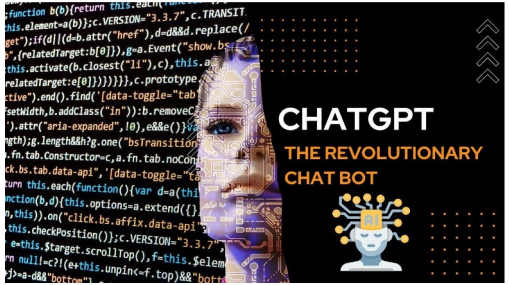
DOES CONTINUAL LEARNING EQUALLY FORGET ALL PARAMETERS?

Background

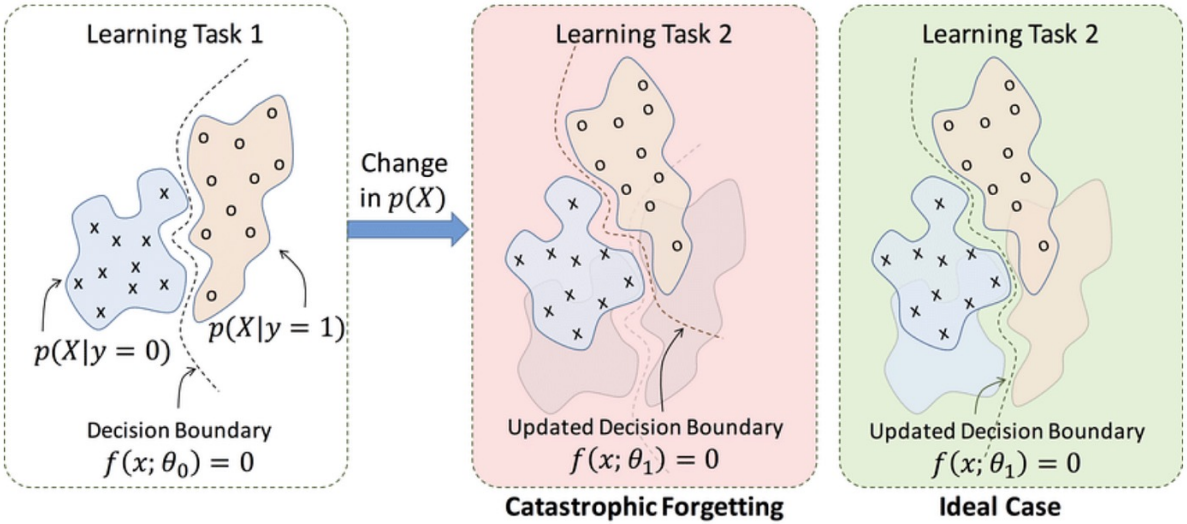
- Models are very strong
  - Writing, translating, programming



However, the model can’t increment learning new task/knowledge

Catastrophic Forgetting

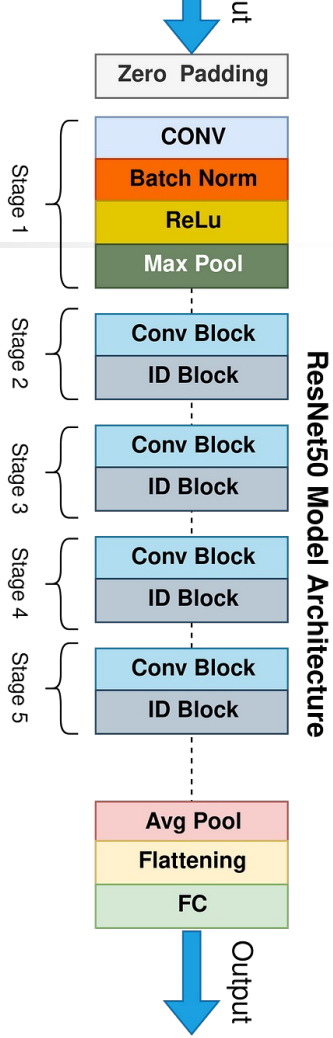
Model reliability and performance degenerate drastically in continual learning (CL) where the data distribution or task in training changes over time, as the model quickly adapts to a new task and overwrites the previously learned weights.



In this paper, They study fundamental but open problem in CL

Background

- Resnet-18 structure
  - Convolution layer
  - Batch norm
  - Convolution block
  - FC layer (fully connected layers)



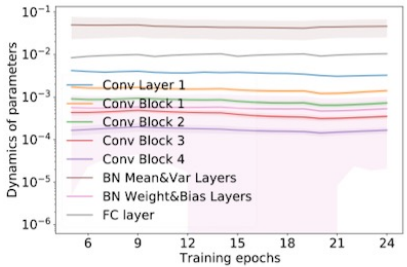
Open Problem

- **Which parts** of networks are more **task-specific** and sensitive to task changes?
- Is the catastrophic forgetting mainly caused by the sensitively changing parameters?
- Can we mitigate catastrophic forgetting by **only finetuning sensitive parameters**?
- To achieve promising performance, how many parameters should be finetuned?
- Is every-step replay necessary and can we replace every-step replay with occasional FT?

An EMPIRICAL STUDY

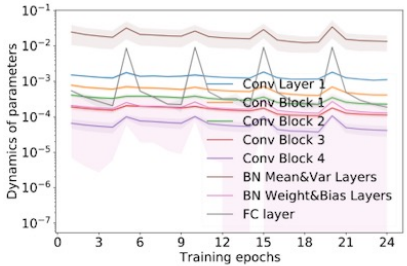
(1) Measuring Forgetting via Training Dynamics

Dynamics of training ResNet-18 by SGD on Seq-CIFAR-10 with buffer size of 500



(a) Dynamics between consecutive **tasks**

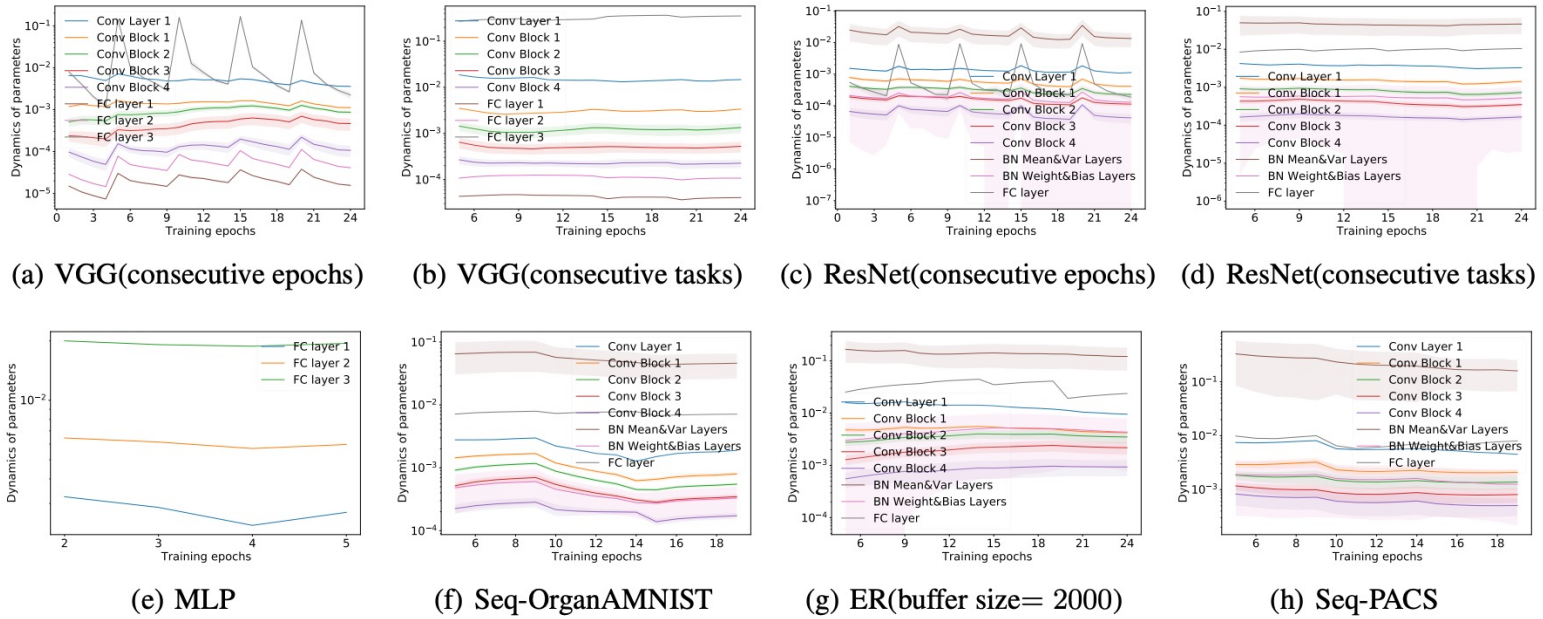
$$\left(\frac{1}{|\theta_l|}\right)\left|\left|\theta^t_{l,n} - \theta^t_{l,n-1}\right|\right|_1 \quad (1)$$



(b) Dynamics between consecutive **epochs**

$$\left(\frac{1}{|\theta_l|}\right)\left|\left|\theta^{t+1}_{l,n} - \theta^t_{l,n}\right|\right|_1 \quad (2)$$

An EMPIRICAL STUDY



(a) VGG(consecutive epochs) (b) VGG(consecutive tasks) (c) ResNet(consecutive epochs) (d) ResNet(consecutive tasks)

(e) MLP (f) Seq-OrganAMNIST (g) ER(buffer size= 2000) (h) Seq-PACS

Figure 1. (a-e) The training dynamics of different metrics for different groups of parameters when applying SGD in CL to train three types of deep neural networks on Seq-CIFAR-10; The training dynamics of other scenarios in ResNet-18: (e) on a non-standard dataset; (g) using a different CL method with a different buffer size; (h) in the domain-IL setting. Note the y-axis is of logarithmic scale.

Observations

- When BN is included in the network, BN layers' mean and variance become the most sensitive changing parameters
- In class-incremental learning. the last FC layer is more sensitive than other parameters.
- The sensitivity of convolutional lavers increases as the layer gets closer to the input.

Insights

- Only a small portion of parameters are much more sensitive than others. This implies that only finetuning these task-specific parameters may suffice to retain the previous tasks.
- The dynamics between consecutive epochs show that all laveys experience more changes when tasks shift. Which can be used to detect task boundaries during CL.

Forgetting Prioritized Finetuning

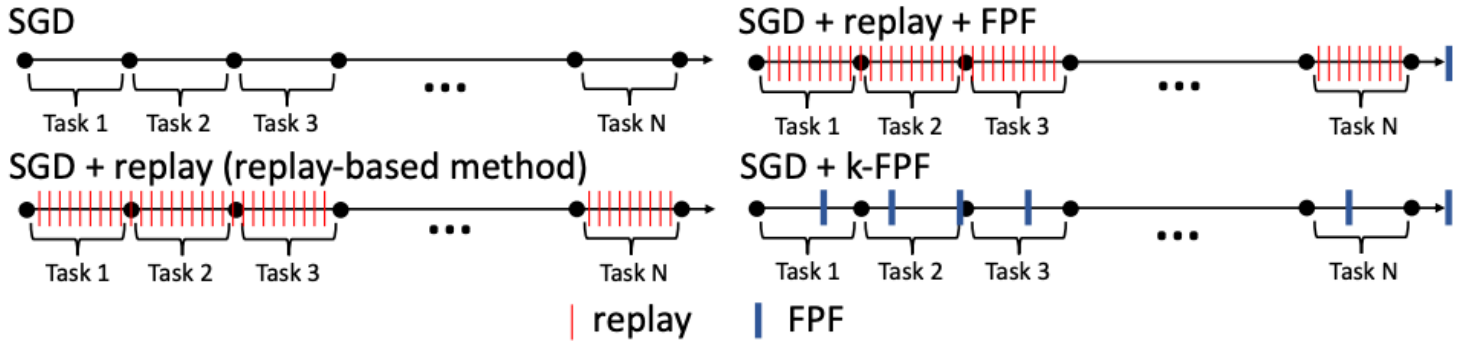


Figure 2. Comparison of SGD, replay-based method, FPF, and k-FPF. SGD trains tasks sequentially without replay. Replay-based methods train models on buffered and current data simultaneously. FPF finetunes the most sensitive parameters for a few iterations using buffered data at the end of arbitrary CL methods. k-FPF periodically (regardless of task boundaries) applies FPF for k times during training.

Selection of Sensitive Parameters

The ranking of sensitivity for different layers does not change. They select sensitive parameters according to dynamics in the early epochs

Sensitivity score: $$S_g = \frac{(1/|g|)\sum_{l \in g} c_l}{\sum_{g=1}^{G}(1/|g|)\sum_{l \in g} c_l} * \mathrm{G}.$$

Comparison with SOTA CL

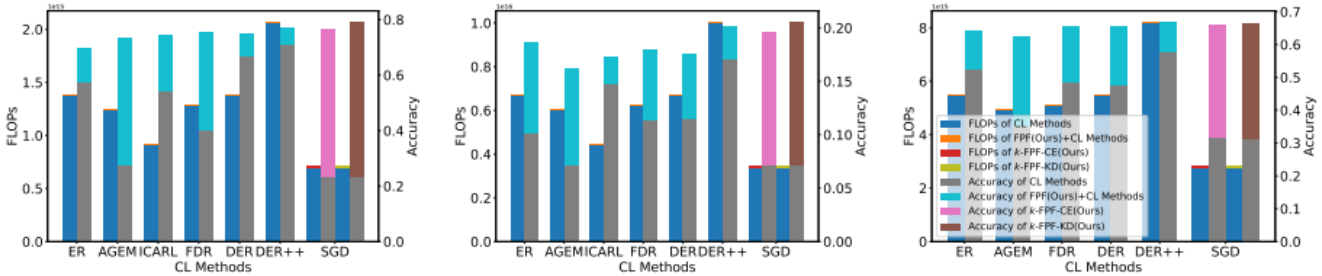
Buffer	Methods	Class-IL Seq-OrganAMNIST	Class-IL Seq-PathMNIST	Class-IL Seq-CIFAR-10	Class-IL Seq-Tiny-ImageNet	Domain-IL Seq-PACS
	JOINT	91.92±0.46	82.47±2.99	81.05±1.67	41.57±0.55	70.85±8.90
	SGD	24.19±0.15	23.65±0.07	19.34±0.06	7.10±0.14	31.43±6.39
	oEWC (Schwarz et al., 2018)	22.71±0.67	22.36±1.18	18.48±0.71	6.58±0.12	35.96±4.59
200	GDUMB (Prabhu et al., 2020)	61.78±2.21	46.31±5.64	30.36±2.65	2.43±0.31	34.16±3.45
	k-**FPF-CE**	75.21±2.03	72.88±3.22	57.97±1.53	13.76±0.72	60.70 ±2.81
	k-**FPF-KD**	80.32±1.16	**74.68±4.72**	58.50±1.03	14.74±0.94	63.15±1.19
	ER (Riemer et al., 2018)	71.69±1.71	51.66±5.86	45.71±1.44	8.15±0.25	51.53±5.10
	FPF+ER	76.92±2.26	67.34±2.68	57.68±0.76	13.08±0.65	65.16±1.97
	AGEM (Chaudhry et al., 2018)	24.16±0.17	27.93±4.24	19.29±0.04	7.22±0.15	40.54±3.43
	FPF+AGEM	72.22±2.45	66.88±3.05	55.33±2.19	12.27±0.49	57.33±0.76
	iCaRL (Rebuffi et al., 2017)	79.61±0.56	54.35±0.94	59.60±1.06	12.13±0.20	-
	FPF+iCaRL	80.28±0.58	71.20±2.19	**63.36±0.91**	**16.99±0.37**	-
	FDR (Benjamin et al., 2018)	68.29±3.27	44.27±3.20	41.77±4.24	8.81±0.19	45.91±3.54
	FPF+FDR	76.10±0.87	70.06±2.78	51.91±2.77	11.52±0.72	57.17±1.31
	DER (Buzzega et al., 2020)	73.28±1.33	54.45±5.92	47.04±3.03	9.89±0.58	46.93±4.94
	FPF+DER	79.63±1.21	67.29±3.75	56.67±2.19	12.65±0.60	61.49±1.37
	DER++ (Buzzega et al., 2020)	78.22±2.05	62.00±3.79	59.13±0.81	12.12±0.69	55.75±2.02
	FPF+DER++	**80.99±0.91**	68.78±2.99	61.69±0.97	13.72±0.40	**65.28±1.02**

Comparison with SOTA CL

- FPF considerably improves the performance of memory-based CL methods
- k-FPF-SGD is better than the best CL methods.
- k-FPF-KD (with additional knowledge distillation loss) further improves the performance of k-FPF-SGD to be comparable to FPF.

Online Continual Learning

- FLOPS of SGD without every-step replay is only about half of other CL methods.
- The extra FLOPS (red) required by FPF. k-FPF-SGD and k-FPF-KD are almost negligible compared to the training FLOPS.



(a) Seq-PathMNIST (b) Seq-Tiny-ImageNet (c) Seq-PACS

Figure 3. Comparison of FLOPs and accuracy between FPF, k-FPF and SOTA methods. **FPF improves all CL methods by a large margin without notably extra computation.** **k-FPF consumes much less computation but achieves comparable performance as FPF.** A large and clear version can be found in Appendix. M.

Thank you