Best ideas of 2021

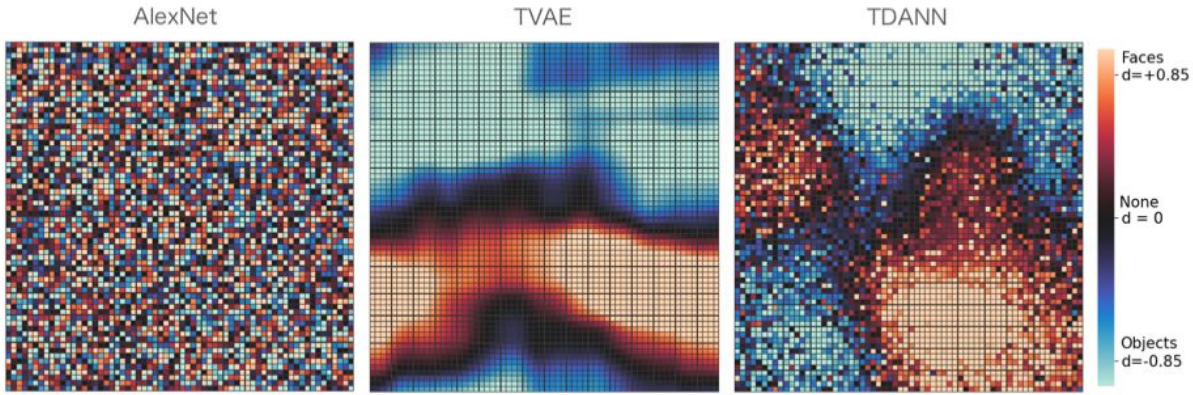
MLBBQ list

No more ML research :-(

Yeah, dawn of the hype :(

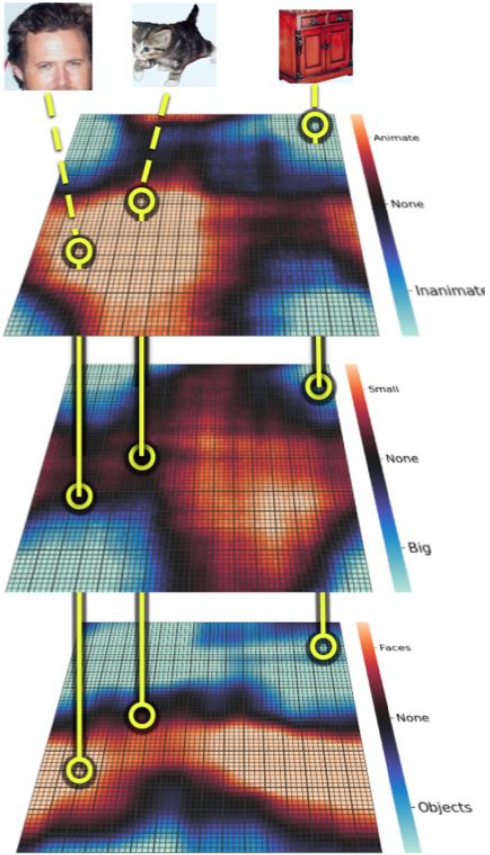
Eloy Geenjaar

Topographic VAEs

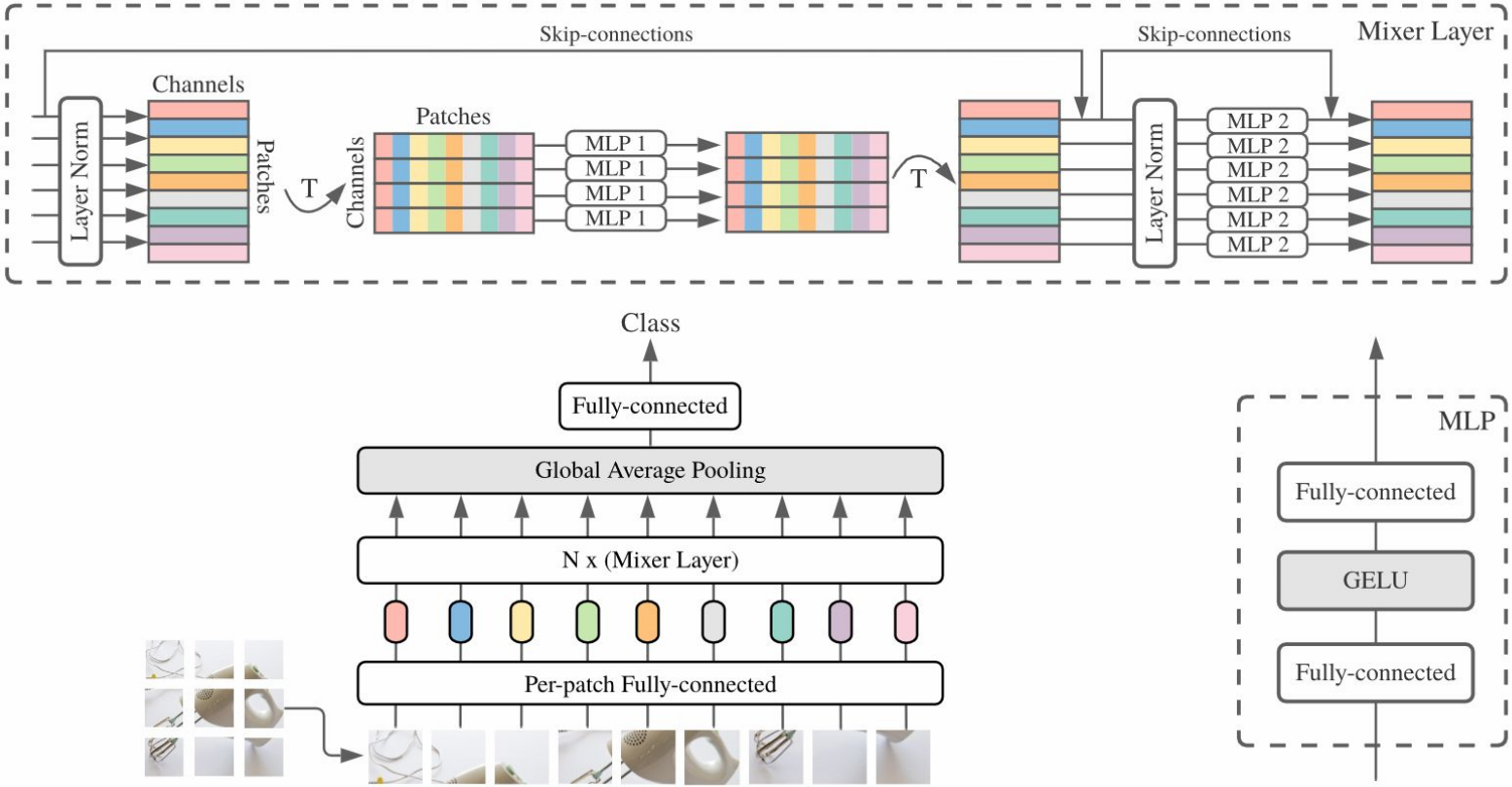


The cortex is topographic. https://arxiv.org/pdf/2110.13911.pdf

Topographic VAEs learn equivariant capsules
https://arxiv.org/pdf/2110.13911.pdf



“MLP-Mixer: An all-MLP Architecture for Vision”



https://arxiv.org/abs/2105.01601 - June 2021

Added by Brad

Explicit Input Patches in Vision

[Masked Autoencoders are Scalable Vision Learner]

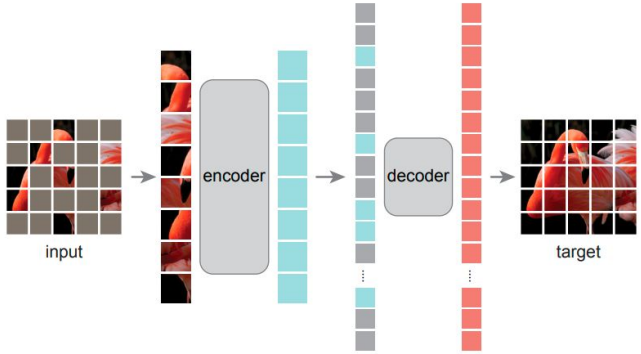


Figure 1. **Our MAE architecture**. During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

Added by Kevin Wang

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

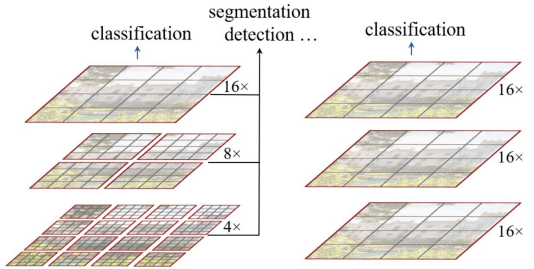


Figure 1. (a) The proposed Swin Transformer builds hierarchical feature maps by merging image patches (shown in gray) in deeper layers and has linear computation complexity to input image size due to computation of self-attention only within each local window (shown in red). It can thus serve as a general-purpose backbone for both image classification and dense recognition tasks. (b) In contrast, previous vision Transformers [20] produce feature maps of a single low resolution and have quadratic computation complexity to input image size due to computation of self-attention globally.

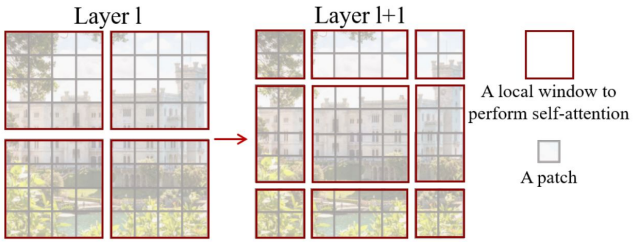


Figure 2. An illustration of the *shifted window* approach for computing self-attention in the proposed Swin Transformer architecture. In layer l (left), a regular window partitioning scheme is adopted, and self-attention is computed within each window. In the next layer $l+1$ (right), the window partitioning is shifted, resulting in new windows. The self-attention computation in the new windows crosses the boundaries of the previous windows in layer l, providing connections among them.

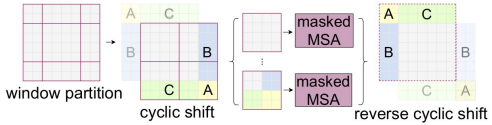
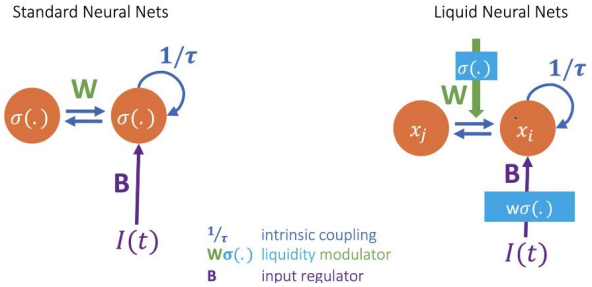
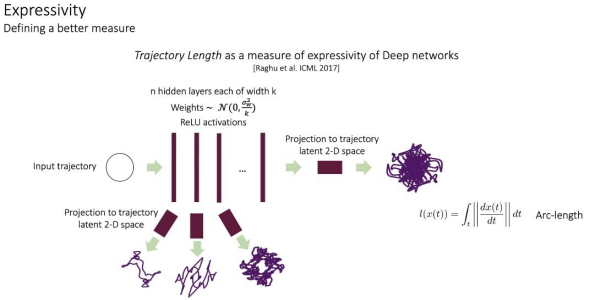


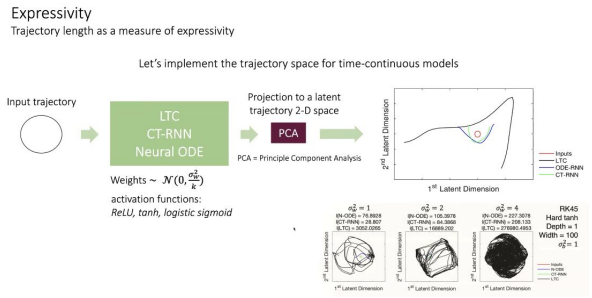
Figure 4. Illustration of an efficient batch computation approach for self-attention in shifted window partitioning.

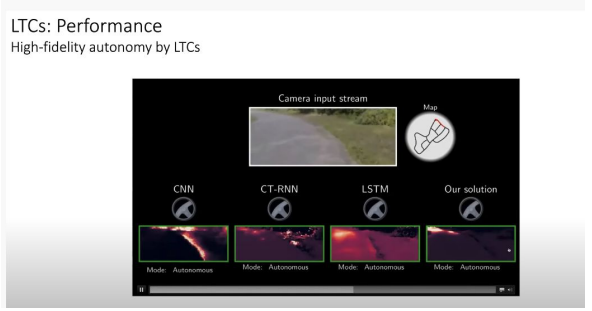
Added by William Ashbee

Liquid Neural Networks (Liquid Time-constant Networks)





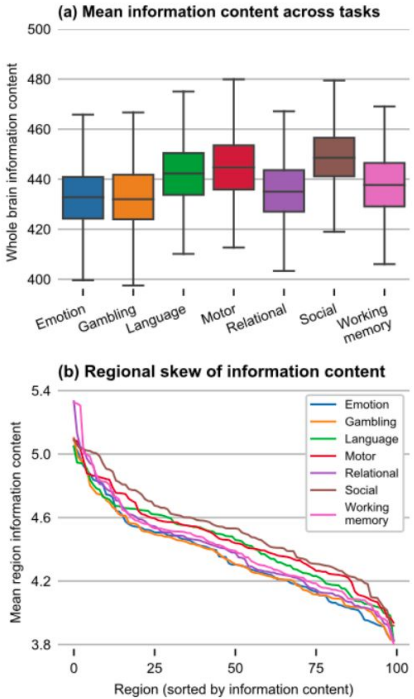


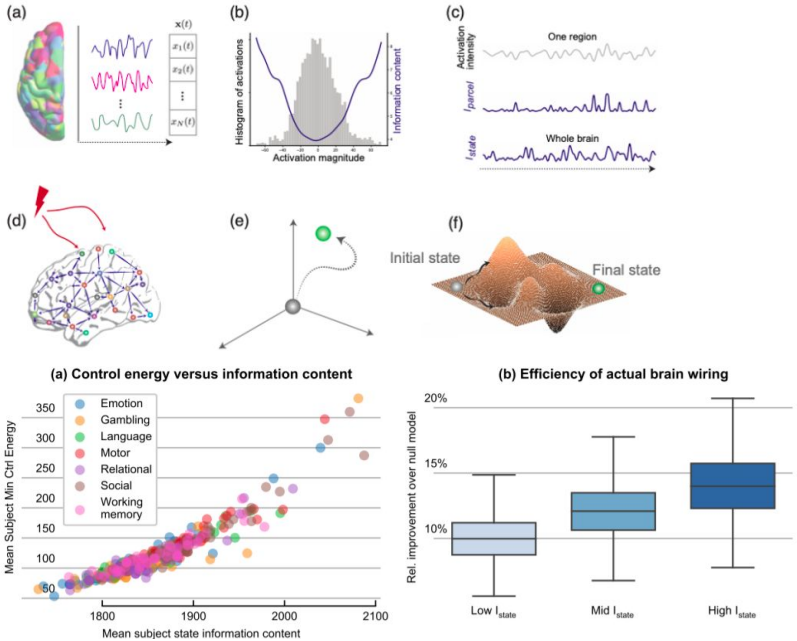


Usman Mahmood

The Information Content of Brain States is Explained by Structural Constraints on State Energetics

By Leon Weninger, Pragya Srivastava, Dale Zhou, Jason Z. Kim, Eli J. Cornblath, Maxwell A. Bertolero, Ute Habel, Dorit Merhof, Dani S. Basset





Other notable papers:

- **SwinIR: Image Restoration Using Swin Transformer** by J. Liang et al. [link]
- **A Continuized Accelerations of Deterministic and Stochastic Gradient Descents, and of Gossip Algorithms** by Even et al. [link]
- **Reward is enough** by Silver et al. [link]
- **BioGrad: Biologically Plausible Gradient-Based Learning for Spiking Neural Networks** by Tang et al. [link]
- **Reduced, Reused and Recycled: The Life of a Dataset in Machine Learning Research** by Koch et al. [link]
- **Can a Fruit Fly Learn Word Embeddings?** by Y. Liang et al. [link]

Added by TJ LaGrow

Source: https://arxiv.org/pdf/2110.13781.pdf

Improving Deep Learning Interpretability by Saliency Guided Training

Aya Abdelsalam Ismail, Héctor Corrada Bravo, Soheil Feizi

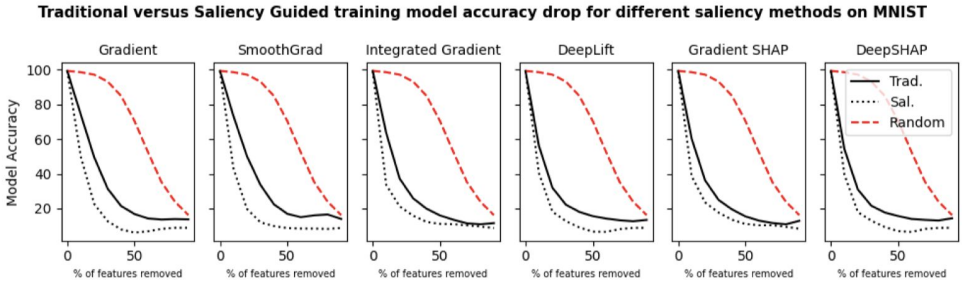


Figure 3: Model accuracy drop when removing features with high saliency using traditional an the saliency guided training for different gradient-based methods against a random baseline. A steeper drop indicates a better performance. We find that regardless of the saliency method used, th performance improves by the saliency guided training.

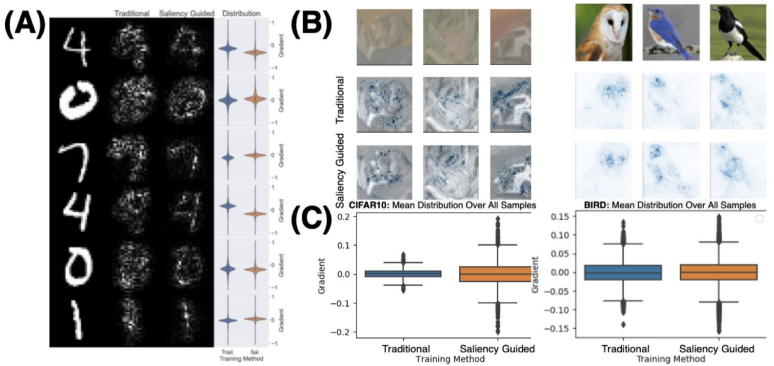


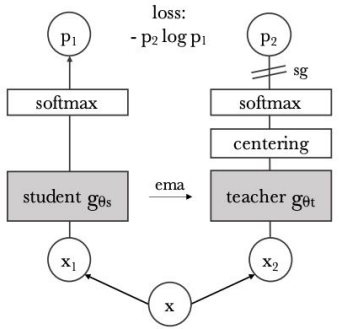
Figure 2: (A) Comparison between different training methods on MNIST along with distributions of gradient values in each sample. (B) Saliency maps for CIFAR10 and BIRD datasets using regular and saliency guided training. (C) Distribution of gradient means across examples. Maps produced by saliency guided training are more precise: most features have gradient values around zero with large gaps between mean and outliers. Here gradients around zero indicate uninformative features, while very large and very small gradients indicate informative features. Saliency guided training helps reduce noisy fluctuating gradients in between as shown in the box plots.

$$\underset{\theta}{\text{minimize}} \; \frac{1}{n} \sum_{i=1}^{n} \left[\mathcal{L}\left(f_\theta\left(X_i\right), y_i \right) + \lambda D_{KL}\left(f_\theta\left(X_i\right) \parallel f_\theta(\widetilde{X}_i) \right) \right]$$

Added by Mahfuz

https://proceedings.neurips.cc/paper/2021/file/e0cd3f16f9e883ca91c2a4c24f47b3d9-Paper.pdf

DINO: Emerging Properties in Self-Supervised Vision Transformer



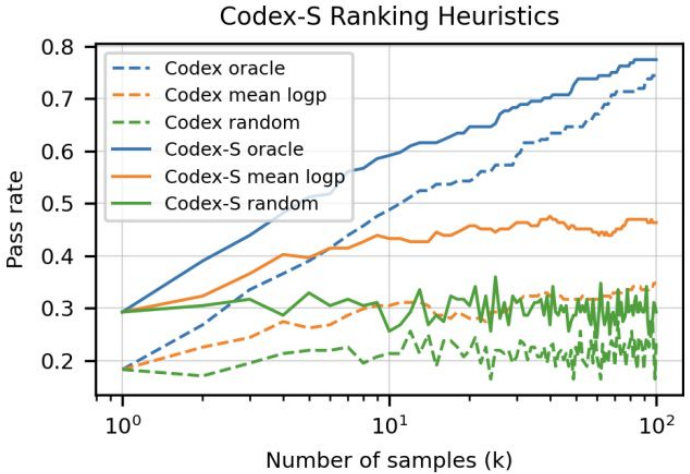
Added by: Mrinal

Evaluating Large Language Models Trained on Code

```python
def incr_list(l: list):
    """Return list with elements incremented by 1.
    >>> incr_list([1, 2, 3])
    [2, 3, 4]
    >>> incr_list([5, 3, 5, 2, 3, 3, 9, 0, 123])
    [6, 4, 6, 3, 4, 4, 10, 1, 124]
    """
    return [i + 1 for i in l]
```

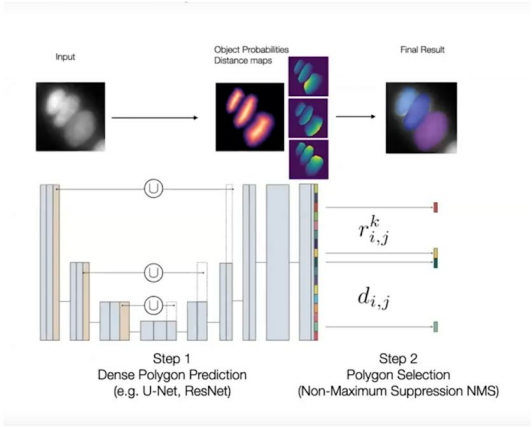
```python
def solution(lst):
    """Given a non-empty list of integers, return the sum of all of the odd elements
    that are in even positions.

    Examples
    solution([5, 8, 7, 1]) ==>12
    solution([3, 3, 3, 3, 3]) ==>9
    solution([30, 13, 24, 321]) ==>0
    """
    return sum(lst[i] for i in range(0,len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```

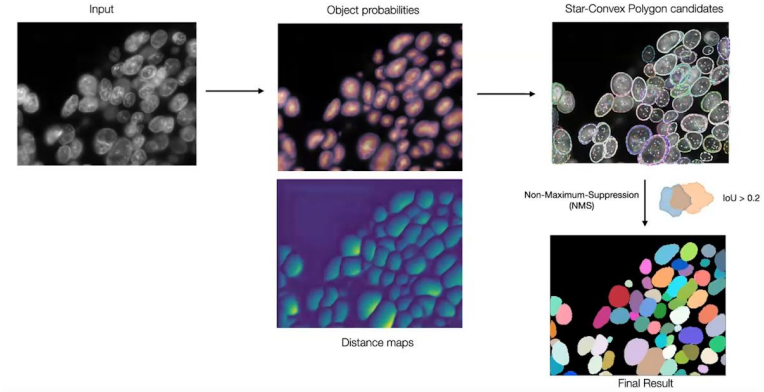
```python
def encode_cyclic(s: str):
    """
    returns encoded string by cycling groups of three characters.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group. Unless group has fewer elements than 3.
    groups = [(group[1:] + group[0]) if len(group) == 3 else group for group in groups]
    return "".join(groups)

def decode_cyclic(s: str):
    """
    takes as input string encoded with encode_cyclic function. Returns decoded string.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group.
    groups = [(group[-1] + group[:-1]) if len(group) == 3 else group for group in groups]
    return "".join(groups)
```

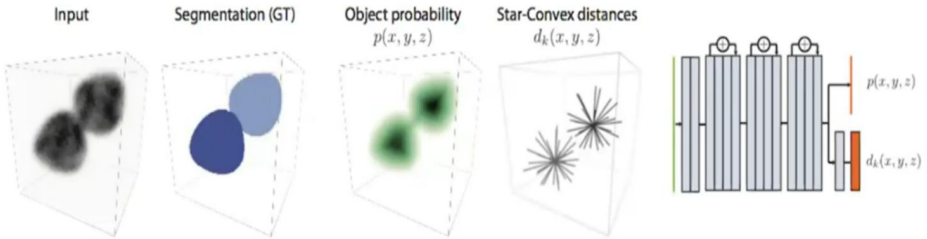


Added by: Mrinal

Star-convex Polyhedra for 3D Object Detection and Segmentation in Microscopy







Masoud

https://arxiv.org/pdf/1908.03636.pdf

Towards Causal Representation Learning

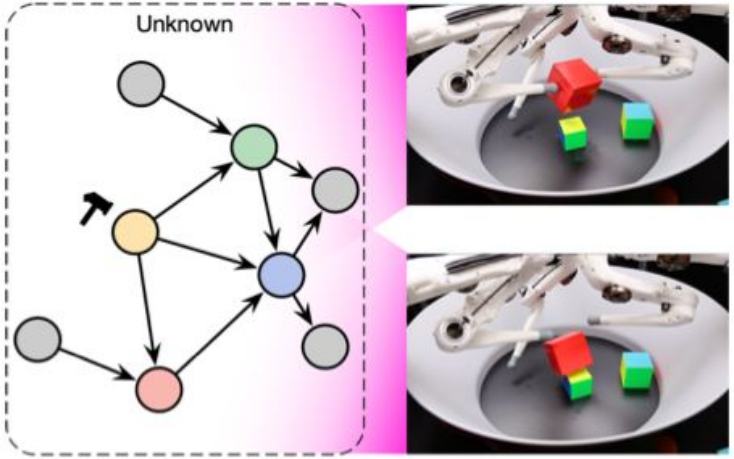


Fig. 3. Example of the SMS hypothesis where an intervention (which may or may not be intentional/observed) changes the position of one finger (↖), and as a consequence, the object falls. The change in pixel space is entangled (or distributed), in contrast to the change in the causal model.

Sajad

https://arxiv.org/pdf/2102.11107v1.pdf

Pruning at Initialization

A range of techniques have emerged in the last couple of years to tackle the problem of pruning at initialization i.e. pruning before training. These methods try to tackle the problem of finding out the most important connections in model before training the model.

SNIP and GraSP are the two most popular methods that have emerged.

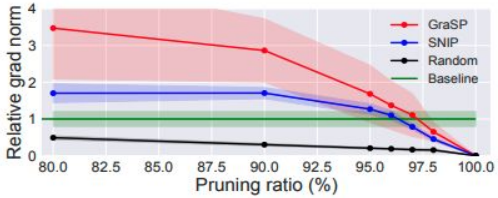


Figure 2: The gradient norm of ResNet32 after pruning on CIFAR-100 of various pruning ratios. Shaded area is the 95% confidence interval calculated with 10 trials.

SNIP: Single Shot Network Pruning Based on Connection Sensitivity

Namhoon Lee et al.

Based on the criterion of connection sensitivity to loss.

$$S(\theta_q) = \lim_{\epsilon \to 0} \left| \frac{\mathcal{L}(\boldsymbol{\theta}_0) - \mathcal{L}(\boldsymbol{\theta}_0 + \epsilon \boldsymbol{\delta}_q)}{\epsilon} \right| = \left| \theta_q \frac{\partial \mathcal{L}}{\partial \theta_q} \right|$$

GraSP: Picking Winning Tickets Before Training By Preserving Gradient Flow

Chaoqi Wang et al.

A larger gradient norm indicates that, to the first order, each gradient update achieves a greater loss reduction characterized by the following directional derivative:

$$\Delta \mathcal{L}(\boldsymbol{\theta}) = \lim_{\epsilon \to 0} \frac{\mathcal{L}\left(\boldsymbol{\theta} + \epsilon \nabla \mathcal{L}(\boldsymbol{\theta})\right) - \mathcal{L}(\boldsymbol{\theta})}{\epsilon} = \nabla \mathcal{L}(\boldsymbol{\theta})^\top \nabla \mathcal{L}(\boldsymbol{\theta})$$

A Taylor approximation is used to characterize how removing one weight will affect the gradient flow after pruning.

$$\begin{aligned} \mathbf{S}\left(\boldsymbol{\delta}\right) &= \Delta \mathcal{L}(\boldsymbol{\theta}_0 + \boldsymbol{\delta}) - \underbrace{\Delta \mathcal{L}(\boldsymbol{\theta}_0)}_{\text{Const}} = 2\boldsymbol{\delta}^\top \nabla^2 \mathcal{L}(\boldsymbol{\theta}_0) \nabla \mathcal{L}(\boldsymbol{\theta}_0) + \mathcal{O}(\|\boldsymbol{\delta}\|_2^2) \\ &= 2\boldsymbol{\delta}^\top \mathbf{H}\mathbf{g} + \mathcal{O}(\|\boldsymbol{\delta}\|_2^2), \end{aligned}$$

MLBBQ LOGO DESIGN

https://docs.google.com/presentation/d/1ZbgWZLNnKpcDPjAemLNrWqsFHjC4GadI_42NAP9XbRg/edit

Idea 1 - B as Brain

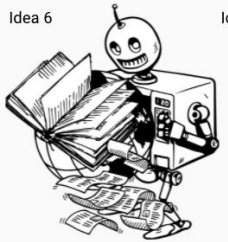
Idea 2 - food and plate

Idea 3 - Robot cooking BBQ

Idea 4 - Robot + Meat

Idea 5 - Host + Robot + Sausage

Idea 6



Idea 7

Added by Farfalla Hu