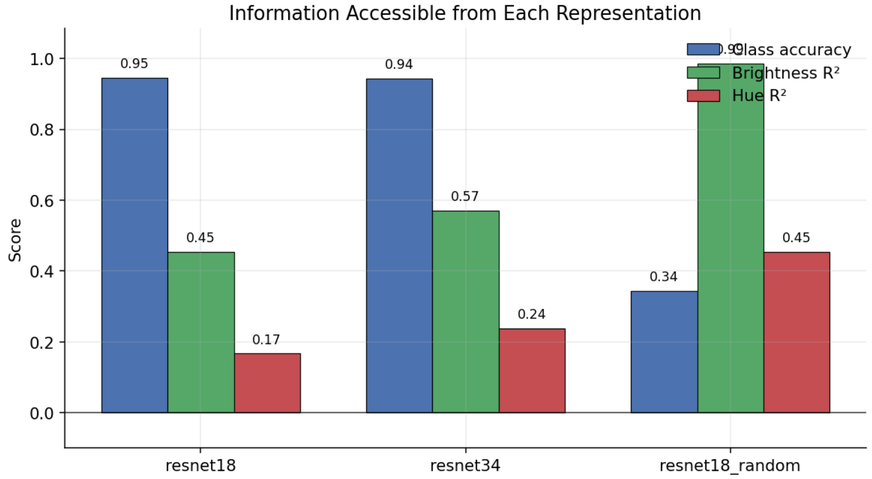
a-Informativeness: Example setup

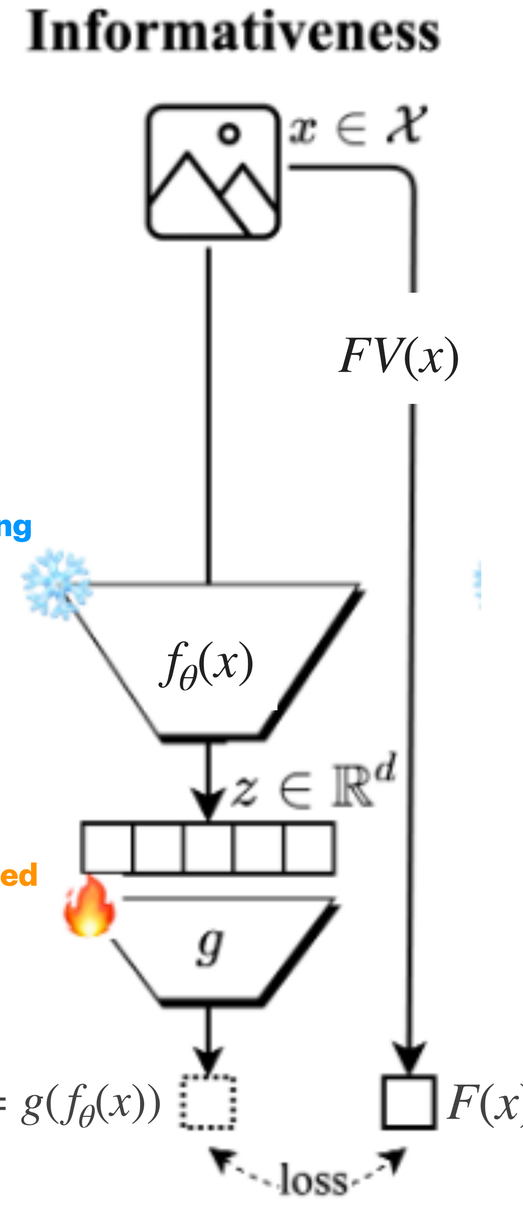
Here, the factors of variation are: class labels ,brightness of image , hue of image











Its weights are frozen during the probing phase

Its weights are being learned during the probing phase

b-Organization: Is the latent space organized well?

- A representation is organized if distances and neighborhoods in latent space are meaningful:

  For each input point x_i we have a corresponding point in the embedding space :

  $$z_i = f_\theta(x_i).$$

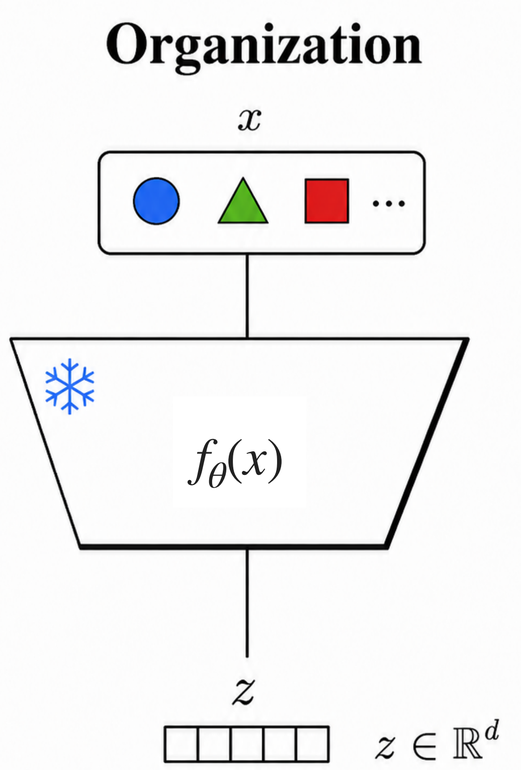
  We compare points in representation space:

  $$d(z_i, z_j)$$

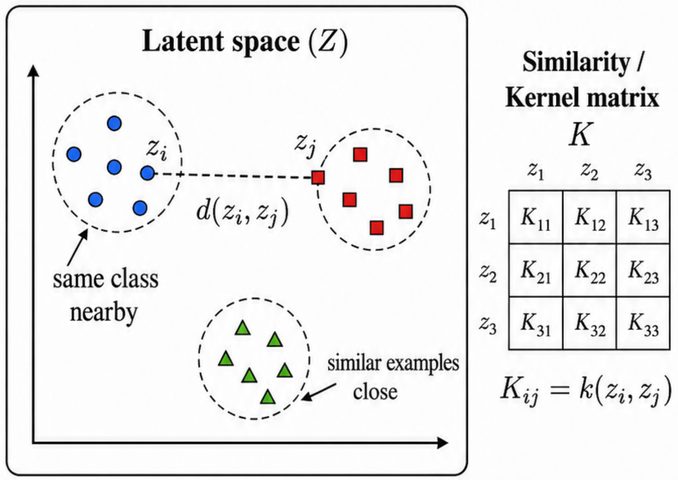
  and ask whether nearby points share useful structure.

  Examples: point from same class of image, same cell type, similar phenotype, nearby pseudotime, similar perturbation response, etc.

- Evaluation Methods include:
  b1-kNN / retrieval
  b2-clustering quality
  b3-kernel or similarity matrix K
  b4-intrinsic dimensionality
  b5-PCA / UMAP (for visualization)



Ideally



b-Organization: evaluation methods

b1.kNN / retrieval (local structure): For each point z_i, find its nearest neighbours in latent space: $neighbors(z_i) = \{z_j \mid z_j \text{ is close to } z_i\}$

A good representation should retrieve examples with similar meaning: $neighbors(z_i) \sim similar$

b2.Clustering quality (global structure): Cluster the embeddings and ask whether the clusters match meaningful groups and whether global groups emerge naturally in the representation space.
Some useful metrics to use in this analysis are:

- Adjusted Rand Index (ARI): Do pairs of points that belong together in the true labels also end up together in the clustering?
- Normalized Mutual Information (NMI): How much information do the clusters contain about the true labels?
- Silhouette score: Are points closer to their own cluster than to other clusters?

b3. Kernel / similarity matrix: Compute pairwise similarity between representations: $K_{ij} = k(z_i, z_j)$.

where k is a similarity measure, for example: $k(z_i, z_j) = z_i^\top z_j$ (dot product) or $k(z_i, z_j) = \dfrac{z_i^\top z_j}{\|z_i\| \|z_j\|}$ (cosine similarity)

This is a good measure to compare different representation learning models, one can compare the kernel of different models.

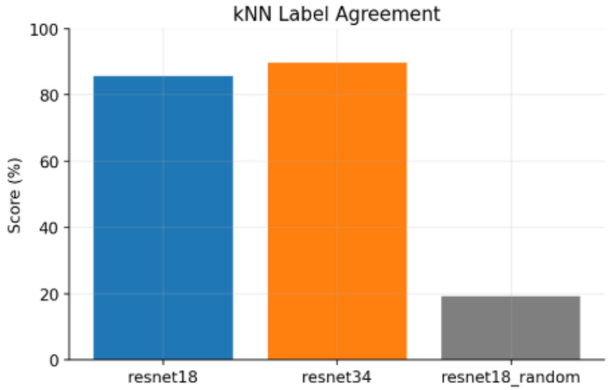
b4. Intrinsic dimensionality: Estimate how many effective dimensions the representation uses: $d_{\text{intrinsic}} \ll D_z$?

This asks whether z lies near a simpler lower-dimensional structure. This connects to the manifold hypothesis of previous lecture.

b5. PCA / UMAP visualization: Project Z into lower dimension to inspect global structure, clusters, gradients, or trajectories.
Note that visualization is only qualitative.

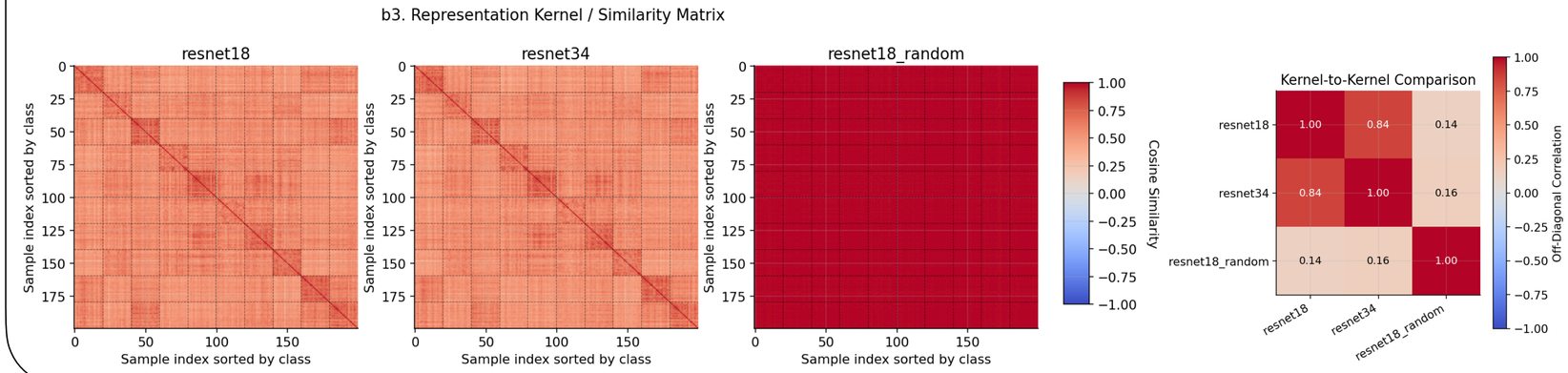
b-Organization: Example setup

b1.kNN / retrieval (local structure):

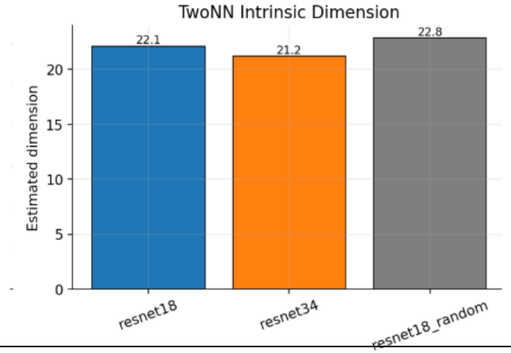


b2.Clustering quality (global structure):

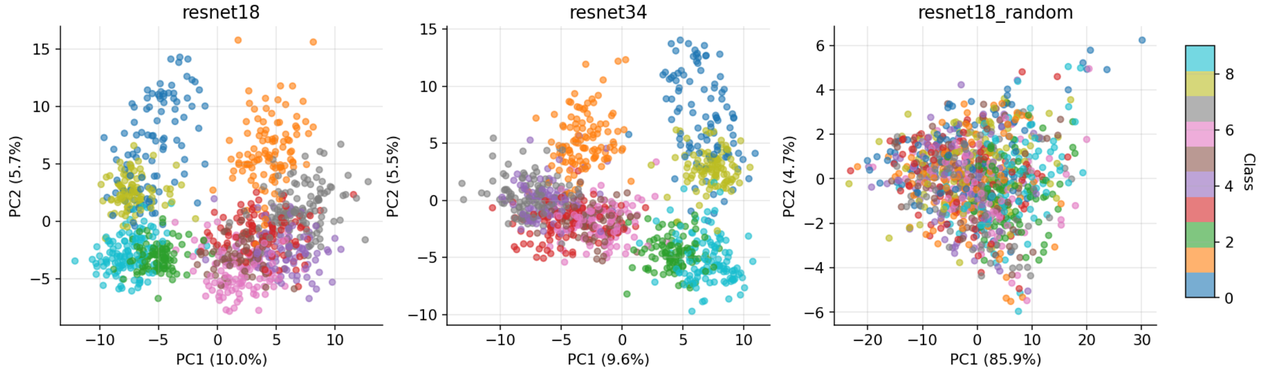
b3. Kernel / similarity matrix:



b4. Intrinsic dimensionality:



b5. PCA / UMAP visualization:



b-Organization: Some Fun Reads!

The Platonic Representation Hypothesis

Neural networks, trained with different objectives on different data and modalities, are converging to a shared statistical model of reality in their representation spaces.

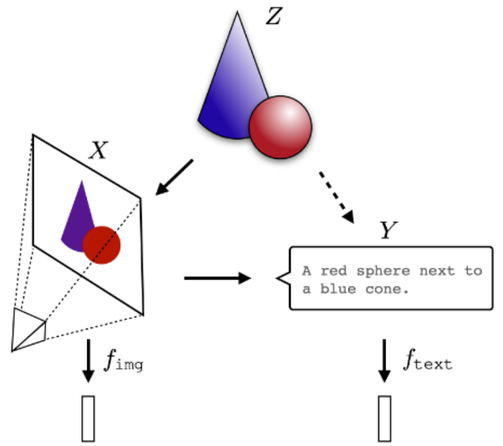
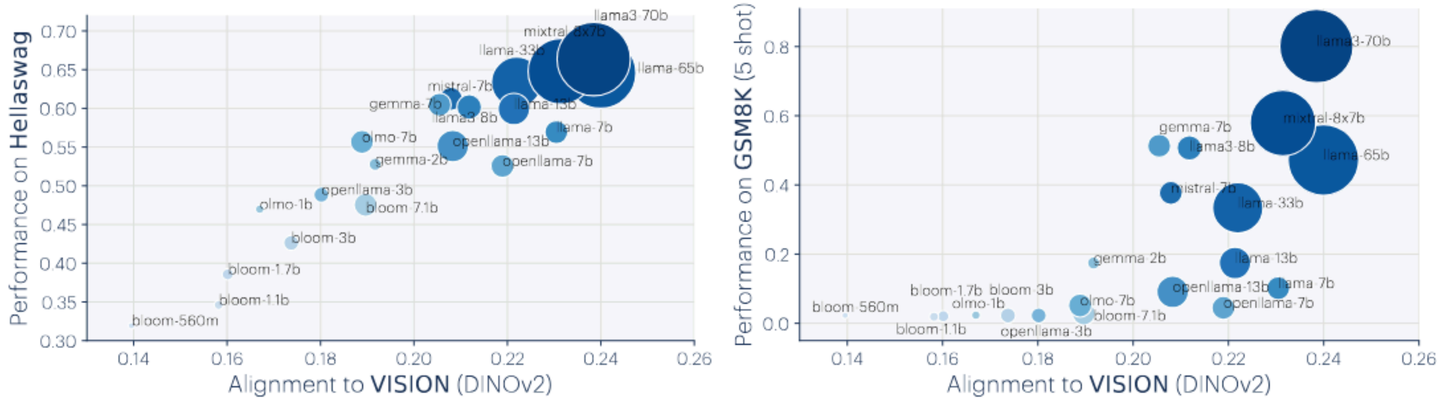


Figure 1. **The Platonic Representation Hypothesis:** Images (X) and text (Y) are projections of a common underlying reality (Z). We conjecture that representation learning algorithms will converge on a shared representation of Z, and scaling model size, as well as data and task diversity, drives this convergence.

The Platonic Representation Hypothesis

Minyoung Huh [*1] **Brian Cheung** [*1] **Tongzhou Wang** [*1] **Phillip Isola** [*1]

The Platonic Representation Hypothesis



Huh, Minyoung, et al. "The platonic representation hypothesis." *arXiv preprint arXiv:2405.07987* (2024).

Read this: https://arxiv.org/pdf/2512.03750

https://arxiv.org/pdf/2512.05717

www.nature.com/articles/s42256-026-01235-7

c-Disentanglement: Does the representation separate factors of variation?

A representation is disentangled if different meaningful factors are encoded separately.

For each input we have:

$$z = f_\theta(x)$$

assume we have known factors of variation for data x:

$$FV_1(x), FV_2(x), \ldots, FV_K(x)$$

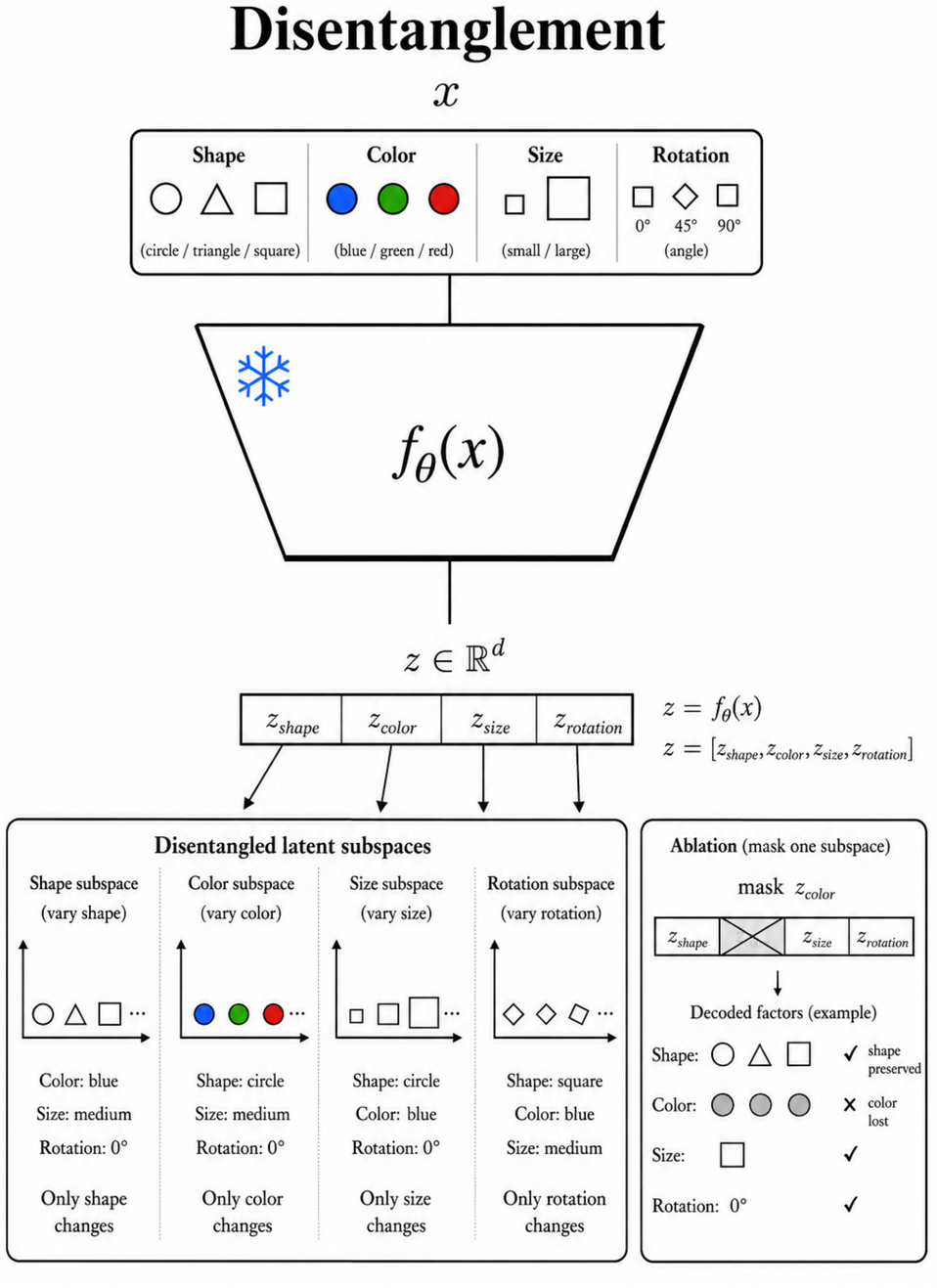
Disentanglement asks "whether different factors are encoded in separate dimensions, directions, or subspaces of z"

A useful way to think about it:

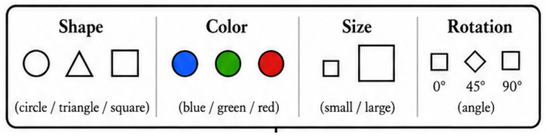
$A_{k,m}$ = how much latent component z_m contains information about FV_k

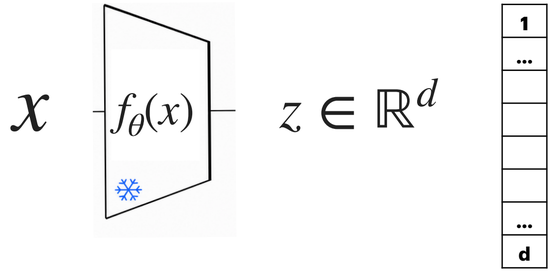
A disentangled representation should make this association matrix sparse and interpretable.

Ideally



c-Disentanglement: Evaluation methods





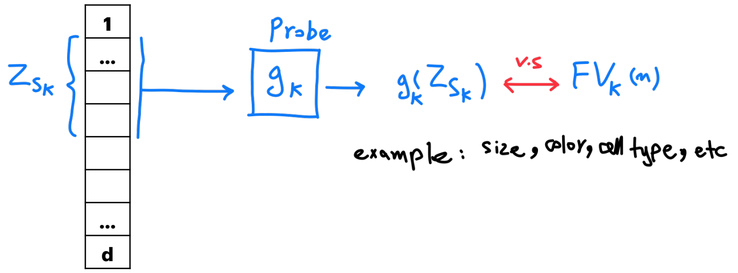
c1. Factor probes : •(i).Train one probe for each known factor of variation:

$$g_k(z) \approx FV_k(x).$$

And we ask Is factor FV_k recoverable from z?

•(ii)Next to test disentanglement, ask whether it is recoverable from a specific part of z: $g_k(z_{S_k}) \approx FV_k(x)$.

where z_{S_k} is a selected dimension, direction, or subspace.



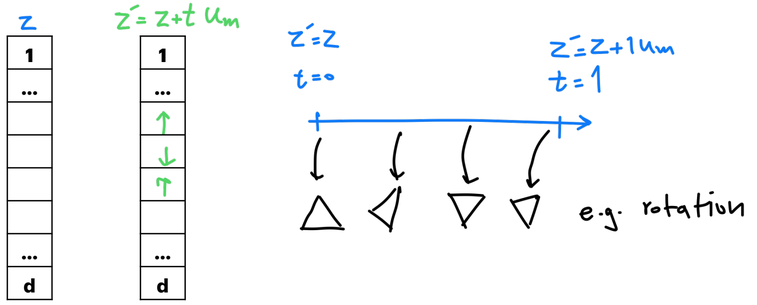
c2. Latent traversal: • Move along (with value of t) one latent direction (u_m):

$$z' = z + tu_m$$

• Then ask which factor changes. In a good disentanglement only one factor changes for specific directions:

FV_k changes for z', FV_j stays stable for $j \neq k$

This means: one latent direction → one main factor



c3. Ablation: • Remove one dimension, direction, or subspace (m) from z:

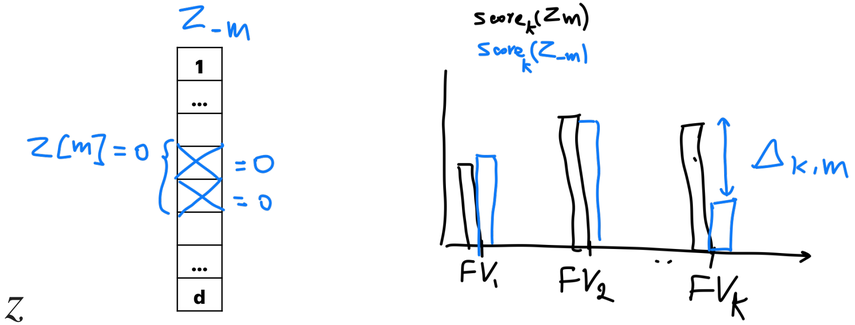
$$z \rightarrow z_{-m}$$

• Then measure which factor prediction drops:

$$\Delta_{k,m} = \text{Score}_k(z) - \text{Score}_k(z_{-m})$$

• Where: $\text{Score}_k(z)$ = performance of probe g_k in predicting $FV_k(x)$ from z

• In good disentanglement: $\Delta_{k,m}$ large for one factor and $\Delta_{j,m}$ small for other factors



c-Disentanglement: Example setup

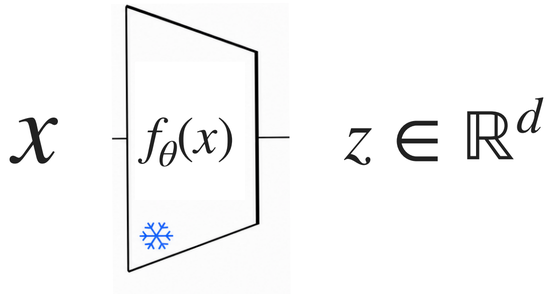
Dataset → **Model** → **Analysis**

dsprites-dataset

FVs:

- shapes (square, ellipse, heart)
- scales (6 scale)
- rotations (40 rotations)
- x-position (32 positions)
- y-positions (32 positions)

(737,280 pre-rendered images)



Model 2: train and freeze an Autoencoder
Model 3: train and freeze a β-VAE
Model 4: train and freeze β-TCVAE

c1. Factor probes :

(i) train probes: $g_k(z) \approx FV_k(x)$

Ask can $g_k(z)$ predict FV_k

(ii) on subspaces: $g_k(z_m) \approx FV_k(x)$

Make a heatmap of

latent dimension z_m ***vs.*** *factor FV_k*

c2. Latent traversal:

->Pick a direction in latent (u_m) then:

$$z'(t) = z + tu_m$$

->ask the decoder (if exists):

$$\hat{x}'(t) = D(z'(t))$$

->And see which FV_k are changing

c3. Ablation:

(i) Remove one dimension/subspace m:

$$z \to z_{-m}$$

(ii) Then measure which factor prediction drops:

$$A_{k,m} = \text{Score}_k(z) - \text{Score}_k(z_{-m})$$

(which latent parts are important which factor)

c-Disentanglement: c1. Factor probes :

c1. Factor probes :

(i) train probes: $g_k(z) \approx FV_k(x)$

Ask can $g_k(z)$ predict FV_k

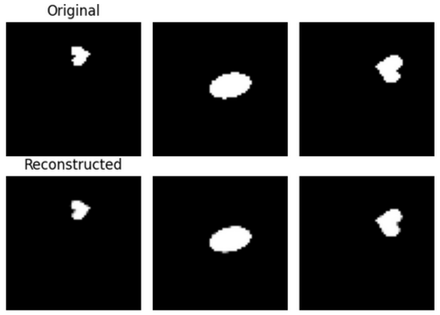
	factor	num_classes	chance_acc
0	shape	3	0.333333
1	scale	6	0.166667
2	rotation	40	0.025000
3	x_position	32	0.031250
4	y_position	32	0.031250

(ii) on subspaces: $g_k(z_m) \approx FV_k(x)$

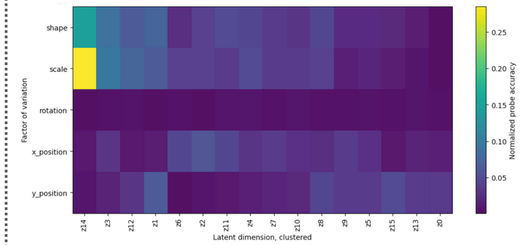
Make a heatmap of

latent dimension z_m **vs.** *factor FV_k*

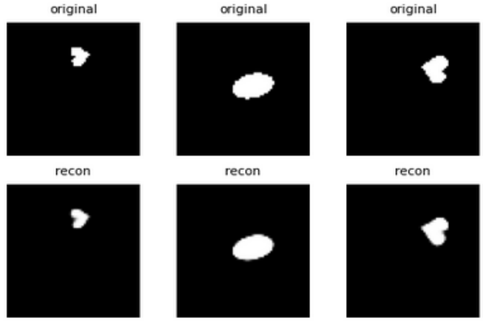
Autoencoder



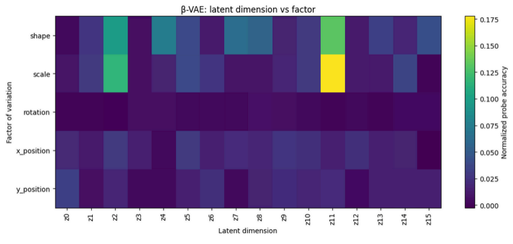
normalized_acc
0.163086
0.378206
0.034692
0.402484
0.405256



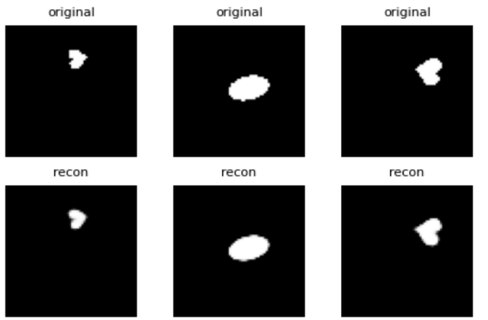
B-VAE



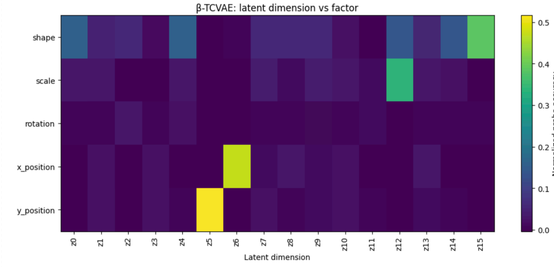
normalized_acc
0.152629
0.396875
0.022046
0.287018
0.277330



β-TCVAE



normalized_acc
0.174927
0.341911
0.019501
0.355791
0.287004



c-Disentanglement: c.2 Latent traversal:

c2. Latent traversal:

->Pick a direction in latent (u_m) then:

$$z'(t) = z + tu_m$$

->ask the decoder (if exists):

$$\hat{x}'(t) = D(z'(t))$$

->And see which FV_k are changing

But how to decide the direction of u_m? There are many ways, some simpler methods are:

A. Coordinate directions $u_m = e_m$

B. Unsupervised directions **PCA** on latent codes: $u_m = \mathrm{PCA}_m(Z)$

C. Factor-supervised directions:

- First encode all samples: $z_i = f_\theta(x_i)$
- For factor FV_k with values $c = 0,...,C_k - 1$ compute:

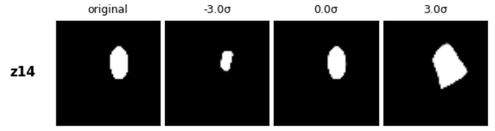
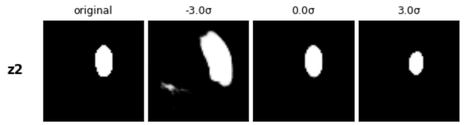
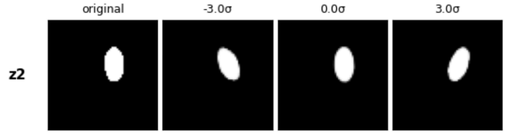
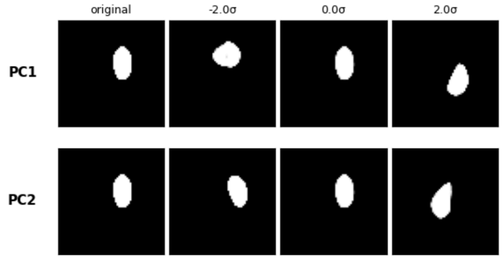
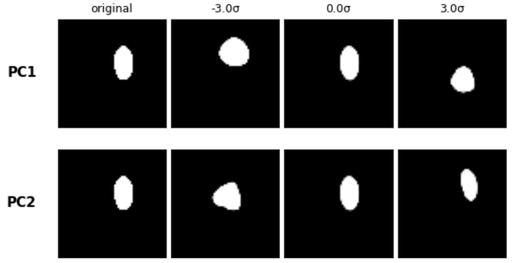
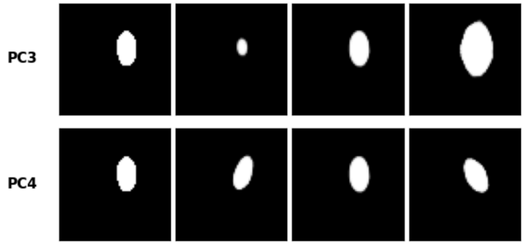
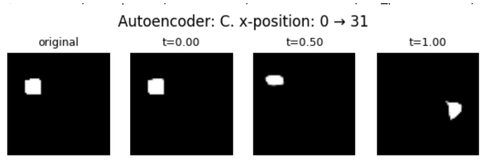
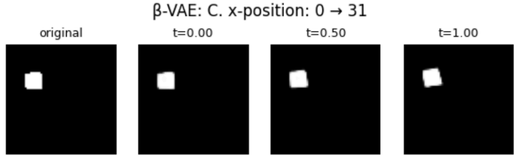
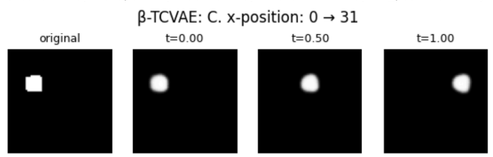
$$\mu_{k,c} = \frac{1}{N_c} \sum_{i:FV_k(x_i)=c} z_i$$

- Define directions between factor values: $u_k = \mu_{k,C_k-1} - \mu_{k,0}$

Example: $u_{\text{scale}} = \mu_{\text{scale},5} - \mu_{\text{scale},0}$ or $u_{square \to heart} = \mu_{heart} - \mu_{square}$

- Then traverse based on u_k

c-Disentanglement: c.2 Latent traversal:

	Autoencoder	B-VAE	β-TCVAE
A. Coordinate directions			
B. Unsupervised directions (PCA)			
C. Factor-supervised directions:			

c-Disentanglement: c.3 Ablation

c3. Ablation:

(i) Remove one dimension/subspace m:

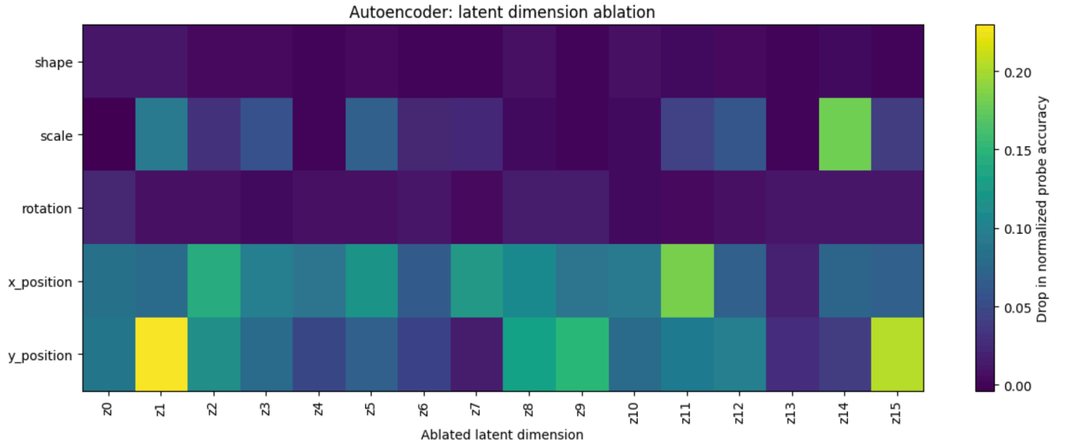
$$z \rightarrow z_{-m}$$

(ii) Then measure which factor prediction drops:

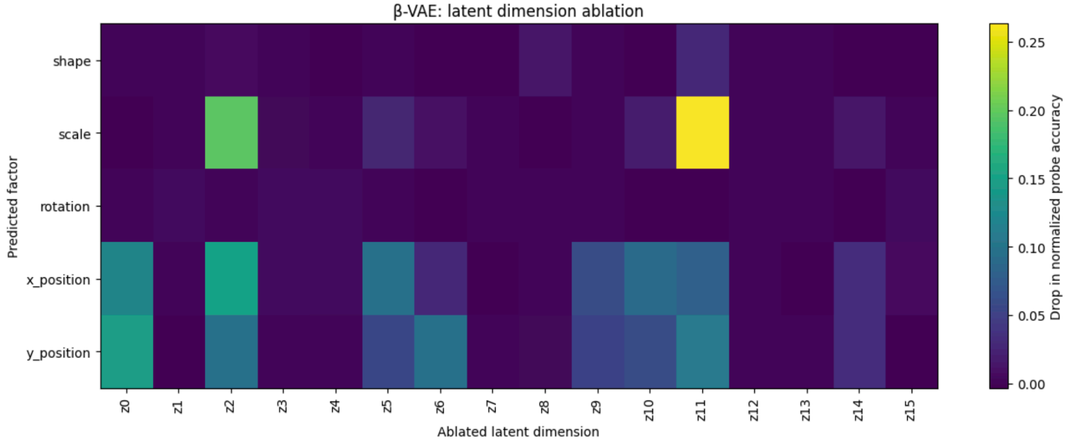
$$A_{k,m} = \text{Score}_k(z) - \text{Score}_k(z_{-m})$$

(which latent parts are important which factor)

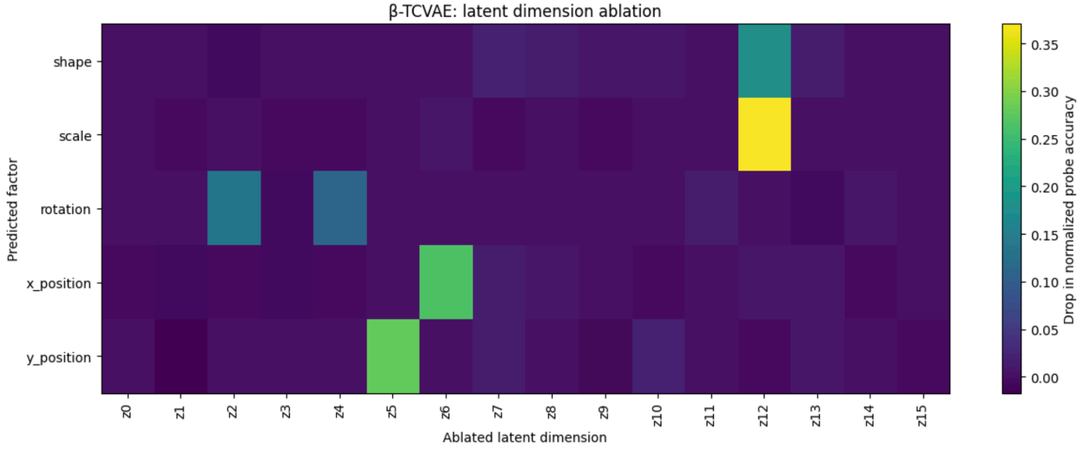
Autoencoder



B-VAE



β-TCVAE



c-Disentanglement: Caveats and further readings

Note that purely unsupervised disentanglement is not reliably identifiable without inductive bias or some form of supervision. Locatello et al (2019). showed this theoretically and empirically, which is why modern work often moves toward:

- weak supervision
- causal assumptions
- contrastive learning
- temporal structure, discrete/quantized latents,
- or task-specific inductive biases

Also look at:

https://research.google/blog/evaluating-the-unsupervised-learning-of-disentangled-representations/?utm_source=chatgpt.com

c-Disentanglement: recent state of research

Impossibility Without Inductive Biases

Locatello et al. (2019) proved unsupervised disentanglement is fundamentally impossible without assumptions. Standard VAEs with different random seeds learn different representations: no guarantee of ground-truth factor recovery. Locatello et al., "Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations," ICML 2019

Shift to Weakly-Supervised Methods

Research currently more focuses on minimal supervision: sparse labels, temporal structure, or known transformations. These inductive biases enable identifiable disentanglement without full supervision.Locatello et al., "Weakly-Supervised Disentanglement Without Compromises," ICML 2020; Klindt et al., "Towards Nonlinear Disentanglement in Natural Data with Temporal Sparse Coding," ICLR 2021

Causal Disentanglement

Modern approaches use causal structure: independent mechanisms principle, interventional robustness. Focus shifts from "separating factors" to "learning causal mechanisms”: more robust to distribution shifts. Suter et al., "Robustly Disentangled Causal Mechanisms," NeurIPS 2019; von Kügelgen et al., "Self-Supervised Disentanglement by Leveraging Structure in Data Augmentations," arXiv 2021

Platonic hypothesis

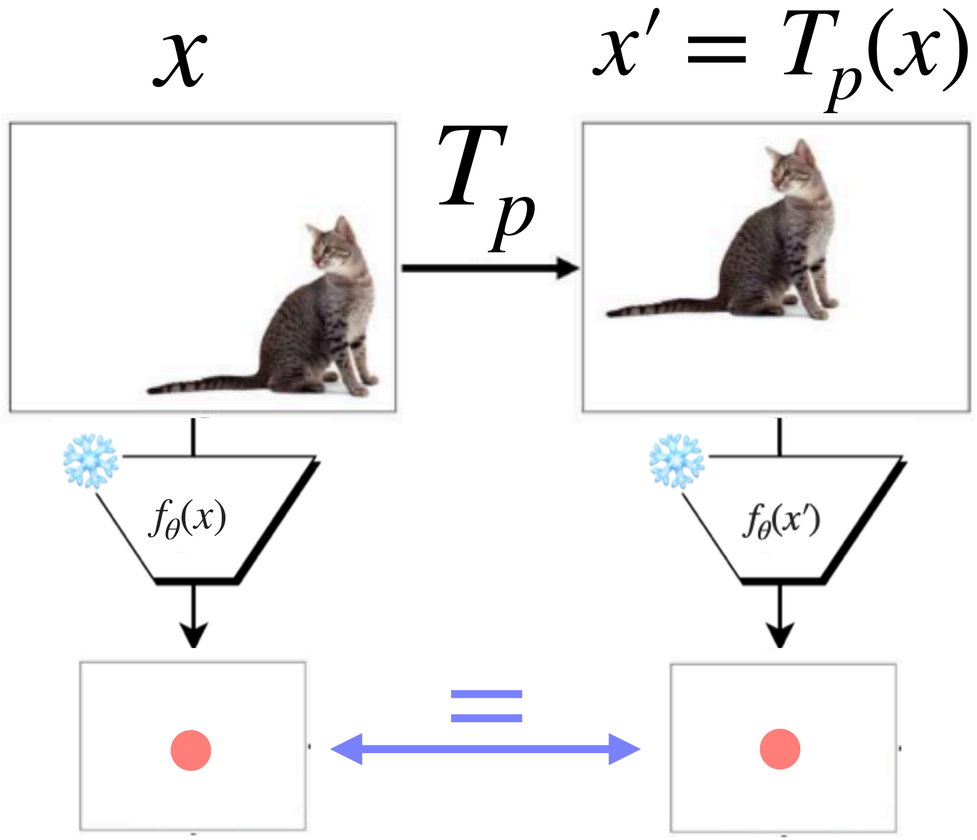
Yet larger models converge in terms of structure of the representation (organization).
Huh, Minyoung, et al. "The platonic representation hypothesis." arXiv preprint arXiv:2405.07987 (2024).
https://arxiv.org/pdf/2512.03750
https://arxiv.org/pdf/2512.05717 www.nature.com/articles/s42256-026-01235-7

Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations

Francesco Locatello [1 2] **Stefan Bauer** [2] **Mario Lucic** [3] **Gunnar Rätsch** [1] **Sylvain Gelly** [3] **Bernhard Schölkopf** [2] **Olivier Bachem** [3]

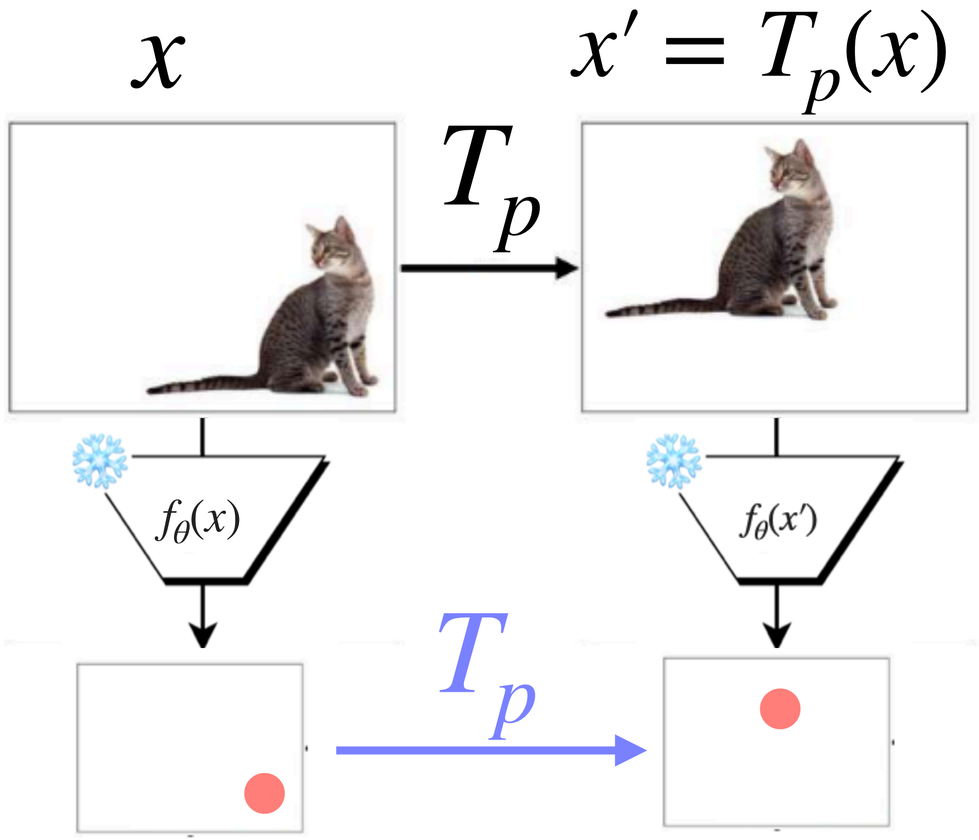
Reminder of what Invariance and Equivariance representations z to transformation T_p are:

Invariance



$z = f_\theta(x)$ $\quad z' = f_\theta(x') = f_\theta(T_p(x))$

Equivariance



$z = f_\theta(x)$ $\quad z' = f_\theta(x') = f_\theta(T_p(x))$

d-Invariance: Does it ignore <u>irrelevant</u> variations?

As before, the representation encoder is:

$$z = f_\theta(x)$$

Now if we apply a controlled transformation T to the input we have:

$$x' = T(x)$$
$$z' = f_\theta(x') = f_\theta(T(x))$$

Our representation is invariant to T_p if:

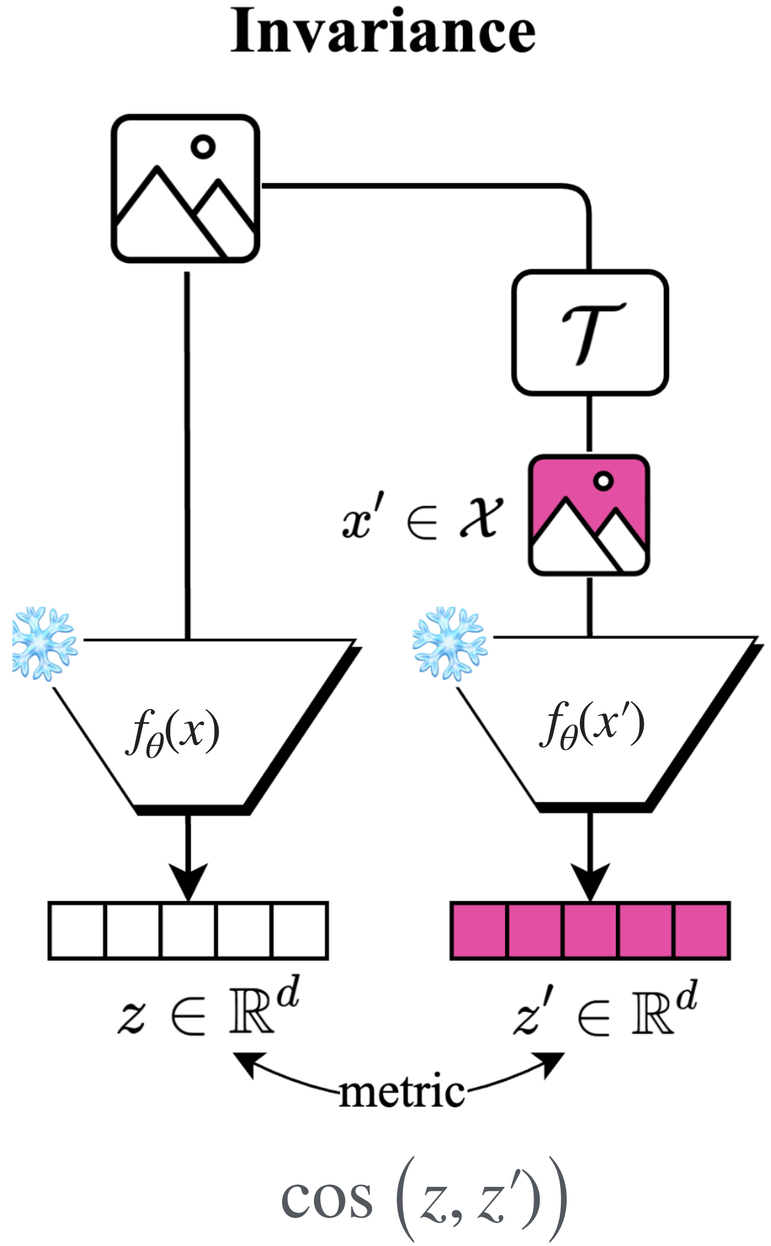
$$f_\theta(T(x)) \approx f_\theta(x)$$

More quantitatively we can measure this using cosine similarity:

$$\text{Inv}_p = \frac{1}{N}\sum_{i=1}^{N} \cos\left(f_\theta(x_i), f_\theta(T(x_i))\right)$$

$\text{Inv}_p \approx 1$: the representation is invariant under transformation T

$\text{Inv}_p << 1$: the representation is not invariant under transformation T



d-Invariance: Example setup

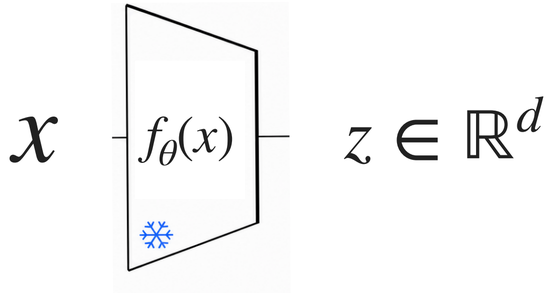
Dataset → **Model** → **Analysis**

dsprites-dataset

FVs:

- shapes (square, ellipse, heart)
- scales (6 scale)
- rotations (40 rotations)
- x-position (32 positions)
- y-positions (32 positions)

(737,280 pre-rendered images)



Model 2: train and freeze an Autoencoder
Model 3: train and freeze a β-VAE
Model 4: train and freeze β-TCVAE

d. Invariance:

We will create paired images:

$$x \text{ and } x' = T(x)$$

And the encoding via frozen model:

$$z = F_\theta(x)$$

$$z' = F_\theta(x') = F_\theta(T(x))$$

And then will measure the Invariance:

$$\text{Inv}(T) = \frac{1}{N}\sum_{i=1}^{N} \cos\left(F_\theta(x_i), F_\theta(T(x_i))\right)$$

Our test Test transformations are:

T_x : change x-position -> relevant!

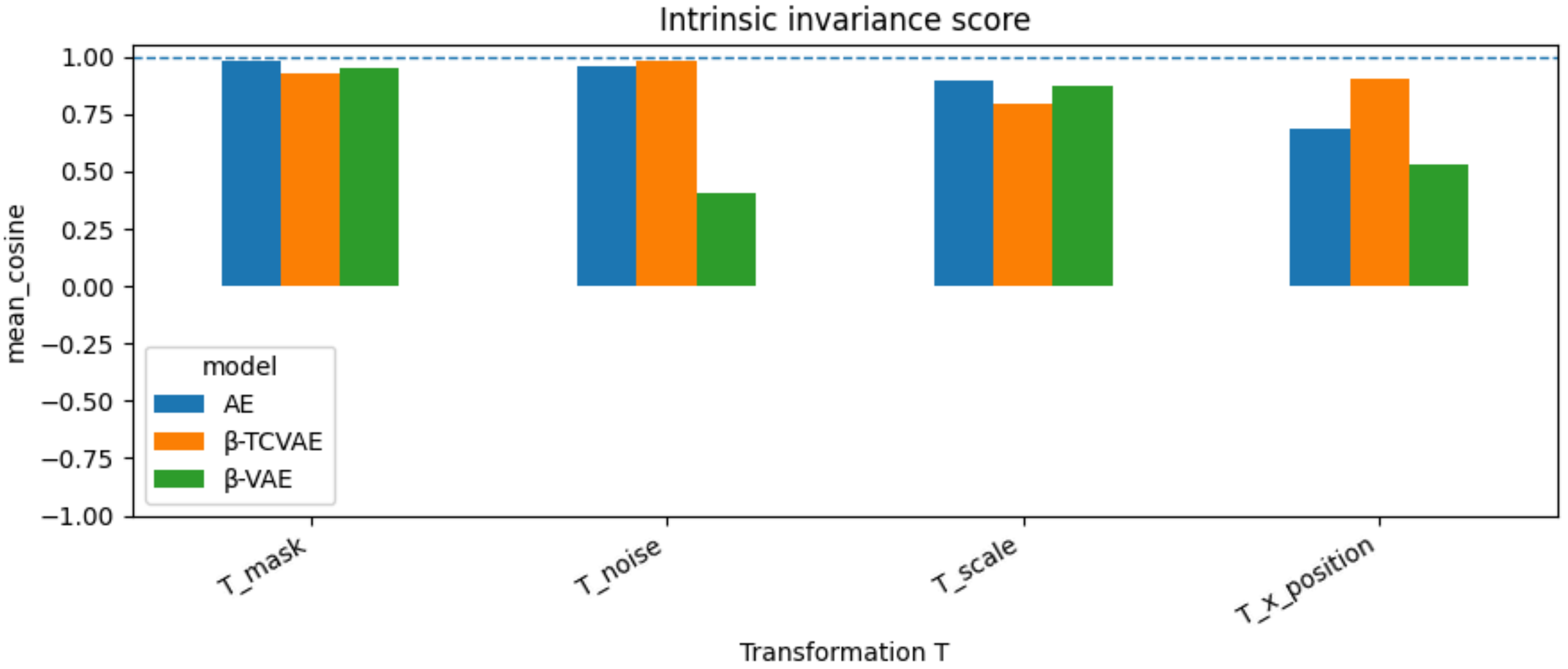
T_{scale} : change scale -> relevant!

T_{noise} : add pixel noise -> irrelevant!

T_{mask} : partial occlusion > irrelevant!

Let's see hoe there models do

d-Invariance: Example setup



d-Invariance:

Modern practice usually chooses T from three sources:

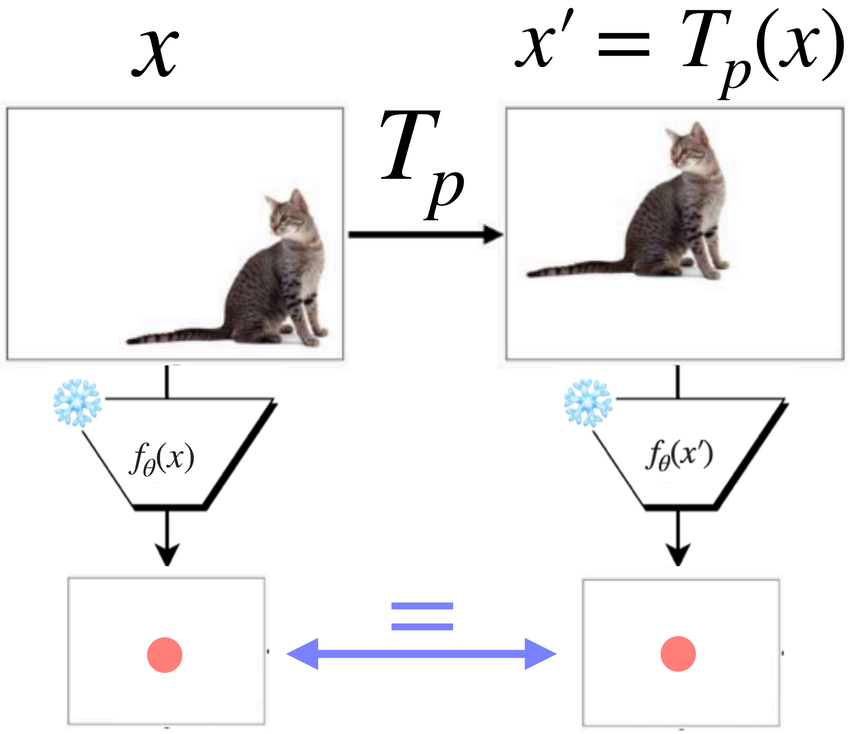
known semantic factors

standard corruption/augmentation families,

domain-specific nuisance transformations (Geometric or on domain)

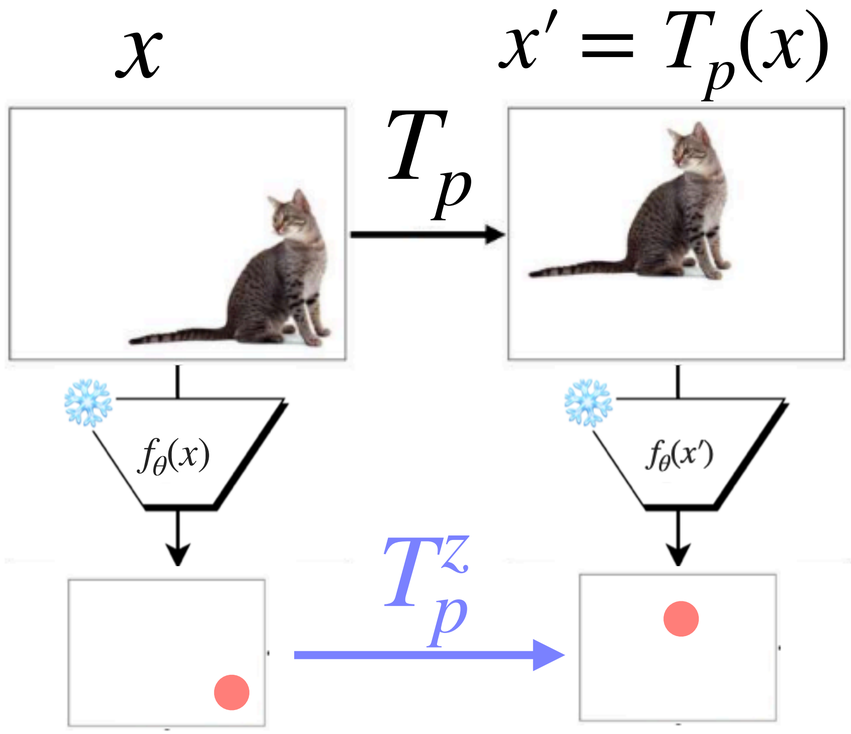
Reminder of what Invariance and Equivariance representations z to transformation T_p are:

Invariance



$z = f_\theta(x)$ $\quad$ $z' = f_\theta(x') = f_\theta(T_p(x))$

Equivariance



$z = f_\theta(x)$ $\quad$ $z' = f_\theta(x') = f_\theta(T_p(x))$

Challange for studying Equivaraince: The input-space transformation T_p is known to us, but its counterpart in the latent-space, transformation T_p^z is usually unknown. Therefore we will try to learn it.

e-Equivariance: Does it track meaningful transformations?

As before, the representation encoder is:

$$z = f_\theta(x)$$

Now if we apply a controlled transformation T_p to the input we have:

$$x' = T_p(x)$$

$$z' = f_\theta(x') = f_\theta(T_p(x))$$

In equivariance to T_p, we ask whether the representation changes in a predictable way:

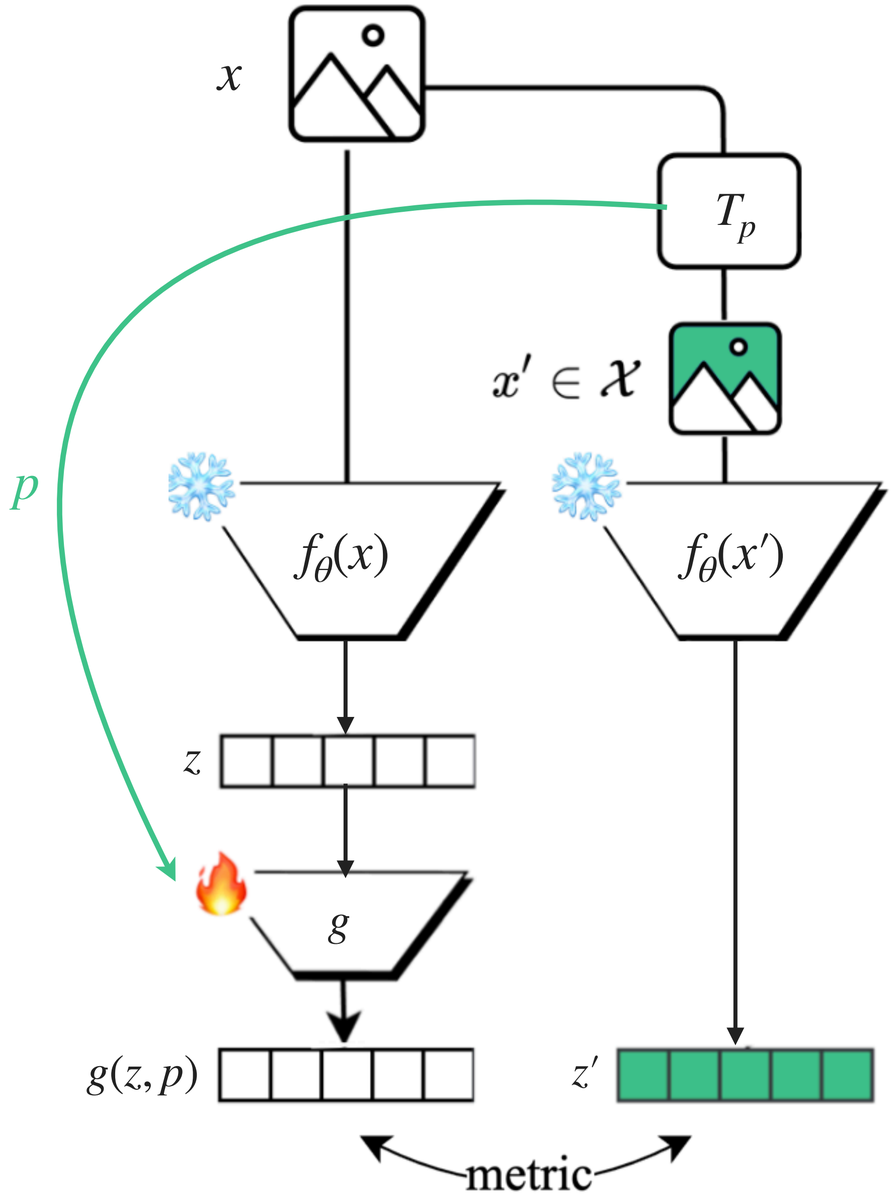
$$F_\theta(T_p(x)) \approx T_p^z(F_\theta(x))$$

Or

$$z' \approx T_p^z(z)$$

Practically, we test whether a T_p^z can be learned from data by a probe g when the transformation $x' = T_p(x)$ is applied:

$$z' \approx g(z, p)$$



$$score(g(z,p), z')$$

e-Equivariance: Example setup

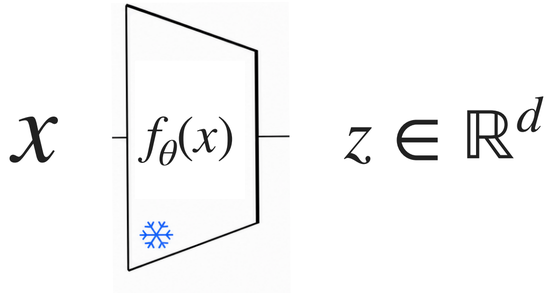
Dataset → **Model** → **Analysis**

dsprites-dataset

FVs:

- shapes (square, ellipse, heart)
- scales (6 scale)
- rotations (40 rotations)
- x-position (32 positions)
- y-positions (32 positions)

(737,280 pre-rendered images)



Model 2: train and freeze an Autoencoder
Model 3: train and freeze a β-VAE
Model 4: train and freeze β-TCVAE

e. Equivaraince:

We will create paired images:

$$x \text{ and } x' = T_p(x)$$

where p is the transformation parameter.

Encode both images with the frozen model:

$$z = F_\theta(x)$$
$$z' = F_\theta(x') = F_\theta(T(x))$$

Equivariance asks whether z' can be predicted from z and p by a probe g:

$$g(z, p)$$

We measure prediction quality:

$$R^2(z', g(z, p))$$

Our test Test transformations are:

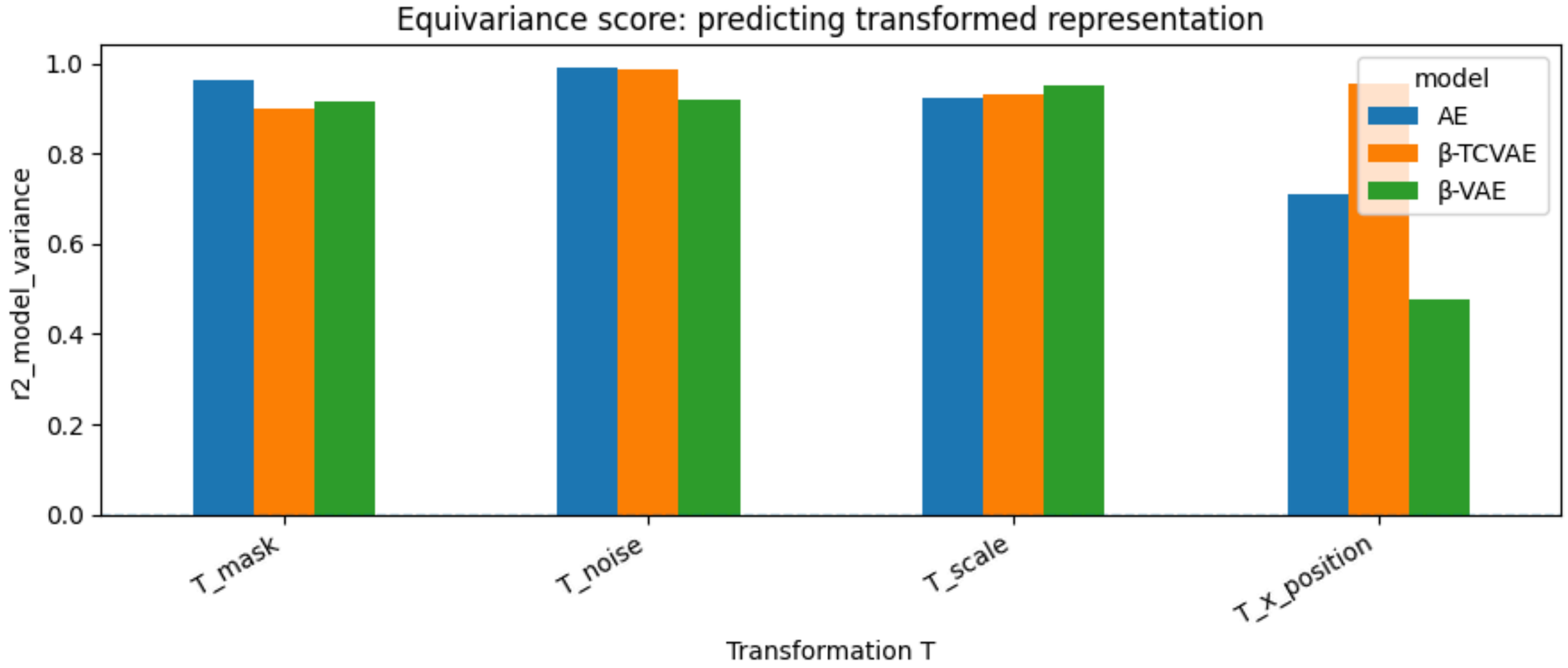
T_x : change x-position -> relevant!

T_{scale} : change scale -> relevant!

T_{noise} : add pixel noise -> irrelevant!

T_{mask} : partial occlusion > irrelevant!

e-Equivariance: Example setup



Read more:

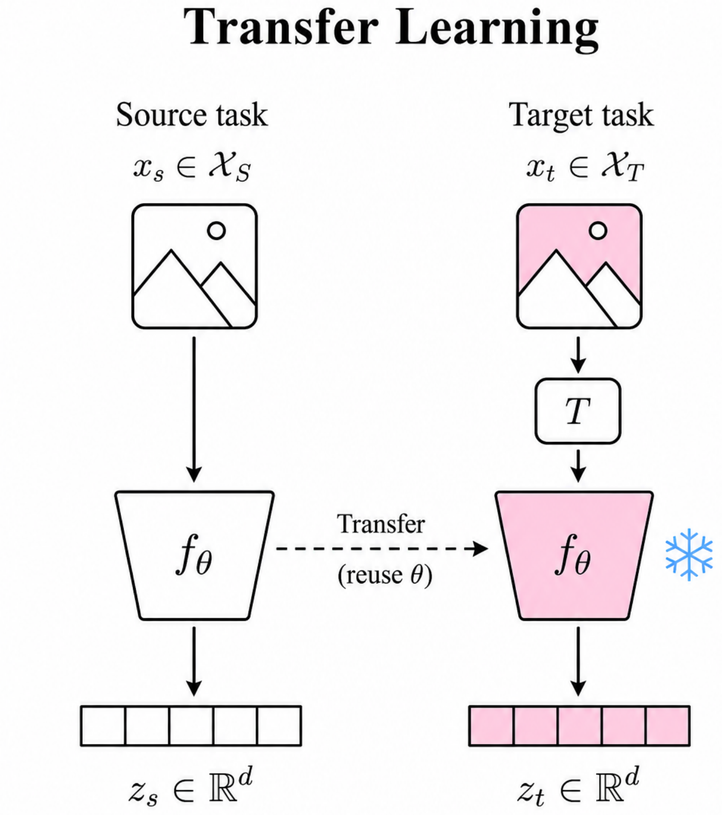
Eastwood, Cian, and Christopher KI Williams. "A framework for the quantitative evaluation of disentangled representations." *International conference on learning representations*. 2018.

Wang, Yifei, et al. "Understanding the role of equivariance in self-supervised learning." *Advances in Neural Information Processing Systems* 37 (2024): 127483-127510.

f-Transferability: Does it generalize to new tasks or domains?

A representation is transferable if the frozen encoder can be reused beyond the task or data distribution it was trained on.

This is the key concept in '**Transfer learning**'.



Yosinski, Jason, et al. "How transferable are features in deep neural networks?." *Advances in neural information processing systems* 27 (2014).

Weiss, Karl, Taghi M. Khoshgoftaar, and DingDing Wang. "A survey of transfer learning." *Journal of Big data* 3.1 (2016): 9.

Zhuang, Fuzhen, et al. "A comprehensive survey on transfer learning." *Proceedings of the IEEE* 109.1 (2020): 43-76.

Recipe Card: **Representation Evaluation**

Data

1-Load the Dataset of interest (evaluation dataset)

$\mathcal{D} = \{x_i, y_i\}_{i=1}^{N}$

2-Define Task specific functions/transformations:

i-Define optional factors of variation in the dataset: $FV(x_i)$

Examples: class label, cell type, disease state, hue, brightness, batch, perturbation.

ii-Define optional transformations: $x_i' = T_\alpha(x_i)$

Examples: noise, crop, rotation, brightness shift, gene dropout, domain shift, translation.

Model

3-Encoder and representation

-Load an encoder: f_θ

-Choose the layer used as the representation: $z_i = f_\theta(x_i)$

-Examples: penultimate layer, bottleneck, CLS token, pooled graph/set embedding.

-Usually freeze the encoder: θ fixed (in torch: requires_grad = False)

Prob

4- We usually train a small probe g_ϕ on top of the representation output z_i:

$$\hat{v}_i = g_\phi(z_i)$$

-where v_i is the property we want to test.

-The probe asks: Is this information accessible from z?

Evaluation plug-ins

5.a Informativeness :Question: Can we recover useful information from z?

-Usually done by comparing: $v = FV(x)$ vs. $\hat{v} = g_\phi(z)$

5.b Organization: Question: Is the latent space meaningful? $d(z_i, z_j)$

-Usually done by analysis of the z organization, examples are: kNN accuracy, clustering scores, Calculating z Kernel for different FVs, intrinsic dimension, PCA/UMAP, etc.

5.c Transferability: Question: Does z work on new tasks or domains?

$$f_\theta \text{ trained on A} \rightarrow g_\phi(z_B) \text{ tested on B}$$

-Usually done by evaluating downstream performance on new data.

5.d Disentanglement: Question: Are different factors encoded separately?

$$g_k(z) \approx FV_k(x)$$

- use factor-specific probes, ablation, MIG/DCI/SAP when ground-truth factors exist.

5.e Invariance: Question: Does z ignore irrelevant transformations (T)?

- Original: x and $z = f_\theta(x)$

Transformed input: $x' = T(x)$ and $z' = f_\theta(x')$.

Good invariance is: $z \approx z'$. A common metric is cosine similarity: $\|z - z'\|_2$

5.f Equivariance: Question: Does z track meaningful transformations?

-Original: x and $z = f_\theta(x)$

-Apply a known transformation with parameter P: $x' = T_P(x)$ and $z' = f_\theta(x')$

f.1 -Parameter equivariance: Can we predict what transformation happened?

$$g_\phi(z \oplus z') \approx P$$

f.2 Representation equivariance: Can we predict the transformed representation?

$$g_\phi(z, p') \approx z'$$

Metric: transformation-prediction accuracy, R^2 , MSE, representation-prediction error.