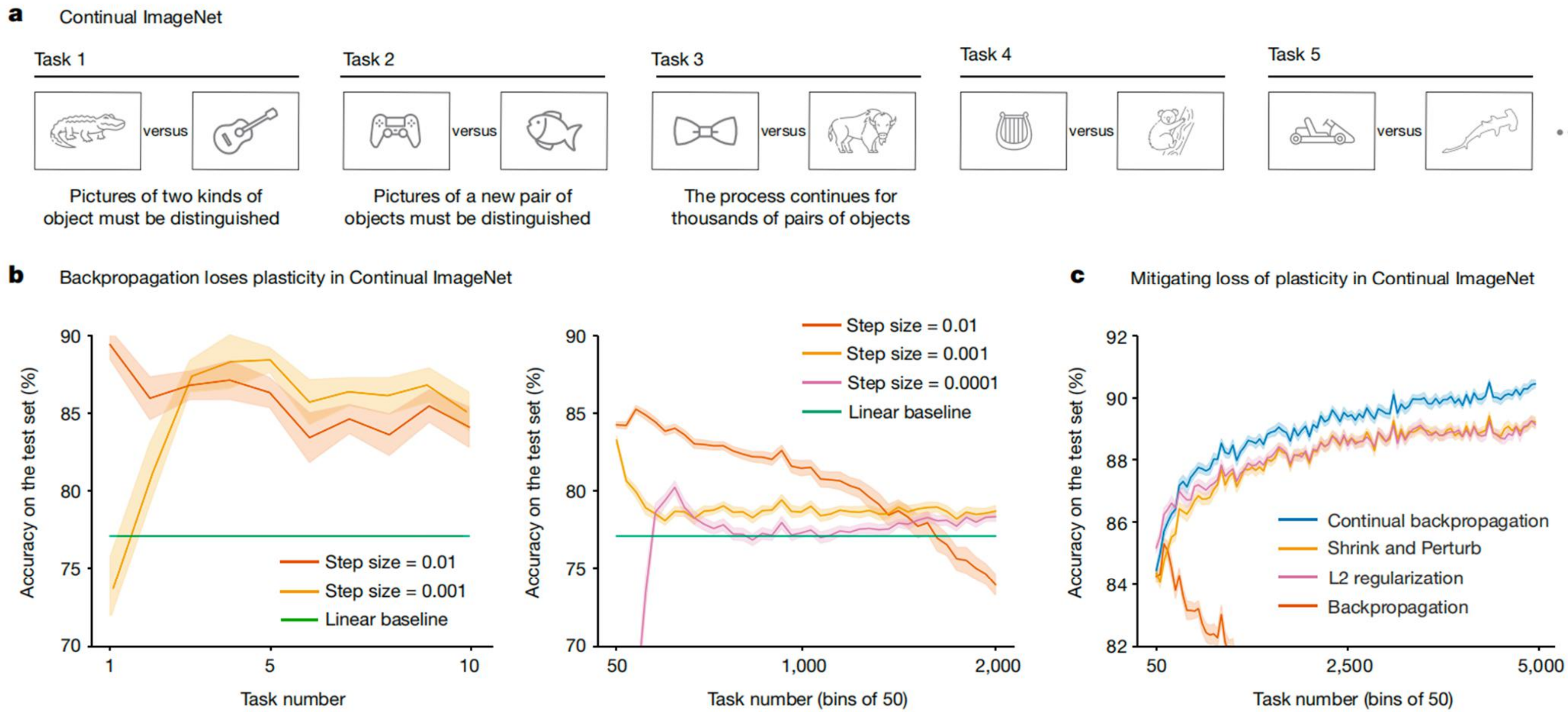
北京大學
PEKING UNIVERSITY

Loss of plasticity in deep continual learning

Shibhansh Dohare[1]✉, J. Fernando Hernandez-Garcia[1], Qingfeng Lan[1], Parash Rahman[1], A. Rupam Mahmood[1,2] & Richard S. Sutton[1,2]

Published online: 21 August 202
Nature
Cite: 19

Key idea Catastrophic Forgetting VS Plasticity



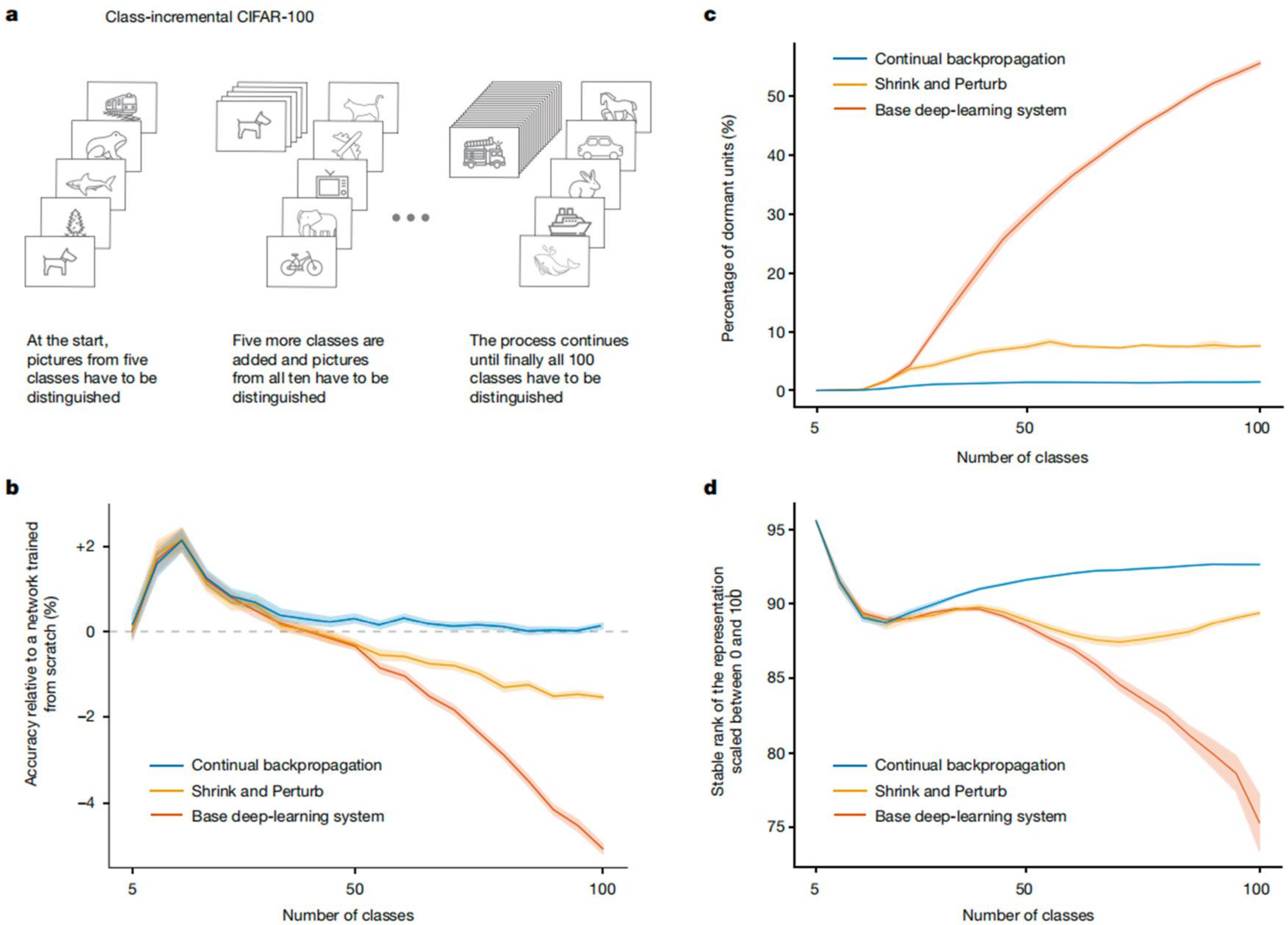
Imagenet上
5k个2分类

Convolutional network

Fig. 1 | Plasticity loss in Continual ImageNet. a–c, In a sequence of binary classification tasks using ImageNet pictures (**a**), the conventional backpropagation algorithm loses plasticity at all step sizes (**b**), whereas the continual backpropagation, L2 regularization and Shrink and Perturb algorithms maintain plasticity, apparently indefinitely (**c**). All results are averaged over 30 runs; the solid lines represent the mean and the shaded regions correspond to ±1 standard error.

Catastrophic Forgetting： lose previously learned knowledge
Plasticity ： ability to adapt

Phenomenon CIFAR-100，每次增加五个类别，和旧类别一起训，使用18层的ResNet



1. 图b纵坐标为相对于从scratch训的准确率
2. 图c纵坐标为不活跃单元的比例
3. 图d纵坐标为稳定秩（Stable rank）

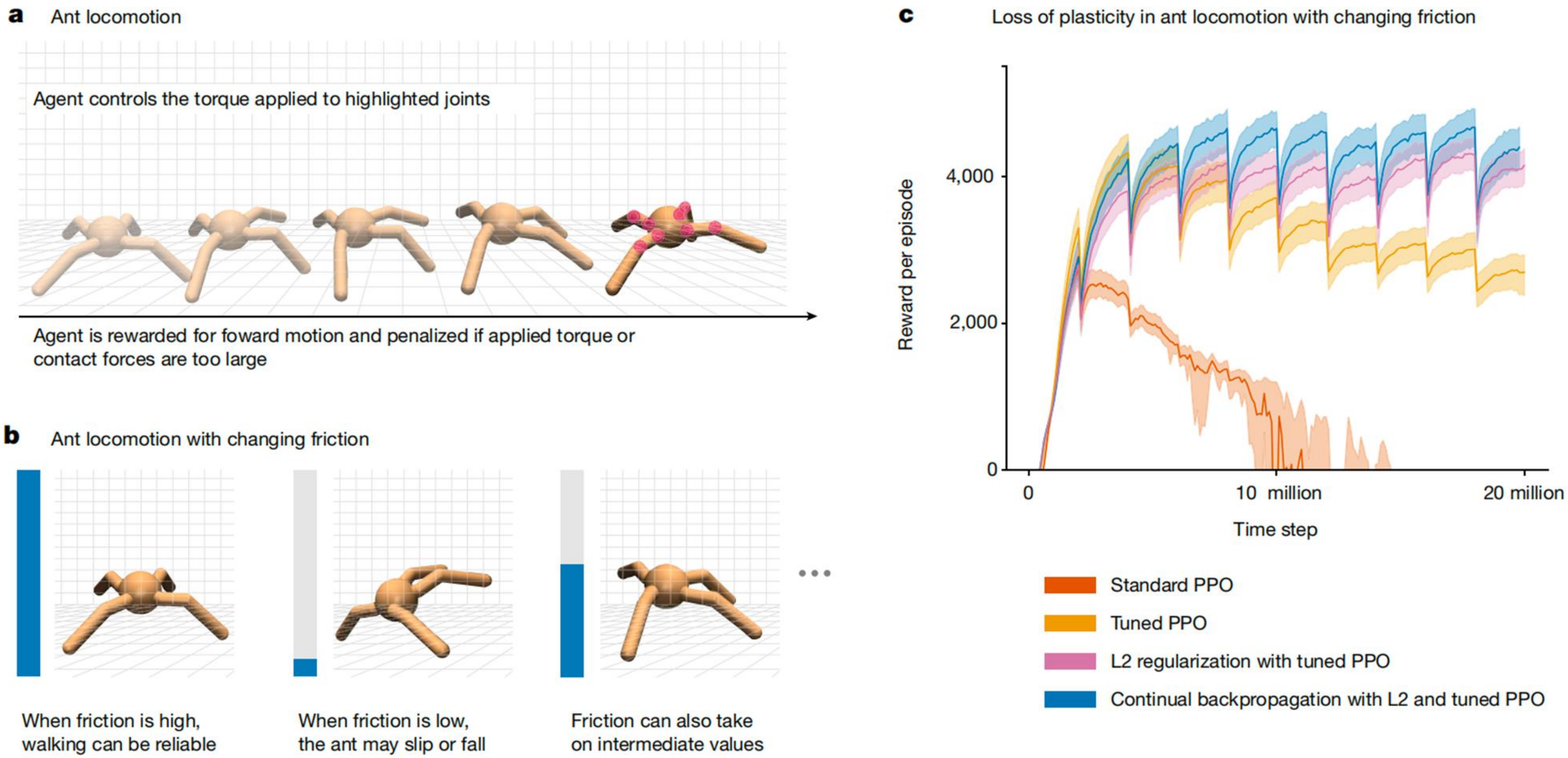
1. 网络学不动了，不活跃比例越来越高
2. 学习力/多样性越来越低

Coling的观点，rank越低越好
- 每个任务使用不同的lora进行cl
- 每个lora的rank都很低
- 碰撞低，遗忘率低

这篇的观点， rank越高越好
- 在同一个网络进行cl
- 总网络的rank越高，plasticity越好

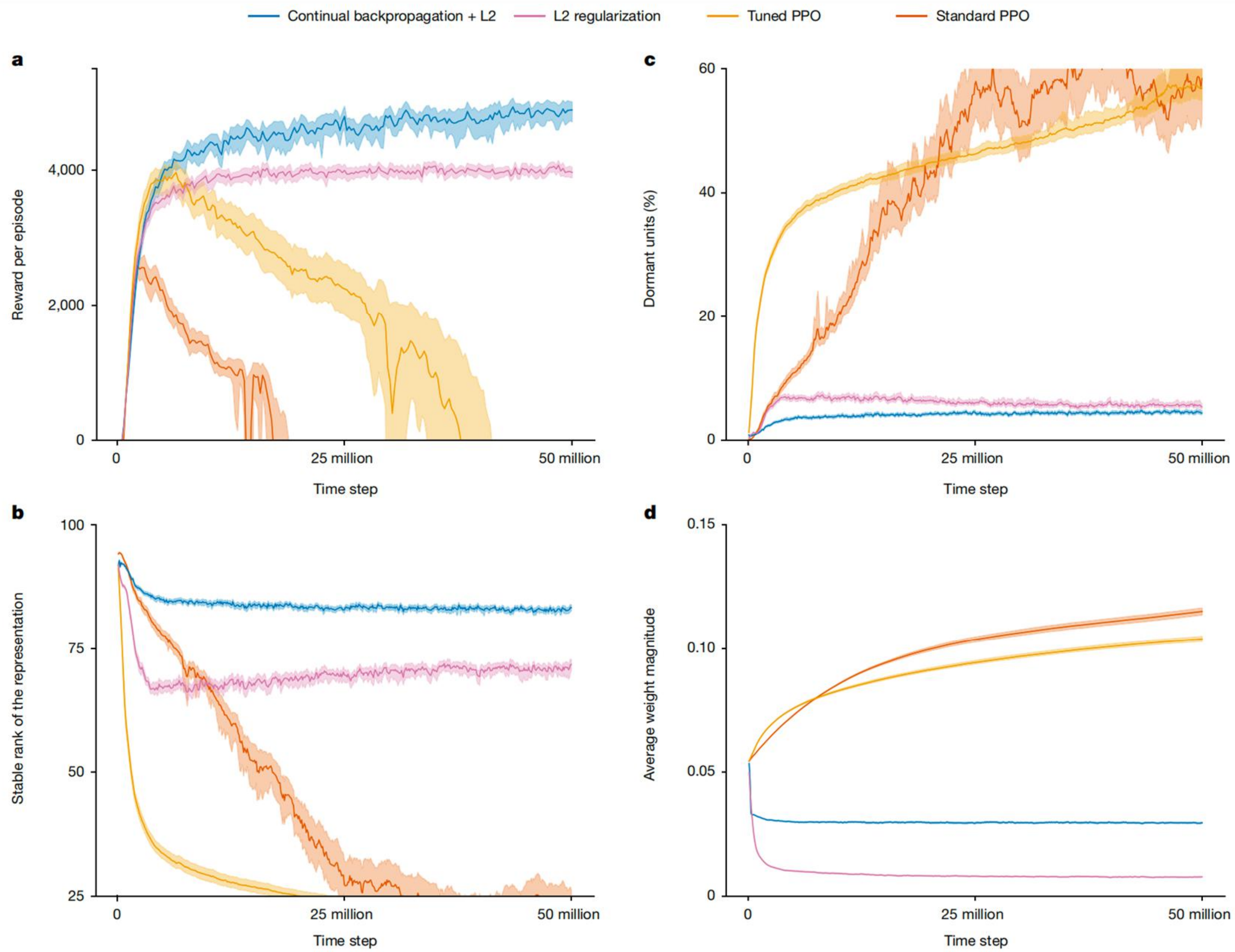
我们认为网络的rank是网络学到任务所用的子空间数，这篇认为是网络剩余的子空间数

Phenomenon



- 蚂蚁机器人任务，让机器人尽可能快速、有效地向前移动，摩擦系数会随时间变化
- 纵坐标为Reward per episode
- 标准的PPO和Tuned PPO在摩擦系数变化时表现出较大波动

Phenomenon



- 将摩擦系数保持在一个中间值，持续5000万步
- 图a: Reward per episode
- 图b: Stable rank of the representation
- 图c: Dormant units
- 图d: Average weight magnitude

1. 非增长性的权重
2. 网络中的持续变化

较小的权重可以减少可塑性丧失，持续的变化注入进一步减轻了可塑性丧失

Method

Contribution utility

$$\mathbf{u}_l[i] = \eta \times \mathbf{u}_l[i] + (1-\eta) \times |\mathbf{h}_{l,i,t}| \times \sum_{k=1}^{n_{l+1}} |\mathbf{w}_{l,i,k,t}|, \tag{1}$$

- reinitialize low-utility units
- the product of **units' activation and outgoing weight**
- 历史Contribution utility +当前Contribution utility

1. $u_l[i]$：第 l 层中第 i 个单元在时间 t 的Contribution utility
2. η：衰减率（decay rate）
3. $\mathrm{h}_{l,i,t}$：第 l层中第i 个单元在时间步t 的输出激活值
4. $\mathrm{w}_{l,i,k,t}$：第 i 个单元的输出权重总和，连接到第 l+1 层的所有单元

Selective Reinitialization

1. output weights **are initialized to zero**
2. threshold m： 为了防止某些单元在重新初始化后再次被初始化, 一个单元需要经历 m次更新后才会被视为“成熟”
3. replacement rate ρ : 每次更新只会重新初始化网络中**一小部分**单元，非常小，每层每200次更新才更新一个单元

Method

对于每一层:
1. 计算每一层的$u_l[i]$
2. 找到成熟的单元（更新次数 > m）
3. 用replacement rate ρ 计算更新单元数量
   1. 找到u_l最小的单元
   2. 初始化输入权重，对该单元的输入权重重新采样
   3. 初始化输入权重，对该单元的输出权重设定为0
   4. 要更新单元数 -1

Algorithm 1. Continual backpropagation for a feed-forward network with L layers

Set replacement rate ρ, decay rate η and maturity threshold m

Initialize the weights $\mathbf{w}_0,\ldots,\mathbf{w}_{L-1}$, in which $\mathbf{w}_l$ is sampled from distribution d_l

Initialize utilities $\mathbf{u}_1,\ldots,\mathbf{u}_{L-1}$, number of units to replace $c_1,\ldots,c_{L-1}$, and ages $\mathbf{a}_1,\ldots,\mathbf{a}_{L-1}$ to 0

For each input $\mathbf{x}_t$ **do**

Forward pass: pass $\mathbf{x}_t$ through the network to get the prediction $\widehat{\mathbf{y}}_t$

Evaluate: receive loss $l(\mathbf{x}_t, \widehat{\mathbf{y}}_t)$

Backward pass: update the weights using SGD or one of its variants

For layer l in $1:L-1$ **do**

Update age: $\mathbf{a}_l = \mathbf{a}_l + 1$

Update unit utility: see equation (1)

Find eligible units: n_{eligible} = number of units with age greater than m

Update number of units to replace: $c_l = c_l + n_{\text{eligible}} \times \rho$

If $c_l > 1$

Find the unit with smallest utility and record its index as r

Reinitialize input weights: resample $\mathbf{w}_{l-1}[:,r]$ from distribution d_l

Reinitialize output weights: set $\mathbf{w}_l[r,:]$ to 0

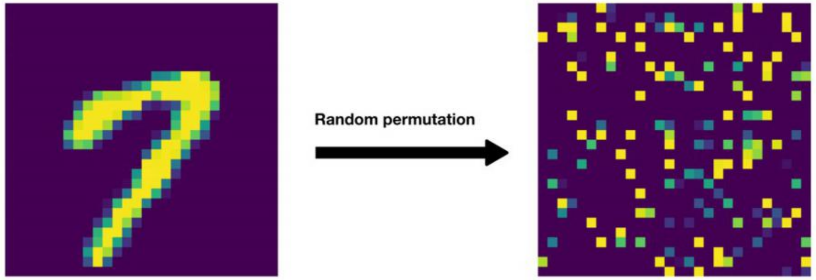
Reinitialize utility and age: set $\mathbf{u}_l[r] = 0$ and $\mathbf{a}_l[r] = 0$

Update number of units to replace: $c_l = c_l - 1$

End For

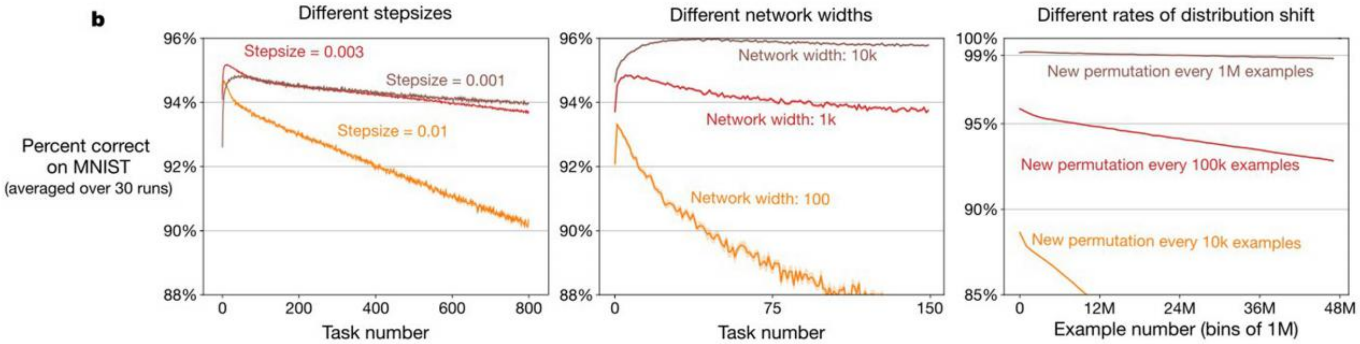
End For

Further analysis

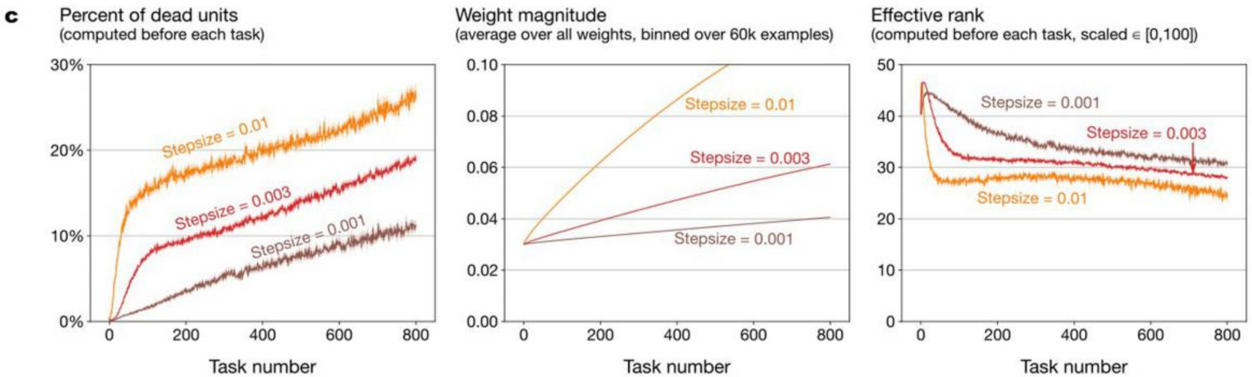


- Permuted MNIST
  - 通过对MNIST图片像素进行相同的随机置换
- 生成了800个不同的Permuted MNIST任务
- 全连接神经网络

- 存在可塑性丧失
  - 学习率
  - 网络规模
  - 任务变化速率



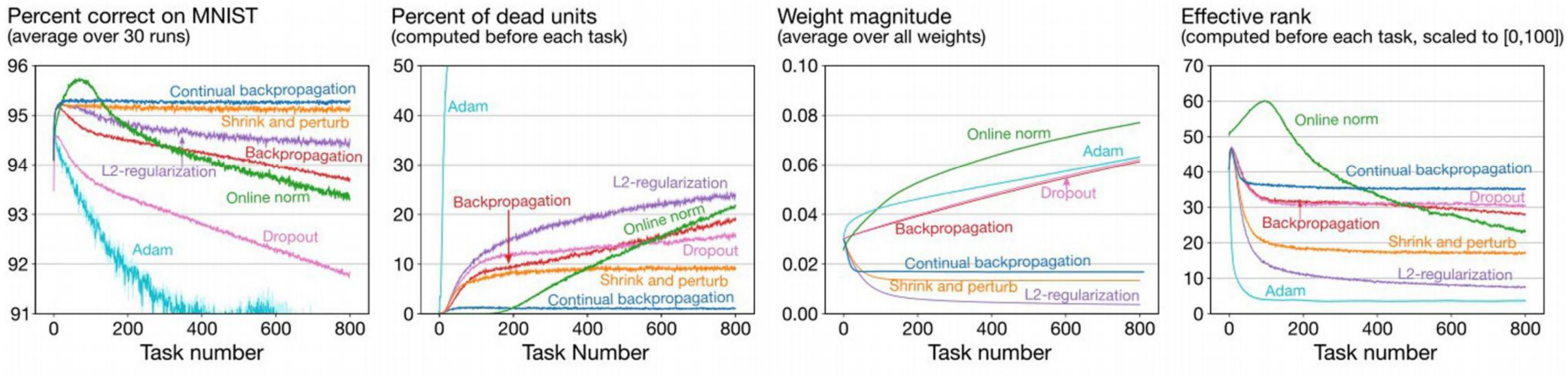
- 在训练过程中，改变的只有模型权重，初始随机分布具备许多促进可塑性的属性
  - 未饱和的权重单元
  - 权重幅值
  - 权重多样性
- 对应三个指标
  - **Percent of dead units**
  - **Weight magnitude [1]**
  - **Effective rank [2][3]**



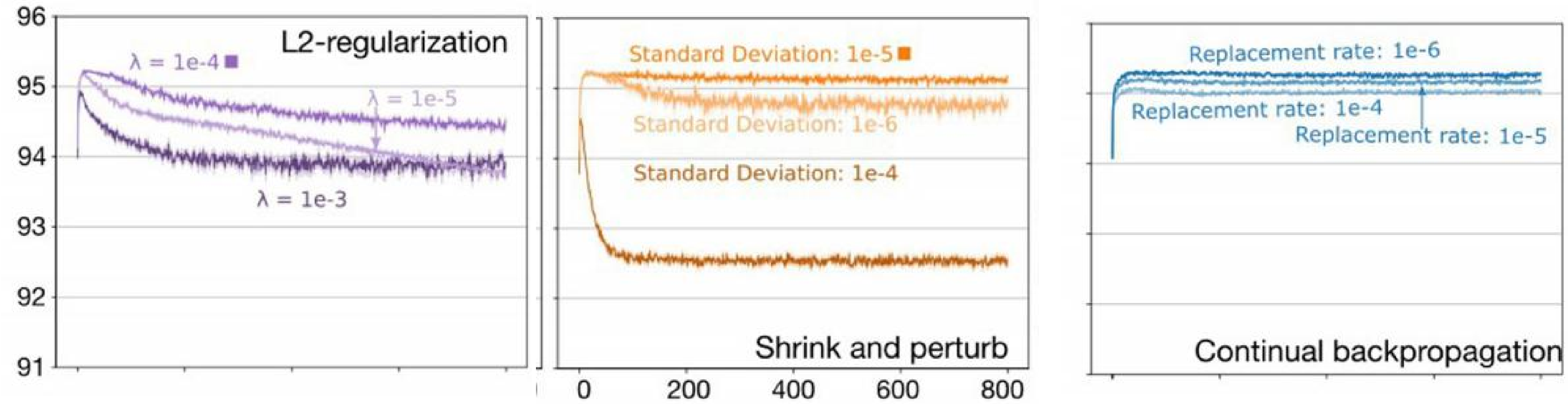
[1]Convex Optimization (Cambridge Univ. Press, 2004).

[2] On the origin of implicit regularization in stochastic gradient descent. (ICLR, 2021).
[3] Implicit regularization in deep learning may not be explainable by norms. (NIPS 2020).

Further analysis



- L2-regularization (紫色): 权重幅值不再增长，但是dead units的比例仍在增加，有效秩仍在下降
- Shrink and Perturb(黄色)：与L2-regularization类似， Shrink可以通过噪声注入随机性， Perturb可以控制权重幅值，看似很完美，但是效果没有他的好
- Dropout(粉色)，Online norm(绿色)，Adam(淡蓝色) 都不如他的(深蓝色)



他的参数敏感好

Disscussion

文章定义的Contribution utility 太简单了？

- 激活值和输出权重的乘积 而且只考虑了输入输出
- 考虑给定unit对整体表示函数的影响？ ([4]后续讲) 从损失函数反向传播考虑？或者传统网络剪枝有很多idea，能否复用？

重新初始化方式？

- 输入权重重新采样，输出权重为0
- 和彩票假说的方式不同([5]后续讲)，是否存在更好的重新初始化方式？传统动态稀疏训练中有没有更好的idea?

从和彩票假说关系？

- 有效秩会随着训练减少 == 比原始网络具有更少的随机性和多样性 == “中奖票”更少 == 可塑性丧失？
- 子网络可能逐渐被“固化”在之前任务上，不再具备对新任务的敏感性，彩票少了，网络中能够有效学习新任务的子网络越来越少，导致网络逐渐失去了处理新任务的能力

[4] Addressing Loss of Plasticity and Catastrophic Forgetting in Continual Learning(iclr24)
[5]The lottery ticket hypothesis: Finding sparse, trainable neural networks (Iclr 2019 oral)

Disscussion

L2正则和L1正则？为什么用L2不用L1?

- 在减少权重幅度上和秩，l1也可以(coling)，为什么本文用L2不用L1，之前投稿的用不用也补L2的实验？
- 在Pre-train上有和投稿那篇非常相似的文章，也使用了L1正则，也可以从彩票假设上解释([6]后续讲)？
- coling那篇，不断的加lora，按照这篇的理论，相当于在**低秩的低秩的(low-rank-lora)**空间内为每一个任务找到**不同的彩票(orthogonal and non-collision)**

LLM上也存在类似现象吗? PEFT上怎么玩?

- Llm也用基于梯度的优化，参数也会固化，难以调整
- 但是参数空间极大，怎么验证？(做几千次训练不现实)
- 在LoRA上？

[6] Finding the Dominant Winning Ticket in Pre-Trained Language Models(ACL 2022)

北京大學
PEKING UNIVERSITY

Addressing Loss of Plasticity and Catastrophic Forgetting in Continual Learning

Mohamed Elsayed
Department of Computing Science
University of Alberta
Alberta Machine Intelligence Institute
mohamedelsayed@ualberta.ca

A. Rupam Mahmood
Department of Computing Science
University of Alberta
CIFAR Canada AI Chair, Amii
armahmood@ualberta.ca

Iclr 2024

6663

Method

Useful units

$$U_{l,i,j}(Z) \doteq \mathcal{L}(\mathcal{W}_{\neg[l,i,j]}, Z) - \mathcal{L}(\mathcal{W}, Z),$$

1. Z ：样本
2. $U_{l,i,j}$ ：第 l 层中第 i 行，第j列权重的**Useful units**

- 某个特定的权重被移除或置为0时，模型损失的精确变化
- 计算模型的反事实损失（将权重设为0后的损失）减去当前模型的实际损失

$$\begin{aligned} U_{l,i,j}(Z) &= \mathcal{L}(\mathcal{W}_{\neg[l,i,j]}, Z) - \mathcal{L}(\mathcal{W}, Z) \\ &\approx \mathcal{L}(\mathcal{W}, Z) + \frac{\partial \mathcal{L}(\mathcal{W}, Z)}{\partial W_{l,i,j}}(0 - W_{l,i,j}) + \frac{1}{2}\frac{\partial^2 \mathcal{L}}{\partial W_{l,ij}^2}(0 - W_{l,i,j})^2 - \mathcal{L}(\mathcal{W}, Z) \\ &= -\frac{\partial \mathcal{L}(\mathcal{W}, Z)}{\partial W_{l,i,j}} W_{l,i,j} + \frac{1}{2}\frac{\partial^2 \mathcal{L}(\mathcal{W}, Z)}{\partial W_{l,i,j}^2} W_{l,i,j}^2. \end{aligned} \quad (2)$$

- 第一项：权重$W_{l,i,j}$对损失的线性影响
- 第二项：权重$W_{l,i,j}$对对损失的二次影响，可以近似计算

属于之前提到的**Shrink and Perturb**方式，用这个方式来更新梯度

$$w_{l,i,j} \leftarrow \rho w_{l,i,j} - \alpha\left(\frac{\partial \mathcal{L}}{\partial w_{l,i,j}} + \xi\right)\left(1 - \bar{U}_{l,i,j}\right)$$

Method

Algorithm 1 UPGD

Given a stream of data $\mathcal{D}$, a network f with weights $\{\boldsymbol{W}_1, ..., \boldsymbol{W}_L\}$.
Initialize step size α, utility decay rate β, and noise standard deviation σ.
Initialize $\{\boldsymbol{W}_1, ..., \boldsymbol{W}_L\}$.
Initialize $\boldsymbol{U}_l, \forall l$ and time step t to zero.
for $(\boldsymbol{x}, \boldsymbol{y})$ in $\mathcal{D}$ **do**
  $t \leftarrow t + 1$
  for l in $\{L, L-1, ..., 1\}$ **do**
    $\eta \leftarrow -\infty$
    $\boldsymbol{F}_l, \boldsymbol{S}_l \leftarrow \texttt{GetDerivatives}(f, \boldsymbol{x}, \boldsymbol{y}, l)$
    $\boldsymbol{M}_l \leftarrow {}^1\!/{}_2 \boldsymbol{S}_l \circ \boldsymbol{W}_l^2 - \boldsymbol{F}_l \circ \boldsymbol{W}_l$
    $\boldsymbol{U}_l \leftarrow \beta \boldsymbol{U}_l + (1-\beta)\boldsymbol{M}_l$
    $\hat{\boldsymbol{U}}_l \leftarrow \boldsymbol{U}_l / (1 - \beta^t)$
    if $\eta < \texttt{max}(\hat{\boldsymbol{U}}_l)$ **then** $\eta \leftarrow \texttt{max}(\hat{\boldsymbol{U}}_l)$
  for l in $\{L, L-1, ..., 1\}$ **do**
    Sample $\boldsymbol{\xi}$ elements from $\mathcal{N}(0, \sigma^2)$
    $\bar{\boldsymbol{U}}_l \leftarrow \phi(\hat{\boldsymbol{U}}_l / \eta)$
    $\boldsymbol{W}_l \leftarrow \boldsymbol{W}_l - \alpha(\boldsymbol{F}_l + \boldsymbol{\xi}) \circ (1 - \bar{\boldsymbol{U}}_l)$

计算**Useful units**
并衰减

从后往前进行
Shrink and Perturb

提出了度量Plasticity的Metric

$$p(Z) = \max\left(1 - \frac{\mathcal{L}(\mathcal{W}^\dagger, Z)}{\max(\mathcal{L}(\mathcal{W}, Z), \epsilon)}, 0\right) \in [0, 1]$$

- W' 是更新后的权重， W是原本的权重
- p(Z) = 0：更新后损失无变化，模型没有任何进步，（低可塑性）
- p(Z) = 1：更新后损失极小，对样本 Z 完全更新并提升了预测能力（高可塑性）

北京大學
PEKING UNIVERSITY

The lottery ticket hypothesis: Finding sparse, trainable neural networks

Jonathan Frankle
MIT CSAIL
jfrankle@csail.mit.edu

Michael Carbin
MIT CSAIL
mcarbin@csail.mit.edu

Iclr 2019 oral
Cite: 3918

Method

The Lottery Ticket Hypothesis. *A randomly-initialized, dense neural network contains a subnetwork that is initialized such that—when trained in isolation—it can match the test accuracy of the original network after training for at most the same number of iterations.*

Strategy 1: Iterative pruning with resetting.

1. Randomly initialize a neural network $f(x; m \odot \theta)$ where $\theta = \theta_0$ and $m = 1^{|\theta|}$ is a mask.
2. Train the network for j iterations, reaching parameters $m \odot \theta_j$.
3. Prune $s\%$ of the parameters, creating an updated mask m' where $P_{m'} = (P_m - s)\%$.
4. Reset the weights of the remaining portion of the network to their values in θ_0. That is, let $\theta = \theta_0$.
5. Let $m = m'$ and repeat steps 2 through 4 until a sufficiently pruned network has been obtained.

- 网络修剪之后，剩余的节点使用**初始化权重**，而不是重新随机化权重。（单靠网络结构是不能表明这种假设的的有效性的，还需要初始化的参数）
    - 从一开始训练一个已经剪枝的网络，效果通常不如**剪枝后重置初始权重**再重新训练，因为剪枝后的网络需要与初始权重的幸运初始化相结合才能有效训练。
- 密集网络比稀疏网络更容易训练，包含更多可能的“中奖票”

北京大學
PEKING UNIVERSITY

Finding the Dominant Winning Ticket in Pre-Trained Language Models

Zhuocheng Gong[1], Di He[2], Yelong Shen[3], Tie-yan Liu[2], Weizhu Chen[3], Dongyan Zhao[1,4,5*], Ji-rong Wen[6] and Rui Yan[6*]

[1]Wangxuan Institute of Computer Technology, Peking University, China
[2]Microsoft Research, Beijing, China, [3]Microsoft Azure AI
[4]Artificial Intelligence Institute of Peking University
[5]State Key Laboratory of Media Convergence Production
[6]Gaoling School of Artificial Intelligence, Renmin University of China
{gzhch, zhaody}@pku.edu.cn, {jrwen, ruiyan}@ruc.edu.cn
{dihe, yelong.shen, tyliu, wzchen}@microsoft.com

ACL 2022

Method

$$\min_{\boldsymbol{\theta}} \; L(\mathcal{D}, f, \boldsymbol{\theta}) + \lambda ||\boldsymbol{\theta} - \boldsymbol{\theta}_0||_1, \quad (1)$$

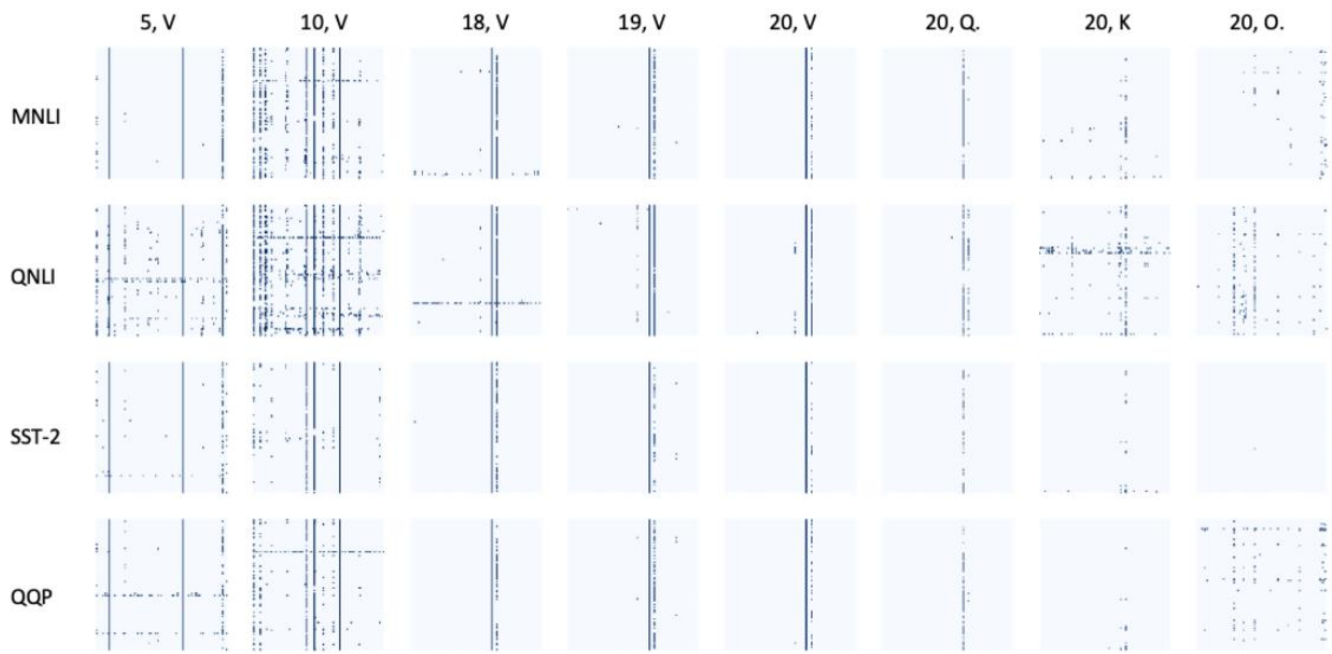
$$\boldsymbol{\theta}_\sigma = \boldsymbol{m}_\sigma \odot \boldsymbol{\theta}, \quad \boldsymbol{m}_\sigma \in \{0, 1\}^{|\boldsymbol{\theta}|} \quad (2)$$

Algorithm 1 Extracting the dominant winning ticket

1: Fine-tune a PLM $f(x; \theta_0)$ with L1 regularization on any dowmstream task dataset $\mathcal{D}$, get $f(x; \theta)$.

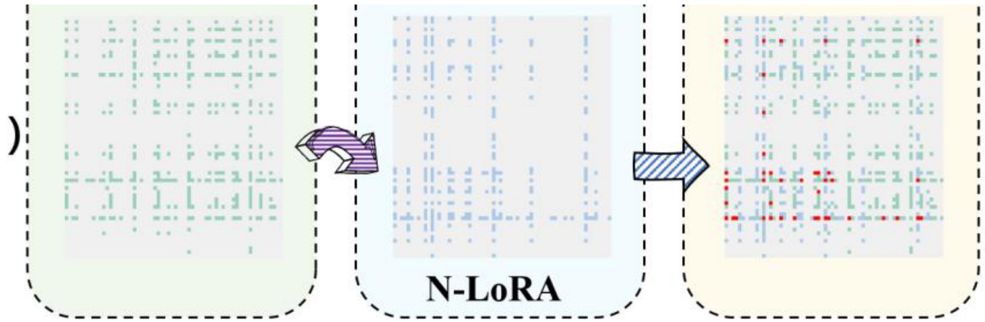
2: Calculate $\Delta\boldsymbol{\theta} = \boldsymbol{\theta} - \boldsymbol{\theta}_0$, then select out out weighted parameters $\boldsymbol{\theta}_\sigma$ with threshold σ.

3: Select the k most dominated dimensions each matrix, which forms the dominant winning ticket.



- 在Robert-large上进行的
- 使用L1后，选择那些彩票的维度作为彩票子网络，彩票子网络具有相似性
- 只用在剪枝微调上，没有拓展到CL

Method



$$L = L_{\text{task}} + L_{\text{sparse}} = L_{t_i} + \lambda \|\Delta W_i\|_1 \quad (4)$$