An Empirical Investigation of the Role of Pre-training in Lifelong Learning

Sanket Vaibhav Mehta [1] **Darshan Patil** [2 3] **Sarath Chandar** [2 4 5] **Emma Strubell** [1]

Abstract

Catastrophic forgetting is a key challenge to the lifelong learning paradigm in machine learning, which is an attractive alternative to the more prominent isolated learning scheme not only due to its resemblance to biological learning, but also its potential to reduce energy waste by obviating excessive model re-training. Recently, various approaches have been proposed to mitigate catastrophic forgetting in neural networks. However, our understanding of the efficacy of these approaches in practice is limited for the following reasons: They typically study randomly initialized networks instead of networks with pre-trained initializations, rarely experiment with large networks (such as BERT), and seldom evaluate on a diverse set of tasks. In response, we investigate existing methods in the context of large, pre-trained models and evaluate their performance on diverse text and image classification tasks. Across all settings, we observe that generic pre-training implicitly alleviates the effects of catastrophic forgetting when learning multiple tasks sequentially compared to randomly initialized models. We further study this phenomenon by analyzing the loss landscape and show that pre-trained weights implicitly ease forgetting because they lead to wider minima for tasks. We also analyze how different pre-training initializations affect forgetting by conducting a large-scale study on a novel dataset of 15 diverse NLP tasks. We conclude that performance depends on both model capacity and qualities of the pre-training corpora.

1. Introduction

The contemporary machine learning paradigm concentrates on isolated learning (Chen & Liu, 2018) i.e., learning a model from scratch for every new task. In contrast, the lifelong learning (LL) paradigm (Thrun, 1996; Chen & Liu, 2018; Parisi et al., 2019) defines a biologically inspired learning approach where models learn tasks in sequence, ideally preserving past knowledge and leveraging it to efficiently learn new tasks. LL has the added benefit of avoiding periodical re-training of models from scratch to learn novel tasks or adapt to new data, with the potential to reduce both computational and energy requirements (Hazelwood et al., 2018; Strubell et al., 2019; Schwartz et al., 2020). In the context of modern machine learning where state-of-the-art models are powered by deep neural networks, *catastrophic forgetting* has been identified as a key challenge to implementing successful LL systems (McCloskey & Cohen, 1989; French, 1999). Catastrophic forgetting is when deep networks forget knowledge learned in previous tasks as information relevant to the current task is incorporated, and mitigating this phenomenon is where much of the previous work in LL has been focused.

At the same time, transfer learning has recently shown impressive results in both computer vision (CV) and natural language processing (NLP).[1] Since the introduction of ImageNet (Deng et al., 2009), the idea of learning generic representations and transferring them to other tasks has became ubiquitous. In this paradigm, representations pre-trained on ImageNet are fine-tuned on downstream tasks, resulting in lower sample complexity and higher overall performance. The field of NLP has followed suit, first with pre-trained word embeddings (Pennington et al., 2014; Mikolov et al., 2013), and later large language models (Devlin et al., 2019; Peters et al., 2018; Howard & Ruder, 2018).

Despite the tremendous success of generic initializations for transfer learning, little is known about their impact on LL settings. In CV, most previous work starts with randomly initialized models when developing LL algorithms. On the other hand, in NLP, practically all work now develops algorithms based on large pre-trained models. Whereas starting from a random initialization may provide a more challenging learning environment, it is simply not practical in the context of deploying state-of-the-art systems for real-world tasks. To the best of our knowledge, there has been no work

[1]Carnegie Mellon University, USA [2]Mila - Quebec AI Institute, Canada [3]University of Montreal, Canada [4]École Polytechnique de Montréal, Canada [5]Canada CIFAR AI Chair. Correspondence to: Sanket Vaibhav Mehta <svmehta@cs.cmu.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

[1]One of the original motivations for transfer learning was discussed as a way to enable lifelong learning, in a NIPS-95 workshop on "Learning to Learn" (Pan & Yang, 2009).

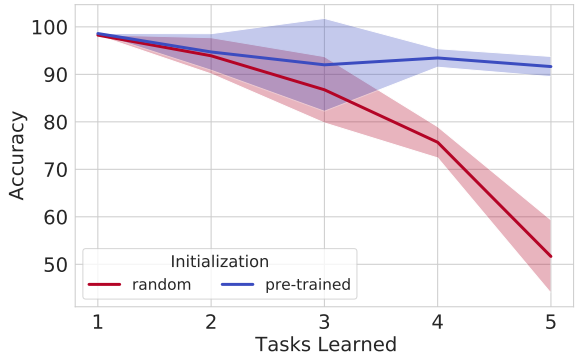


Figure 1: Pre-trained and randomly initialized DistilBERT on Split YahooQA dataset. Performance of the first task visualized over sequential learning of tasks (averaged across 5 runs). **Pre-trained initialization results in significantly less forgetting.**

systematically analyzing the effect of generic pre-trained initialization on catastrophic forgetting in LL scenarios.

Figure 1 shows that simply changing the network initialization to pre-trained weights can significantly reduce forgetting on the first task when doing sequential training on five tasks. This observation motivates us to ask "Does pre-training implicitly alleviate catastrophic forgetting?". To answer this question we conduct a systematic study on existing CV and NLP benchmarks and observe that pre-training indeed leads to less forgetting, and we hypothesize that pre-trained weights already have a good inductive bias to implicitly alleviate forgetting. To explain this behavior we build upon two separate lines of recent works—Hao et al. (2019) and Neyshabur et al. (2020) show that in the context of transfer learning, pre-trained weights lead to a flat basin in the loss landscape when fine-tuning on a single task. Mirzadeh et al. (2020b) argues that the geometric properties of the local minima found for each task play an important role in forgetting, and they propose to modify the training regime (learning rate decay, batch size, dropout) to widen the tasks' local minima.

To verify the above hypothesis, we analyze the loss landscape of the first task while the model is training incrementally on subsequent tasks. For pre-trained initialization, we see that minima obtained after training on a sequence of tasks still remain in the relatively low loss contour of the first task when compared with random initialization. Further, tracking the loss along the linear interpolation between the first task's minima and subsequent ones confirms that models initialized with pre-trained weights undergo a more gradual change in the loss compared to randomly initialized weights. These observations hint at the flatness of the minima reached in the case of pre-trained initialized models.

To quantify the flatness of the loss landscape, we evaluate the sharpness metric (Keskar et al., 2017) and verify that pre-trained weights indeed lead to flat basins in comparison to random weights while training sequentially. These analyses help us showcase that continual training from pre-trained weights induces wide task minima, which is shown to alleviate forgetting (Mirzadeh et al., 2020b).

We also investigate the effect of the type of pre-trained initialization by analyzing the extent to which four pre-trained transformer language model variants (Sanh et al., 2019; Devlin et al., 2019; Liu et al., 2019) undergo forgetting. On an existing benchmark spanning 5 diverse NLP tasks (de Masson d'Autume et al., 2019), we observe that increasing the capacity of the model and diversity of the pre-training corpus play an important role in alleviating forgetting. To further stress-test these models on a large number of diverse tasks, we introduce a dataset with 15 diverse NLP tasks and observe that forgetting becomes more prominent in this setting across all four models.

Our main contributions can be summarized as follows:

- We observe that initializing models with generic pre-trained weights results in less forgetting compared to random weights despite achieving higher performance on each task. To bolster this observation, we perform a systematic study and validate that this behavior persists across applications (NLP and CV) and in the context of two existing approaches: Elastic weight consolidation (Kirkpatrick et al., 2017) and experience replay (Chaudhry et al., 2019). We note that sequential training on diverse tasks is still challenging for models initialized with pre-trained weights.

- We examine the above-mentioned behavior from the loss landscape perspective. We hypothesize and empirically verify that pre-trained models alleviate forgetting as they have an implicit bias towards wider task minima. The effect of these wider minima is that changes in weights from learning subsequent tasks results in a smaller change to the current task loss, which helps reduce forgetting.

- To understand the role of varying pre-trained initializations, we analyse a suite of pre-trained Transformer language models and showcase that model capacity and diversity of the pre-training corpus do play a role in alleviating forgetting.

- For the pre-training initialization study, we introduce a new benchmark for lifelong learning in NLP consisting of 15 diverse NLP tasks, which proves more challenging than previous settings for the Transformer models considered in our study.

2. Related Work

Transfer learning from generic pre-trained models has enabled significant recent progress in ML (Zhuang et al., 2021). This trend started in the CV field with the ImageNet dataset (Deng et al., 2009). Transfer learning in NLP has witnessed its own "ImageNet revolution" where large models pre-trained on self-supervised tasks have shown impressive results across many language understanding tasks (Peters et al., 2018; Howard & Ruder, 2018; Radford et al., 2018; Devlin et al., 2019; Raffel et al., 2019; Liu et al., 2019).

Lifelong learning approaches focus on mitigating the catastrophic forgetting phenomenon and can be categorized into three groups: (1) *Regularization-based* approaches augment the loss function with extra penalty terms preventing important parameters learned on previous tasks from significantly deviating while training on the new task (Kirkpatrick et al., 2017; Zenke et al., 2017); (2) *memory-based* approaches augment the model with episodic memory for sparse experience replay of previous task examples (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2018; Wang et al., 2020) (3) *network expansion-based* approaches dynamically expand the network based upon new tasks (Rusu et al., 2016; Aljundi et al., 2017; Sodhani et al., 2020). We consider regularization and memory-based approaches in this work.

Meta-learning involves creating models that learn to learn over time. Several works propose meta-learning-based approaches for LL (Riemer et al., 2019; Finn et al., 2019; Javed & White, 2019; Wang et al., 2020). Caccia et al. (2020) propose a two-phase continual learning scenario where the first phase is pre-training (using MAML (Finn et al., 2017)) and the second phase involves continual deployment with task revisiting. They make the point that in many scenarios (Lomonaco et al., 2019), it would be unrealistic to deploy agents with no pre-training in a LL setting. Whereas some of these works do use pre-trained initializations for their models, many do not, and none have extensively studied the effect of pre-training on alleviating catastrophic forgetting.

Optimization and loss landscape. Hao et al. (2019) show that for single-task generalization, pre-training leads to wider optima for BERT models. Keskar et al. (2017) explore how larger batch sizes lead to sharper minima and worse generalization in the single-task learning setting. Mirzadeh et al. (2020b) look at how catastrophic forgetting can be impacted by the training regime, and show that certain hyperparameter settings produce wider minima which lead to less catastrophic forgetting. Finally, Mirzadeh et al. (2020a) compare minima that result from multitask learning and continual learning, and show that the the minima resulting from continual learning are linear mode connected to the optimal sequential multitask minima, but not to each other, which results in forgetting and a corresponding drop in performance. All of these works either explore the relation between pre-training and flatness of minima in single-task settings, or between flatness of minima and model generalization capability. We extend this line of work by examining whether benefits from pre-training can persist across training on several tasks, assessing the effects of pre-training on loss landscapes over the course of LL, and validating a hypothesis explaining the effects of pre-training on LL.

3. Preliminaries

3.1. Problem Setup

We consider a setup where we receive a continuum of data from different tasks in sequential manner: $(x_1, y_1, t_1), \cdots, (x_i, y_i, t_i), \cdots$. Each triplet (x_i, y_i, t_i) consists of a task descriptor $t_i \in \mathcal{T}$, input data $x_i \in \mathcal{D}_{t_i}$ and target labels $y_i \in \mathcal{Y}_{t_i}$. In our setup, we consider an explicit task descriptor t_i because the same input x_i can appear in multiple different tasks but with different labels. For example, we can have a product review with positive sentiment along with the grammatical acceptability judgements. Following Lopez-Paz & Ranzato (2017), we assume that the continuum is locally i.i.d, i.e., each triplet (x_i, y_i, t_i) satisfies $(x_i, y_i) \overset{iid}{\sim} \mathcal{P}_{t_i}(X, Y)$. Based upon the observed data, our goal is to learn a predictor $f : \mathcal{X} \times \mathcal{T} \rightarrow \mathcal{Y}$ where we want to evaluate test pairs (x, t) from previously observed tasks (backward transfer) and current task at any time during the continual training of our model.

3.2. Datasets and Tasks

We conduct experiments on CV and NLP applications, using the following datasets:

Split CIFAR-100 This dataset is based on the CIFAR-100 image classification dataset (Krizhevsky & Hinton, 2009). The 100 classes are randomly split into 20 5-way classification tasks, with each task containing 2500 train examples and 500 test examples.

Split CIFAR-50 This dataset takes the first 50 classes of the CIFAR-100 dataset, and randomly splits them into five 10-way classification tasks. Each task contains 5000 training examples and 1000 test examples. This dataset serves as a homogeneous counterpart to the diverse 5-dataset, so it was constructed to match the structure and length of 5-dataset.

5-dataset (Ebrahimi et al., 2020) consists of five 10-way image classification datasets: CIFAR-10 (Krizhevsky & Hinton, 2009), MNIST (LeCun, 1998), Fashion-MNIST (Xiao et al., 2017), SVHN (Netzer et al., 2011), and notMNIST (Bulatov, 2011).

Split YahooQA This dataset is a 10-way topic classification dataset (Zhang et al., 2015) used to create five 2-way classification tasks. We randomly split topics into different tasks.

Each task contains around $279,000$ training examples and $12,000$ testing examples.

5-dataset-NLP consists of text classification datasets (Zhang et al., 2015) from five diverse domains and constitutes four tasks: (1) News classification (AGNews, 4-way classification); (2) Sentiment analysis (Yelp, Amazon, 5-way classification); (3) Wikipedia article classification (DBPedia, 14-way classification); and (4) question and answer topic categorization (YahooQA, 10-way classification). We follow the data processing procedure mentioned in (de Masson d'Autume et al., 2019) and have $115,000$ training examples and $7,600$ test examples per domain.

15-dataset-NLP consists of 15 diverse text classification tasks: (1) CoLA (Warstadt et al., 2019); (2) BoolQ (Clark et al., 2019); (3) SST-2 (Socher et al., 2013); (4) QQP[2]; (5) YahooQA (Zhang et al., 2015); (6) Yelp (Zhang et al., 2015); (7) Event Factuality (Poliak et al., 2018); (8) Argument Aspect Mining (Stab et al., 2018); (9) Explicit Discourse Marker Prediction (Prasad et al., 2019; Kim et al., 2020); (10) QNLI (Wang et al., 2018); (11) Rocstory (Mostafazadeh et al., 2016); (12) MNLI (Williams et al., 2018); (13) SciTail (Khot et al., 2018); (14) Implicit Discourse Relation Classification (Prasad et al., 2019; Kim et al., 2020); and (15) Emotion Detection (Saravia et al., 2018). For more details on this dataset, see Appendix B.

3.3. Evaluation

Let $S_{t,\tau}$ denote the score (e.g., accuracy) on the task τ after training on task t. After model finishes training on the task t, we compute the **average accuracy** (A_t) , **forgetting** (F_t) and **learning accuracy** (LA_t) metrics as proposed by (Lopez-Paz & Ranzato, 2017; Riemer et al., 2019). F_t measures the influence of learning task t on the performance of all previously seen tasks $\tau, (1 \leq \tau < t)$. LA_t measures the learning capability when model sees the new task t. Say we learn the t^{th} task, then A_t, F_t and LA_t are defined as follows:

$$A_t = \frac{1}{t}\sum_{\tau=1}^{t} S_{t,\tau}; LA_t = \frac{1}{t}\sum_{\tau=1}^{t} S_{\tau,\tau}$$

$$F_t = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \max_{\tau' \in \{1,\cdots,t-1\}} (S_{\tau',\tau} - S_{t,\tau}) \quad (1)$$

3.4. Methods

We consider prominent approaches from the literature for our analysis. We first consider the **finetune (FT)** approach, where we simply fine-tune the model on each task in sequence with no additional constraints on learning. **Elastic weight consolidation (EWC)** (Kirkpatrick et al., 2017) is a regularization-based approach that tries to mitigate forgetting by limiting learning for parameters important to previously learned tasks, as measured by the Fisher information matrix. In the **experience replay (ER)** (Chaudhry et al., 2019) method, we augment the base model with episodic memory module which retains examples from the previously seen tasks. We retain one example per task per class and randomly select examples for storage.

[2]https://www.quora.com/share/First-Quora-Dataset-Release-Question-Pairs

4. Does pre-training implicitly alleviate forgetting?

Having defined the formal problem definition, evaluation metrics, and methods for alleviating the forgetting phenomenon, in this section we conduct experiments to tease apart the role of pre-training for LL. We are interested in answering the following questions: (Q1) How much does pre-training help in alleviating the forgetting? (Q2) Do pre-trained weights undergo similar forgetting on diverse (5-dataset, 5-dataset-NLP) and homogeneous tasks (Split CIFAR-50, Split YahooQA)? (Q3) How do different pre-trained initializations affect forgetting?

Experimental design. To answer these questions convincingly, we select various datasets across text and vision domains. For text classification, we conduct experiments on Split YahooQA, 5-dataset-NLP, and 15-dataset-NLP. For image classification, we consider Split CIFAR-50, Split CIFAR-100, and 5-dataset. We utilize the DistilBERT$_{\text{BASE}}$ (Sanh et al., 2019) architecture for text classification and the ResNet-18 (He et al., 2016) architecture for image classification. To isolate the effect of pre-training, we consider two variants for each of these architectures: pre-trained models (**DistilBERT-PT**, **ResNet-18-PT**) and randomly initialized models (**DistilBERT-R**, **ResNet-18-R**). For our study, we need to ensure that there are as few confounding factors as possible. Therefore, we keep all other hyperparameters the same and vary only the initialization (for more details refer to Appendix A). To measure the severity of forgetting, we ideally want sufficient training samples to ensure either a pre-trained model or randomly initialized model of the same capacity can achieve similar learning accuracy on each task. To control for this behavior we either select a large training corpus whenever available (e.g., 279k examples for the Split YahooQA task) or run our experiments for multiple epochs (5 epochs for CV tasks).

4.1. How much does pre-training help in alleviating forgetting?

From Table 1, **we see that models with pre-trained initializations** (ResNet-18-PT, DistilBERT-PT) **undergo significantly less forgetting in comparison to models with random initializations** (ResNet-18-R, DistilBERT-R). This trend holds across all three methods. For text classifica-

Table 1: Comparing performance in terms of average accuracy, forgetting, and learning accuracy for Finetune, EWC, ER methods after training on the last task. ↑ indicates higher is better, ↓ indicates lower is better. All metrics are averaged across 5 runs. **Overall, we observe that pre-trained initialization undergoes significantly less forgetting compared to the randomly initialized model.**

	w/o Pretraining (ResNet-18-R/ DistilBERT-R)			w/ Pretraining (ResNet-18-PT/ DistilBERT-PT)		
	Accuracy(%) ↑	Forgetting(%) ↓	LA(%) ↑	Accuracy(%) ↑	Forgetting(%) ↓	LA(%) ↑
Split YahooQA (1 epoch)						
Finetune	71.18(±4.94)	28.76(±6.16)	94.20(±0.01)	85.89(±3.60)	11.68(±4.51)	95.23(±0.02)
EWC	79.67(±1.98)	18.15(±2.49)	94.19(±0.00)	90.81(±1.59)	5.49(±1.98)	95.21(±0.02)
ER	76.93(±1.04)	21.60(±1.32)	94.22(±0.02)	89.26(±0.72)	7.45(±0.90)	95.22(±0.04)
5-dataset-NLP (1 epoch)						
Finetune	45.36(±4.32)	35.30(±5.44)	73.60(±0.04)	64.25(±4.52)	16.77(±5.64)	77.67(±0.07)
EWC	56.50(±4.43)	21.11(±5.65)	73.39(±0.09)	70.12(±2.09)	9.15(±2.71)	77.44(±0.11)
ER	55.41(±3.39)	22.70(±4.35)	73.57(±0.10)	70.28(±1.61)	9.27(±2.06)	77.70(±0.01)
Split CIFAR50 (1 epoch)						
Finetune	37.87(±2.79)	8.09(±2.27)	45.75(±1.16)	88.57(±1.01)	3.02(±0.61)	91.59(±0.63)
EWC	38.28(±0.76)	6.12(±1.69)	44.40(±1.43)	88.63(±0.84)	3.12(±0.79)	91.75(±0.57)
ER	37.78(±2.58)	8.61(±1.36)	46.34(±1.71)	88.90(±0.72)	2.71(±0.73)	91.49(±0.38)
5-dataset (1 epoch)						
Finetune	29.41(±4.45)	39.40(±6.00)	68.62(±2.67)	62.58(±5.42)	31.02(±5.42)	93.60(±0.29)
EWC	31.57(±6.92)	37.47(±2.81)	69.05(±4.81)	64.43(±3.83)	29.20(±3.92)	93.63(±0.33)
ER	37.94(±8.58)	29.97(±7.62)	67.90(±3.26)	72.67(±2.59)	20.53(±2.59)	93.19(±0.32)
Split CIFAR100 (1 epoch)						
Finetune	45.76(±3.17)	18.28(±2.26)	63.65(±1.41)	89.09(±1.47)	6.05(±1.06)	95.06(±0.67)
EWC	46.00(±2.78)	18.13(±2.32)	63.78(±0.98)	89.44(±1.48)	5.71(±0.88)	95.03(±0.64)
ER	52.26(±1.44)	14.30(±0.80)	65.97(±0.70)	90.23(±1.44)	5.19(±0.94)	95.05(±0.54)
Split CIFAR50 (5 epochs)						
Finetune	42.76(±3.14)	23.68(±1.09)	66.44(±2.14)	88.96(±1.48)	5.20(±0.80)	94.16(±0.68)
EWC	45.28(±2.53)	20.65(±1.47)	65.93(±1.28)	88.98(±1.03)	5.31(±0.58)	94.29(±0.46)
ER	45.76(±1.76)	20.63(±1.41)	66.38(±2.65)	88.77(±0.85)	5.16(±0.54)	93.93(±0.40)
5-dataset (5 epochs)						
Finetune	33.72(±2.53)	51.51(±2.58)	85.23(±1.99)	58.63(±2.70)	36.96(±2.68)	95.59(±0.16)
EWC	31.57(±6.92)	37.47(±2.81)	69.05(±4.81)	57.23(±4.18)	38.35(±3.95)	95.58(±0.34)
ER	50.58(±4.50)	35.02(±5.35)	85.60(±1.31)	70.92(±4.00)	24.32(±3.89)	95.25(±0.37)
Split CIFAR100 (5 epochs)						
Finetune	38.89(±2.20)	39.11(±2.20)	77.96(±0.96)	85.47(±1.41)	10.91(±1.23)	96.37(±0.55)
EWC	37.37(±1.47)	40.12(±1.82)	77.47(±1.54)	85.79(±0.90)	10.63(±0.51)	96.42(±0.59)
ER	48.60(±1.86)	29.84(±1.32)	78.10(±0.66)	88.67(±1.64)	7.65(±1.23)	96.29(±0.61)

tion tasks (Split YahooQA, 5-dataset-NLP), we see that both models have comparable learning accuracy (LA) and significantly less forgetting for DistilBERT-PT. This can be completely attributed to the pre-trained initialization. For image experiments, we see that ResNet-18-R suffers from low learning accuracy when trained in an online fashion (1 epoch). This can be addressed by increasing the number of epochs (to 5). For example, we see that the learning accuracy on 5-dataset increases from 68 to around 85 when the number of epochs is increased. Now on 5-dataset with 5 epochs, ResNet-18-PT (36.96) undergoes less forgetting when compared to ResNet-18-R (51.51). Specifically, despite task accuracy starting at a higher base for ResNet-18-PT, *the absolute forgetting value is still lower compared to ResNet-18-R models*. Additionally, this effect also holds when considering a sequentially finetuned pre-trained model (with no additional regularization to alleviate forgetting) to a randomly initialized model trained with state-of-the-art LL methods. For example, on 5-dataset-NLP, sequentially finetuning DistilBERT-PT undergoes less forgetting (16.77) compared to the state-of-the-art ER method (22.79) when applied to DistillBERT-R. This raises an interesting research direction — explicitly focusing on learning generic features while training sequentially apart from just focusing on the forgetting aspect of LL.

4.2. Do pre-trained weights undergo similar forgetting on diverse and homogeneous tasks?

Most work in the LL literature evaluate algorithms on the Split MNIST, Split CIFAR-10, Split CIFAR-100, and FewRelations benchmarks (Chaudhry et al., 2019; Wang et al., 2019), which are homogenous in nature; different tasks in these benchmarks are sourced from the same underlying data distribution. From Table 1, we see that ResNet-18-PT does not undergo a significant amount of forgetting when sequentially fine-tuned on Split CIFAR-50, Split CIFAR-100 (homogenous tasks). On Split CIFAR-50, forgetting is around 3-5 accuracy points. Surprisingly, the state-of-the-art ER method also undergoes a similar amount of forgetting,

Table 2: Comparing performance in terms of average accuracy, forgetting, and learning accuracy for sequential finetuning after training on the last task. ↑ indicates higher is better, ↓ indicates lower is better. All metrics are averaged across 5 runs. **Overall, we observe that models pre-trained on diverse corpora (RoBERTa$_{\text{BASE}}$) undergo minimal forgetting across both 5 and 15 diverse tasks.**

Model	$\|\theta\|$	5-dataset-NLP (1 epoch)			15-dataset-NLP (1 epoch)		
		Accuracy(%) ↑	Forgetting(%) ↓	LA(%) ↑	Accuracy(%) ↑	Forgetting(%) ↓	LA(%) ↑
DistilBERT$_{\text{BASE}}$	66M	64.25(±4.52)	16.77(±5.64)	77.67(±0.07)	47.51(±4.16)	18.33(±4.94)	64.57(±1.09)
BERT$_{\text{BASE}}$	110M	67.02(±2.40)	14.24(±2.92)	78.42(±0.08)	51.79(±1.72)	20.35(±1.95)	70.78(±0.33)
RoBERTa$_{\text{BASE}}$	336M	71.25(±1.60)	9.80(±1.92)	79.10(±0.10)	**55.35**(±1.43)	**20.57**(±1.38)	**74.54**(±0.66)
BERT$_{\text{LARGE}}$	125M	71.62(±1.47)	9.42(±1.84)	79.16(±0.04)	48.43(±9.36)	27.98(±8.15)	74.55(±1.76)

thereby raising a question about the applicability of these datasets when studying forgetting in the context of the pre-trained ResNet models. It may be possible to manually cluster tasks based upon semantic closeness, rendering severe interference to make these benchmarks more challenging (Ramasesh et al., 2020). Given the generic nature of the pre-trained initialization, we ask: What happens when we train the model sequentially on diverse datasets, where diverse datasets are ones that span multiple sources? To answer this question, we conduct experiments on 5-dataset (image classification) and 5-dataset-NLP (text classification). From Table 1, **we empirically observe that pre-trained models are susceptible to forgetting when exposed to diverse tasks**. Particularly, DistilBERT-PT undergoes a 16.77 point drop in accuracy when trained on 5-dataset-NLP. Similarly, ResNet-18-PT undergoes a 31.02 point drop in accuracy when trained on 5-dataset.

4.3. How do different pre-trained initializations affect forgetting?

To examine the impact of varying pre-trained initialization on forgetting, we choose to evaluate different pre-trained Transfomer models, DistilBERT$_{\text{BASE}}$ (Sanh et al., 2019), BERT$_{\text{BASE}}$, BERT$_{\text{LARGE}}$ (Devlin et al., 2019), RoBERTa$_{\text{BASE}}$ (Liu et al., 2019), on text classification tasks. From the previous subsection, we observe that pre-trained models are relatively more susceptible to forgetting when sequentially training on diverse tasks. In response, we conduct a thorough investigation on the **5-data-NLP** dataset. From Table 2, we observe that when keeping the pre-training corpora the same and increasing the capacity of the model — DistilBERT$_{\text{BASE}}$ (66M), BERT$_{\text{BASE}}$ (110M), and BERT$_{\text{LARGE}}$ (336M) — we observe that larger models undergo less forgetting on sequential training of diverse NLP tasks. Further, to explore the impact of the diversity of the pre-training corpora, we compare BERT$_{\text{BASE}}$ (110M) with RoBERTa$_{\text{BASE}}$ (125M). We observe that the RoBERTa$_{\text{BASE}}$ model performs far superior to BERT$_{\text{BASE}}$, thus hinting at the necessity of diverse pre-training corpora to implicitly alleviate forgetting. Further, to stress-test these models, we introduce a novel suite of 15 diverse text classification tasks (for more details see Appendix B). We observe that by increasing the number of tasks in the sequence, pre-trained models undergo severe forgetting. Surprisingly, the RoBERTa$_{\text{BASE}}$ model out-performs BERT$_{\text{LARGE}}$ despite having many fewer parameters. We conclude based on the empirical results that **diversity of pre-training corpora is highly relevant when it comes to easing catastrophic forgetting while training on a diverse sequence of tasks.**

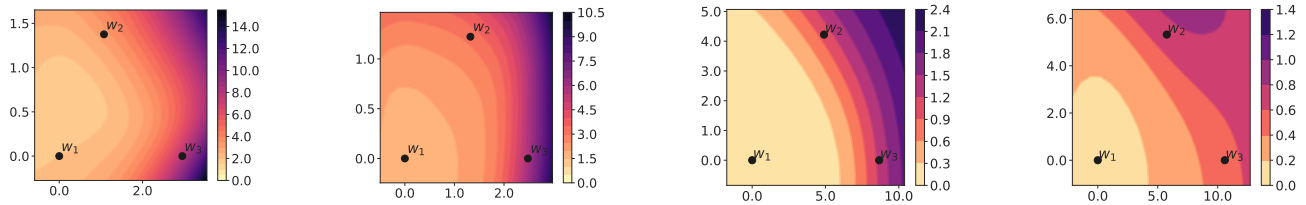
5. Exploring the Loss Landscape

To better understand how pre-training reduces forgetting, we perform experiments analyzing where models are situated in the loss landscape after training on each task. We denote model parameters after training on task k as w_k. If we define forgetting as the increase in loss for a given task during training , Mirzadeh et al. (2020b) show that the forgetting can actually be bounded by:

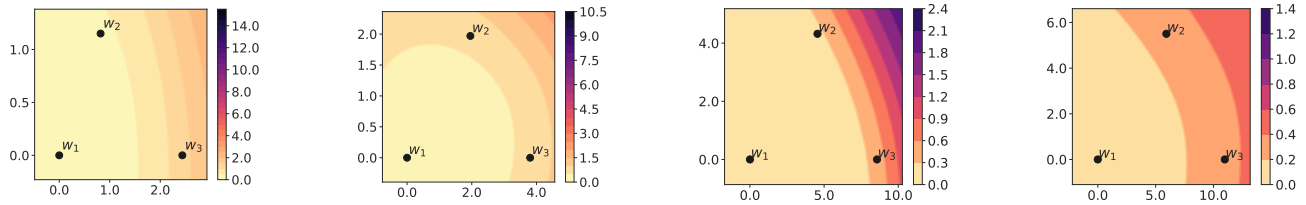
$$L_1(w_2) - L_1(w_1) \approx \frac{1}{2}\Delta w^\top \nabla^2 L_1(w_1)\Delta w \leq \frac{1}{2}\lambda_1^{max}\|\Delta w\|^2 \tag{2}$$

where $L_1(w)$ represents the loss on Task 1 with parameters w, $\Delta w = w_2 - w_1$, and λ_1^{max} is the largest eigenvalue of $\nabla^2 L_1(w_1)$. The magnitude of the eigenvalues of $L_1(w)$ can be used to characterize the curvature of the loss function (Keskar et al., 2017), and thus λ_1^{max} can be thought of as a proxy for the flatness of the loss function (lower is flatter). From Equation 2, we can see that the flatter the minima, the less forgetting occurs in the model.

We hypothesize that the improvements from pre-training shown in the previous section might be because pre-training leads to a more favorable loss landscape. Specifically, pre-training results in wider, flatter minima for each task. The effect of these wider minima is that the change in weights from learning on future tasks results in a smaller change on the actual loss for the current task, which leads to less forgetting. We verify this idea in two parts for models trained using the FT method. First we use loss contours and a sharpness metric to show that pre-training leads to flatter minima. We then interpolate between model checkpoints to show that the wider optima lead to smaller changes in loss.



(a) 5-dataset w/o PT (b) Split CIFAR-50 w/o PT (c) 5-dataset-NLP w/o PT (d) Split YahooQA w/o PT



(e) 5-dataset w/PT (f) Split CIFAR-50 w/PT (g) 5-dataset-NLP w/PT (h) Split YahooQA w/PT

Figure 2: Loss contours for Task 1 on each dataset. Each contour visualizes the model parameters after training on each of the first three tasks for randomly initialized models (top row) and pre-trained initialized models (bottom row). **Pre-training results in significantly wider optima.**

5.1. Loss Contour

In Figure 2, we visualize the contours of the test loss for the Task 1. We plot the locations of the model (w_1, w_2, w_3) after training on each of the first three tasks. Pre-training results in significantly wider optima. In fact, as the model is trained on tasks sequentially, the pre-trained model still remains mostly at the same loss level as compared to randomly initialized models, despite moving approximately the same (or even more) euclidean distance away from the original model. For example, in the plot for the pre-trained model on 5-dataset (Figure 2e), the model after the second task (w_2) remains at the same Task 1 loss level as after just training on Task 1 (w_1). It is a couple of loss levels higher for task 3 (w_3). For the randomly initialized model (Figure 2a), the euclidean distances between the model parameter vectors are approximately the same as for the pre-trained model, but the differences in Task 1 loss levels are significantly higher.

5.2. Sharpness

As another measure of the wideness of the minima, we calculate a sharpness metric (Keskar et al., 2017) for the model on each task as it goes through training. The metric tries to find the maximum value of the loss in the neighborhood of the minima, and calculates the difference between the maximum and the minimum loss value, scaled by the loss value. The maximization is performed in a subspace of the entire parameter space $\mathbb{R}^n$, specified by a projection matrix $A \in \mathbb{R}^{n \times p}$. For our experiments, we randomly sample our matrix A and set $p = 100$ as in Keskar et al. (2017). The neighborhood of the metric is given by Equation 3, where A^+ is the pseudo inverse of A, x is the parameter vector and ϵ is a hyperparameter controlling the size of the neighborhood. Equation 4 defines the sharpness metrics, where $f(x)$ denotes the loss value with parameters x. We calculate the sharpness metric at $\epsilon = 5 \times 10^{-4}$ and 10^{-3}. After training on each task, we compute the sharpness values of the minima reached by the model on that task. We then take the sharpness value of the run to be the mean of the values across the sequence of tasks. We present the mean and standard deviation across 5 runs.

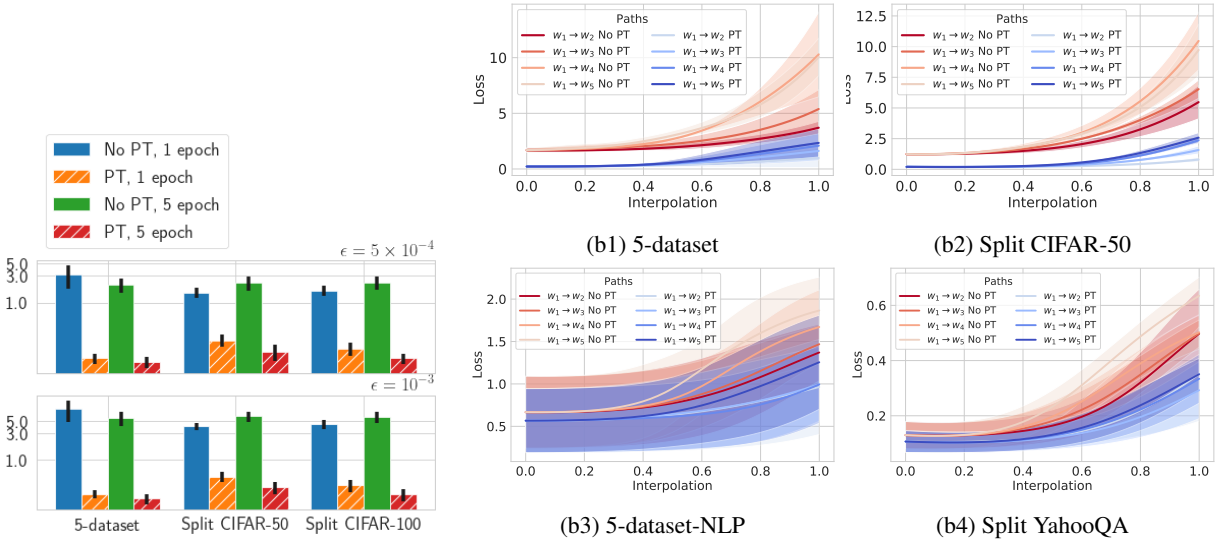
$$C_\epsilon = \{z \in \mathbb{R}^p : -\epsilon(|(A^+x)_i| + 1) \leq z_i \leq \epsilon(|(A^+x)_i| + 1)\ \forall i \in \{1 \dots p\}\} \quad (3)$$

$$\phi_{x,f} := \frac{(\max_{y \in C_\epsilon} f(x + Ay)) - f(x)}{1 + f(x)} \times 100 \quad (4)$$

For the language models, we encountered problems allocating and inverting A because of the large memory requirements (further explanation of the implementation is given in Appendix A). Thus, we only present the sharpness values for the vision experiments in Figure 3a. We see that for all datasets, the sharpness values for the pretrained initialized models are significantly lower than the values for the randomly initialized models.

5.3. Linear Model Interpolation

Ideally, to minimize forgetting, as the model sequentially trains on tasks, its loss on previous tasks would not change. This would be satisfied if the loss surface between model checkpoints were flat. To visualize this, we linearly interpolate between w_1 and the other checkpoints of the model,



(a) Average sharpness (log scale; lower is flatter) of minima across tasks in a 100 dimensional random subspace. **Pre-training reduces the sharpness of minima by an order of magnitude.**

(b) Loss interpolation plots for each dataset. Blue is pre-trained models, red is randomly initialized models. We interpolate between the checkpoint after Task 1 to the checkpoint after every other task, tracking the loss in the process. **In general, the loss landscape is flatter along these paths for pre-trained initialization models compared to randomly initialized models.**

Figure 3: Sharpness metrics and linear interpolation plots for pre-trained and random initialized models.

tracking the test loss on Task 1. This can be interpreted as viewing a slice of the contour plots shown in Figure 2 along the line that connects w_1 to each checkpoint. The results are shown in Figure 3b. The pre-trained plots are shown in hues of blue, and the random plots are shown in hues of red. These plots show that the pretrained initialized models experience a much more gradual increase in the loss compared to the randomly initialized models, even when interpolating to checkpoints created after training on several tasks. We provide more instances of these visualizations in Appendix C.

6. Discussion

In this paper, we study the effect of pre-training on lifelong learning across a variety of datasets and modalities, and we find that compared to models with random initializations, models with pretrained initializations undergo significantly less forgetting. Specifically, despite task accuracy starting at a higher base for pre-trained models, *the absolute forgetting value is still lower for pre-trained models*. This effect even holds when comparing a sequentially finetuned pre-trained model (with no additional regularization to improve performance or reduce forgetting) to a randomly initialized model trained with state-of-the-art lifelong learning methods.

To explain this effect, we perform several analyses of the loss landscapes produced in the course of training.We find that the minima created by the pre-trained models at the end of training on each task are significantly flatter and wider than those created by the randomly initialized models. This means that even when pre-trained models drift away from the original flat task minima, the task loss does not increase significantly, which results in less forgetting. We also explore the effect of different pre-trained models on performance for an NLP domain and find that while increased model capacity helps up to a certain point when considering shorter task sequences, when considering longer and more diverse task sequences, the quality of the pre-trained representations matter much more than model capacity.

Based on these results, a potential line of work could be to develop a regularizer that keeps the model from drifting too far from the pre-trained initialization, instead searching for task minima in the low loss basin. It could also be interesting to explore where the multitask minima are in relation to the pre-trained initialization, as Mirzadeh et al. (2020a) show that the sequential multitask minima are linear mode connected to minima after each task in lifelong learning. The flatness of the minima for every model starting from a pre-trained initialization could suggest a way to regularize the sequential training process with the pre-trained initialization such that that the model ends up at the multitask minima. One final takeaway from these results is that lifelong learning methods should focus on creating more general representations instead of simply reducing catastrophic forgetting, as more general representations appear to result in more robust learning.

References

Aljundi, R., Chakravarty, P., and Tuytelaars, T. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3366–3375, 2017.

Bulatov, Y. Notmnist dataset. Technical report, Google (Books/OCR), 2011. URL `http://yaroslavvb.blogspot.it/2011/09/notmnist-dataset.html`.

Caccia, M., Rodríguez, P., Ostapenko, O., Normandin, F., Lin, M., Page-Caccia, L., Laradji, I. H., Rish, I., Lacoste, A., Vázquez, D., and Charlin, L. Online fast adaptation and knowledge accumulation (osaka): a new approach to continual learning. In *NeurIPS*, 2020.

Chaudhry, A., Ranzato, M., Rohrbach, M., and Elhoseiny, M. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2018.

Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P. K., Torr, P. H., and Ranzato, M. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

Chen, Z. and Liu, B. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

de Masson d'Autume, C., Ruder, S., Kong, L., and Yogatama, D. Episodic memory in lifelong language learning. In *Advances in Neural Information Processing Systems*, pp. 13122–13131, 2019.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, 2019.

Ebrahimi, S., Meier, F., Calandra, R., Darrell, T., and Rohrbach, M. Adversarial continual learning. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M. (eds.), *Computer Vision – ECCV 2020*, pp. 386–402. Springer International Publishing, 2020.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Finn, C., Rajeswaran, A., Kakade, S., and Levine, S. Online meta-learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1920–1930. PMLR, 09–15 Jun 2019.

French, R. M. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Hao, Y., Dong, L., Wei, F., and Xu, K. Visualizing and understanding the effectiveness of bert. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4134–4143, 2019.

Hazelwood, K., Bird, S., Brooks, D., Chintala, S., Diril, U., Dzhulgakov, D., Fawzy, M., Jia, B., Jia, Y., Kalro, A., Law, J., Lee, K., Lu, J., Noordhuis, P., Smelyanskiy, M., Xiong, L., and Wang, X. Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 620–629, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Howard, J. and Ruder, S. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339, 2018.

Javed, K. and White, M. Meta-learning representations for continual learning. *arXiv preprint arXiv:1905.12588*, 2019.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Khot, T., Sabharwal, A., and Clark, P. Scitail: A textual entailment dataset from science question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Kim, N., Feng, S., Gunasekara, C., and Lastras, L. Implicit discourse relation classification: We need to talk about evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5404–5414, 2020.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

LeCun, Y. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Lomonaco, V., Maltoni, D., and Pellegrini, L. Rehearsal-free continual learning over small non-iid batches. *arXiv preprint arXiv:1907.03799*, 2019.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pp. 6467–6476, 2017.

McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Mirzadeh, S. I., Farajtabar, M., Gorur, D., Pascanu, R., and Ghasemzadeh, H. Linear mode connectivity in multitask and continual learning. *arXiv preprint arXiv:2010.04495*, 2020a.

Mirzadeh, S. I., Farajtabar, M., Pascanu, R., and Ghasemzadeh, H. Understanding the role of training regimes in continual learning. *arXiv preprint arXiv:2006.06958*, 2020b.

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P., and Allen, J. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 839–849, 2016.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

Neyshabur, B., Sedghi, H., and Zhang, C. What is being transferred in transfer learning? *arXiv preprint arXiv:2008.11687*, 2020.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10): 1345–1359, 2009.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237, 2018.

Poliak, A., Haldar, A., Rudinger, R., Hu, J. E., Pavlick, E., White, A. S., and Van Durme, B. Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 67–81, 2018.

Prasad, R., Webber, B., Lee, A., and Joshi, A. Penn discourse treebank version 3.0. In *LDC2019T05*. Philadelphia: Linguistic Data Consortium., 2019.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding with unsupervised learning. *Technical report, OpenAI*, 2018.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

Ramasesh, V. V., Dyer, E., and Raghu, M. Anatomy of catastrophic forgetting: Hidden representations and task semantics. *arXiv preprint arXiv:2007.07400*, 2020.

Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., , and Tesauro, G. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Saravia, E., Liu, H.-C. T., Huang, Y.-H., Wu, J., and Chen, Y.-S. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3687–3697, 2018.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. Green ai. *Commun. ACM*, 63(12):54–63, November 2020. ISSN 0001-0782. doi: 10.1145/3381831. URL https://doi.org/10.1145/3381831.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Sodhani, S., Chandar, S., and Bengio, Y. Toward training recurrent neural networks for lifelong learning. *Neural computation*, 32(1):1–35, 2020.

Stab, C., Miller, T., Schiller, B., Rai, P., and Gurevych, I. Cross-topic argument mining from heterogeneous sources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3664–3674, 2018.

Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3645–3650, 2019.

Thrun, S. Is learning the n-th thing any easier than learning the first? In *Advances in neural information processing systems*, pp. 640–646, 1996.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Wang, H., Xiong, W., Yu, M., Guo, X., Chang, S., and Wang, W. Y. Sentence embedding alignment for lifelong relation extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 796–806, 2019.

Wang, Z., Mehta, S. V., Poczos, B., and Carbonell, J. G. Efficient meta lifelong-learning with limited memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 535–548, 2020.

Warstadt, A., Singh, A., and Bowman, S. R. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Williams, A., Nangia, N., and Bowman, S. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, 2018.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3987–3995. JMLR. org, 2017.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, pp. 649–657, 2015.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2021.

A. Implementation Details

Vision Experiments For all vision experiments, we use the full ResNet-18 (He et al., 2016) architecture, with the final linear layer replaced (the number of outputs corresponds to the total number of classes in all given tasks). During inference, only the subset of outputs corresponding to the given task is considered. All images are resized to 224×224, and normalized with $\mu = (0.485, 0.456, 0.406)$ and $\sigma = (0.229, 0.224, 0.225)$. We used a SGD optimizer with the learning rate set to .01 for all methods (we did a hyperparameter search for both pre-trained and randomly initialized models and found the learning rate 0.01 resulted in a good learning accuracy for both pre-trained and randomly initialized models). The batch size was set to 10 for the Split CIFAR-50 and Split CIFAR-100 experiments and 64 for the 5-dataset experiments. The memory per class for ER was set to 1, and the λ parameter for EWC was also set to 1.

NLP Experiments For most of the text classification experiments, we use the Transformer architecture based text encoder, $\text{DistilBERT}_{\text{BASE}}$ (Sanh et al., 2019) to encode our input. In a single sentence text classification task, x_t is an input sentence to be classified. In a sentence-pair classification task, concatenation of x_t^1 and x_t^2 sentences separated by a $[SEP]$ symbol is considered as a input x_t. DistilBERT produces a contextual representation of each token in x_t including a special beginning of the sentence token symbol $[CLS]$. We use the representation of the $[CLS]$ symbol from model as features for a linear task classifier. We have a separate classifier for each task. We mainly set hyper-parameters to default implementation from HuggingFace.[3] We use Adam as our optimizer, set dropout 0.1, the base learning rate to $2e^{-5}$, batch size to 32 and the maximum total input sequence length after tokenization to 128. For EWC, following (Wang et al., 2020), we set the regularization strength λ to 5000 and for ER, following (Chaudhry et al., 2019), the memory per class per task is set to 1.

A.1. Sharpness

The matrix $A \in \mathbb{R}^{n \times p}$ used for projecting the parameters onto a subspace is randomly sampled and then normalized row-wise. Since this matrix is very large, the computation of the pseudo inverse A^+ (required for calculating the bounds in Equation 3) is very memory intensive and unstable. Instead, we directly calculate A^+x by finding the least squares solution to $Ab = x$. To find the maximum referenced in Equation 4, we use the L-BFGS-B algorithm.[4] We set the maximum number of iterations for the algorithm to 10, and to speed up computation, we directly provide the gradients along with the loss to the algorithm, instead of using the default 2-point finite difference gradient estimation.

For ResNet-18 ($n = 11M$), we set $p = 100$. However, for DistilBERT ($n = 66M$) when we set $p = 100$, we notice extremely small values for the sharpness metric. With the increase in the number of parameters, n, we should ideally increase random subspace projection dimension p. Setting larger $p(> 100)$ values for DistilBERT, however, leads to memory issues relating to allocating space for A and computing the bounds (even with the more efficient method discussed above). So instead of evaluating the sharpness metric in a random manifold, we perform the maximization in the entire space $\mathbb{R}^n$ (basically setting $A = I_n$). According to Keskar et al. (2017), when ϵ is small enough and $A = I_n$, the sharpness metric in Equation 4 relates to the largest eigenvalue of $\nabla^2 f(x)$. In Table 3, we report sharpness values for DistilBERT on 5-dataset-NLP and Split YahooQA datasets for $\epsilon \in \{5e^{-5}, 1e^{-4}, 5e^{-4}\}$. We see that values in the case of pre-trained models (w/ PT) are lower compared to randomly initialized models (w/o PT), thereby, validating the relative flatness of the task minima in the case of pre-trained models.

Table 3: Average sharpness (lower is flatter) of tasks minima. DistilBERT-PT (w/ PT) reduces the sharpness in comparison to DistilBERT-R (w/o PT).

	$\epsilon = 5 \times 10^{-5}$		$\epsilon = 10^{-4}$		$\epsilon = 5 \times 10^{-4}$	
	w/o PT	w/ PT	w/o PT	w/ PT	w/o PT	w/ PT
5-dataset-NLP	32.67 ± 1.17	28.27 ± 1.19	213.61 ± 11.46	128.97 ± 10.49	596.82 ± 13.70	552.09 ± 17.28
Split YahooQA	10.41 ± 0.39	8.77 ± 0.44	53.23 ± 7.02	43.03 ± 4.21	545.06 ± 6.40	422.85 ± 44.31

[3] https://github.com/huggingface/transformers

[4] We used the implementation provided by scipy at https://docs.scipy.org/doc/scipy/reference/optimize.minimize-lbfgsb.html

B. Datasets

One of the objectives of our work is to study the role of different pre-trained initializations in lifelong learning. To enable this study, we introduce **15-dataset-NLP**, a novel suite of diverse tasks for lifelong learning. It consists of fifteen text classification tasks covering a broad range of domains and data sources. Although there exists a setup with 4 tasks spanning 5 datasets, **5-dataset-NLP** (de Masson d'Autume et al., 2019), we show that our introduced benchmark proves more challenging (see Table 2 and Section 4.3) than the previous setup for the Transformer models (e.g., DistilBERT, BERT, RoBERTa) considered in our study.

15-dataset-NLP benchmark consists of single sentence or sentence pair classification tasks. We design our benchmark from existing tasks such that (1) the overall dataset includes various domains, (2) different tasks are (dis)similar to each other, thereby, facilitating both transfer and interference phenomena. All tasks under consideration differ in dataset size (from 8.5k-400k), so for our experiments, we only use between 8.5-14k training examples from each task. Lifelong learning from highly imbalanced data is an interesting problem, and we feel that our introduced benchmark can be used to investigate this problem as well. As our data is gathered from publicly available sources, for some tasks we do not have access to hidden test examples. In such cases, we consider dev examples as test split and sample examples from train split for validation[5]. We describe the tasks below and Table 4 details the evaluation metrics and train/dev/test split sizes for each task.

1. Linguistic acceptability aims at identifying whether the given sequence of words is a grammatical sentence. The Corpus of Linguistic Acceptability (**CoLA**) ((Warstadt et al., 2019) consists of English sentences annotated with their grammatical judgements. The data spans multiple domains, specifically books and journal articles.

2. Boolean QA is a reading comprehension task of answering yes/no questions for a given passage. The Boolean Questions (**BoolQ**) (Clark et al., 2019) dataset consists of short passages with yes/no questions about the passage. The questions are sourced from anonymous Google users and paired up with passages from Wikipedia articles.

3. Sentiment analysis is a binary classification task of identifying the polarity (positive/negative sentiment) of a given text. The Stanford Sentiment Treebank (**SST-2**) (Socher et al., 2013) corpus consists of sentences from Rotten Tomatoes movie reviews annotated with their sentiment.

4. Paraphrase detection aims at identifying whether two sentences are semantically equivalent. The Quora Question Pairs (**QQP**) corpus constitutes of question pairs from **Quora**[6] website annotated for semantic equivalence of question pairs.

5. Q&A categorization is a topic classification task of categorizing question and answer text pairs into existing topics. The Yahoo! Answers Comprehensive Questions and Answers (**YahooQA**) (Zhang et al., 2015) corpus contains data corresponding to the ten largest categories from Yahoo! Webscope program.

6. Review rating prediction is a five-way classification task of predicting the number of stars the user has given in a review given the corresponding text. The **Yelp** (Zhang et al., 2015) dataset contains business reviews obtained from the Yelp Dataset Challenge (2015).

7. Event factuality prediction is the task of determining whether an event described in the text occurred. The factuality annotations from the **Decomp** corpus are recast into an NLI structure and we use the modified dataset from Diverse NLI Collection (Poliak et al., 2018).

8. Argument aspect mining is concerned with the automatic recognition and interpretation of arguments (assessing the stance, source, and supportability for a given topic). The Argument Aspect Corpus (**AAC**) (Stab et al., 2018) has over 25,000 arguments spanning eight topics annotated with three labels (no argument, supporting argument, opposing argument). Stab et al. (2018) collected the data from web documents representing a range of genre and text types, including blogs, editorials, forums, encyclopedia articles.

9. The explicit discourse marker prediction task aims at classifying the discourse markers between sentences. Specifically, words like 'and', 'but', 'because', 'if', 'when', 'also', 'while', 'as' mark the conceptual relationship between sentences (**DISCONN8**) and are considered as labels for this task as discussed in (Prasad et al., 2019; Kim et al., 2020). We use examples from the Penn Discourse TreeBank 3.0 marked for explicit discourse relationship for our experimentation.

[5]We plan to release sampled example indices for replicability of our results

[6]https://www.quora.com/share/First-Quora-Dataset-Release-Question-Pairs

10. Question-answering NLI (**QNLI**) is a task adapted from the SQuAD by converting it into the sentence pair classification task (Wang et al., 2018). QNLI is a binary classification task of detecting whether the context sentence contains the answer to the question.

11. Binary Sentence Ordering (BSO) is a binary classification task to determine the order of two sentences. This task is similar to pre-training objectives considered in recent works. We use Roc Stories (**RocBSO**) (Mostafazadeh et al., 2016) corpus for constructing the dataset for this task.

12. Natural language inference (NLI) is a three-way classification task of predicting whether the premise entails the hypothesis (entailment), contradicts the hypothesis (contradiction), or neither (neutral). The Multi-Genre Natural Language Inference (**MNLI**) (Williams et al., 2018) corpus consists of sentence pairs from different sources (transcribed speech, fiction, and government report) annotated for textual entailment.

13. Multi-choice QA is a reading comprehension task wherein given a passage and question, models need to pick up the right option out of provides ones. Khot et al. (2018) cast the multiple-choice science exam questions into an NLI structure to convert them to the binary classification task. We use the **SciTAIL** (Khot et al., 2018) dataset released by them for our experimentation.

14. Implicit discourse relation classification is a common task of identifying discourse relations between two text spans or arguments. The Penn Discourse TreeBank 3.0 (**PDTB3L1**) (Prasad et al., 2019; Kim et al., 2020) contains a hierarchical annotation scheme (top-level senses, fine-grained level-2 senses) and we use top-level senses comprising of four labels (expansion, comparison, contingency, temporal) for our experimentation.

15. Emotion detection is a classification task of detecting the emotions from a given text snippet. We use **Emotion** dataset (Saravia et al., 2018) which contains Twitter messages with six emotions: anger, fear, joy, love, sadness, and surprise.

B.1. Task Sequences

The task sequences for the **Split CIFAR-50** and **Split CIFAR-100** experiments were generated by randomly sampling classes without replacement for each task, similar to Chaudhry et al. (2019). Thus, the sequences were different for every random seed, but since we ran each method with the same 5 seeds, each method was trained and tested on the same 5 sequences.

For **Split YahooQA**, we created 5 tasks by using disjoint groups of consecutive classes (e.g. $\{0, 1\}, \{2, 3\} \ldots$). These tasks were than randomly ordered for each task sequence, and each method was trained and tested using the same 5 random sequences.

For **5-dataset**, we randomly select the following dataset orders:

Seq1 SVHN→notMNIST→Fashion-MNIST→CIFAR-10→MNIST

Seq2 SVHN→MNIST→notMNIST→Fashion-MNIST→CIFAR-10

Seq3 CIFAR-10→SVHN→notMNIST→Fashion-MNIST→MNIST

Seq4 notMNIST→Fashion-MNIST→CIFAR-10→SVHN→MNIST

Seq5 CIFAR-10→MNIST→notMNIST→SVHN→Fashion-MNIST

For **5-dataset-NLP**, we randomly select the following dataset orders (first 4 are consistent with (de Masson d'Autume et al., 2019)):

Seq1 Yelp→AGNews→DBPedia→Amazon→YahooQA

Seq2 DBPedia→YahooQA→AGNews→Amazon→Yelp

Seq3 Yelp→YahooQA→Amazon→DBpedia→AGNews

Seq4 AGNews→Yelp→Amazon→YahooQA→DBpedia

Table 4: **15-dataset-NLP**: Task/Dataset description and statistics. All tasks are either single sentence or sentence pair classification. |Train|, |Dev|, |Test| denotes the number of examples in train, dev, test splits respectively. |L| denotes the number of classes for each tasks.

Task	Dataset/ Corpus	Domain(s)/ Text source(s)	|Train|	|Dev|	|Test|	|L|	Metrics
Linguistic Acceptability	CoLA	Journal articles & books	7,695	856	1,043	2	Matthews correlation
Boolean Question Answering	BoolQ	Google queries, Wikipedia passages	8,483	944	3,270	2	Acc.
Sentiment Analysis	SST-2	Movie reviews	9,971	873	872	2	Acc.
Paraphrase Detection	QQP	Quora questions	10,794	4,044	4,043	2	Acc. & F1
Q & A Categorization	YahooQA	Yahoo! Answers	13,950	4,998	4,998	10	Acc.
Review Rating Prediction	Yelp	Business reviews	12,920	3,999	3,998	5	Acc.
Event Factuality	Decomp	FactBank	10,176	4,034	3,934	2	Acc.
Argument Aspect Detection	AAC	Web documents	10,893	2,025	4,980	3	Acc. & F1
Explicit Discourse Marker Prediction	DISCONN8	Penn Discourse TreeBank	9,647	1,020	868	8	Acc. & F1
Question Answering NLI	QNLI	Wikipedia	9,927	5,464	5,463	2	Acc.
Binary Sentence Order Prediction	RocBSO	Roc story, corpus	10,000	2,400	2,400	2	Acc.
Natural Language Inference	MNLI	speech, fiction, govt. reports	11,636	4,816	4,815	3	Acc.
Multi-choice Science QA	SciTAIL	Science exams	11,145	1,305	1,304	2	Acc.
Implicit Discourse Relation Classification	PDTB3L1	Penn Discourse TreeBank	13,046	1,183	1,046	4	Acc. & F1
Emotion Detection	Emotion	Twitter	9,600	2,000	2,000	6	Acc. & F1

Seq5 YahooQA→Yelp→DBPedia→AGNews→Amazon

For **15-dataset-NLP**, we randomly select and use the following 5 dataset orders:

Seq1 Decomp→BoolQ→AAC→Yelp→DISCONN8→SST-2→QQP→YahooQA→QNLI →RocBSO→MNLI→SciTAIL→CoLA→PDTB3L1→Emotion

Seq2 CoLA→QQP→MNLI→QNLI→Emotion→SST-2→BoolQ→Decomp→AAC→SciTAIL →RocBSO→Yelp→PDTB3L1→YahooQA→DISCONN8

Seq3 SciTAIL→BoolQ→SST-2→AAC→DISCONN8→YahooQA→QNLI→RocBSO→PDTB3L1 →Emotion→Decomp→MNLI→QQP→CoLA→Yelp

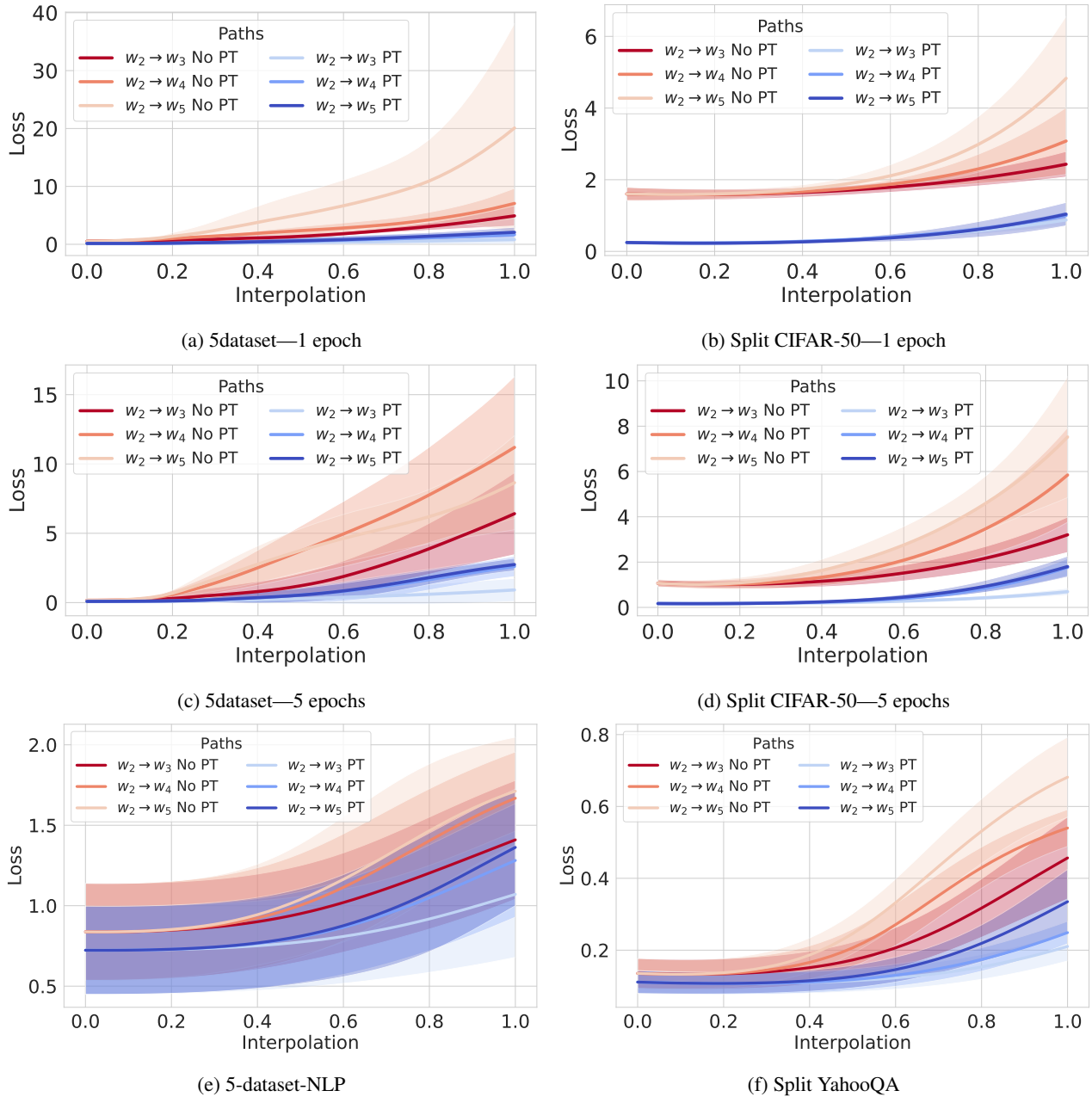
Seq4 DISCONN8→QNLI→CoLA→YahooQA→AAC→SciTAIL→PDTB3L1→Emotion →Decomp→RocBSO→QQP→Yelp→MNLI→BoolQ→SST-2

Seq5 Emotion→SST-2→RocBSO→YahooQA→AAC→MNLI→CoLA→DISCONN8→QQP →QNLI→Decomp→PDTB3L1→SciTAIL→Yelp→BoolQ

C. Loss Landscape

C.1. Linear Interpolation Loss

In Figure 4 we present additional linear model interpolation visualizations as described in Section 5.3. Specifically, we track the **Task 2** test loss as we interpolate from the model checkpoint after training on Task 2 **($\mathbf{w}_2$)** to the other checkpoints. These plots show that the trends presented in Figure 3b holds for Task 2 as well, thereby verifying that pre-trained initialized models lead to flatter task minima for subsequent tasks. Pre-trained initialized models experience a much more gradual increase in the loss compared to the randomly initialized models.



(a) 5dataset—1 epoch

(b) Split CIFAR-50—1 epoch

(c) 5dataset—5 epochs

(d) Split CIFAR-50—5 epochs

(e) 5-dataset-NLP

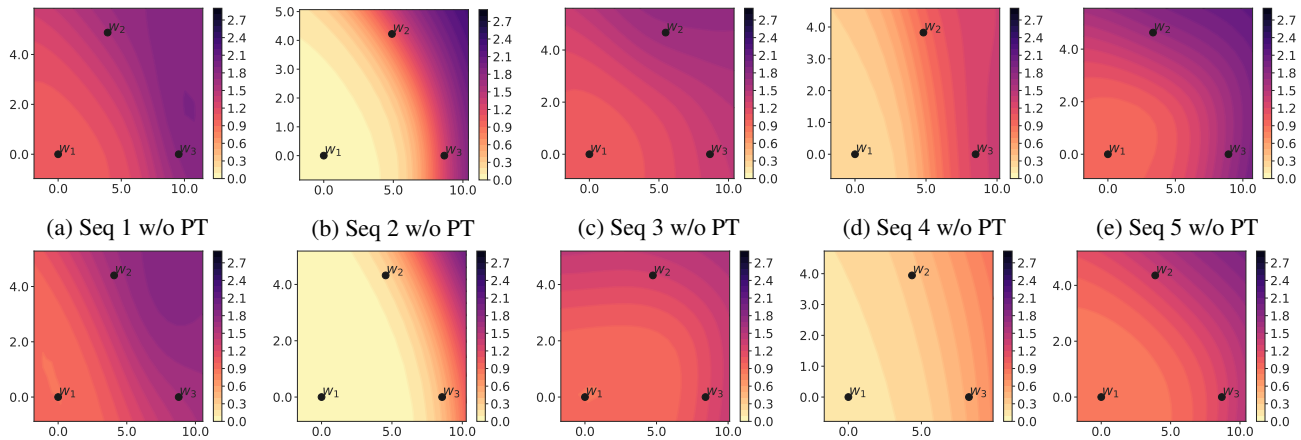
(f) Split YahooQA

Figure 4: Loss interpolation plots for each dataset. Blue is pre-trained models, red is randomly initialized models. We interpolate between the checkpoint after **Task 2** to the checkpoint after every other task, tracking the loss in the process. **In general, the loss landscape is flatter along these paths for pre-trained initialization models compared to randomly initialized models.**

C.2. Loss Contours

In this section we present loss contours for Task 1 and Task 2 for all task sequences (refer to Section B.1 for more details) for **5-dataset-NLP**, **Split YahooQA**, **Split CIFAR-50**, and **5-dataset**. For **Split CIFAR-50** and **5-dataset**, we also present the loss contours for 5-epoch training. In line with our observation from the sharpness and linear model interpolation analyses, pre-trained initialized models lead to flatter task minima for subsequent tasks as well.

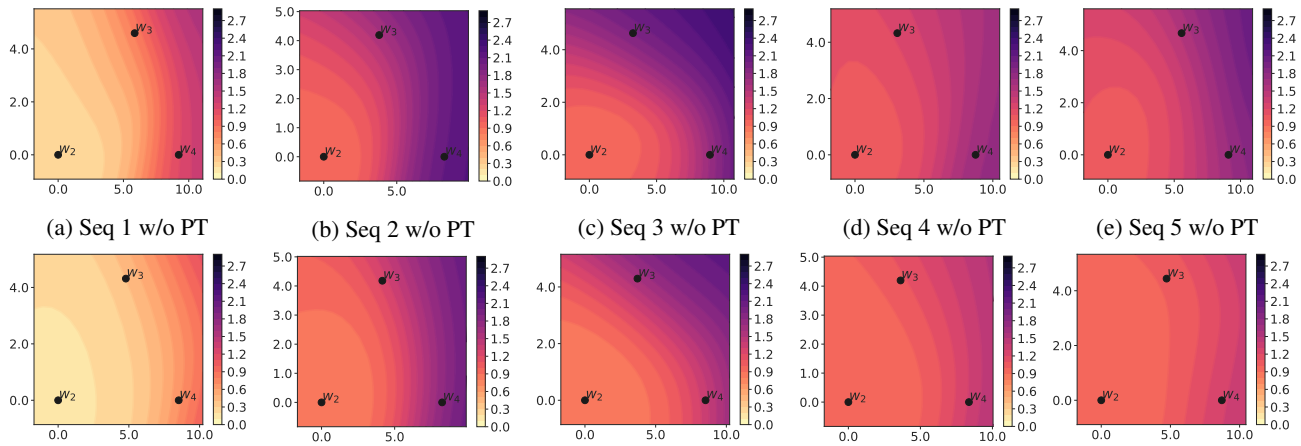
C.2.1. 5-DATASET-NLP



(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 5: Loss contours for **Task 1** on 5 task sequences of **5-dataset-NLP**. Each contour shows the location of the model parameters after training sequentially on **Task 1 ($\mathbf{w}_1$), Task 2 ($\mathbf{w}_2$), Task 3 ($\mathbf{w}_3$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

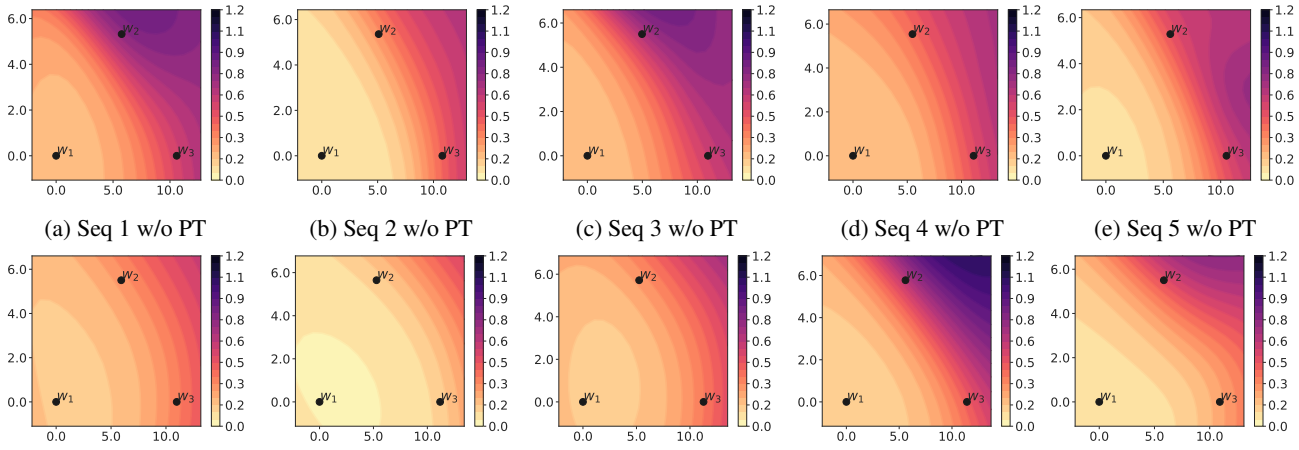


(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 6: Loss contours for **Task 2** on 5 task sequences of **5-dataset-NLP**. Each contour shows the location of the model parameters after training sequentially on **Task 2 ($\mathbf{w}_2$), Task 3 ($\mathbf{w}_3$), Task 4 ($\mathbf{w}_4$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

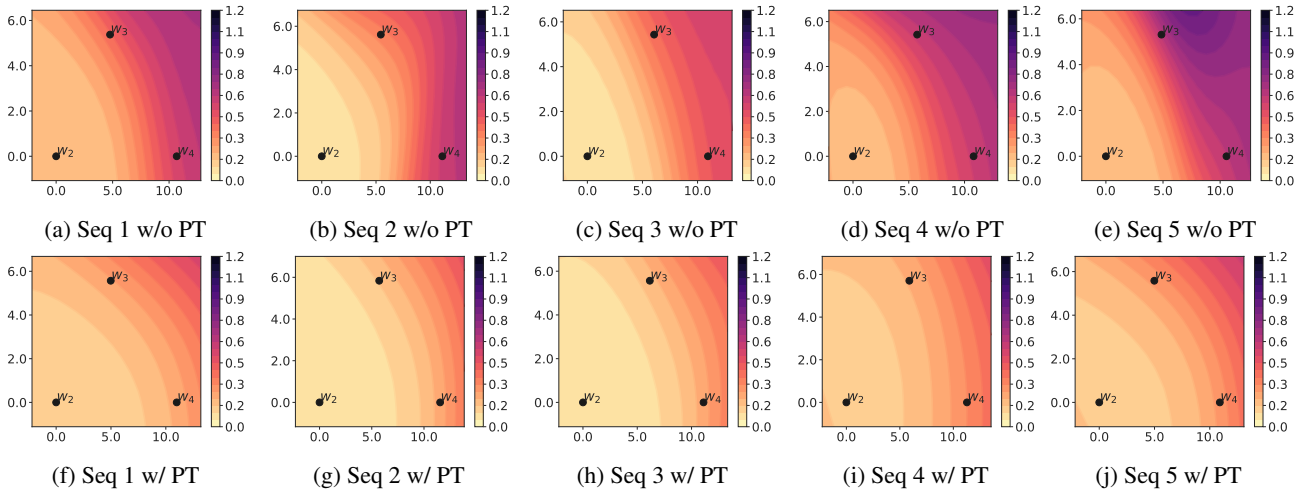
C.2.2. SPLIT YAHOOQA



(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 7: Loss contours for **Task 1** on 5 task sequences of **Split YahooQA**. Each contour shows the location of the model parameters after training sequentially on **Task 1 ($\mathbf{w}_1$), Task 2 ($\mathbf{w}_2$), Task 3 ($\mathbf{w}_3$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

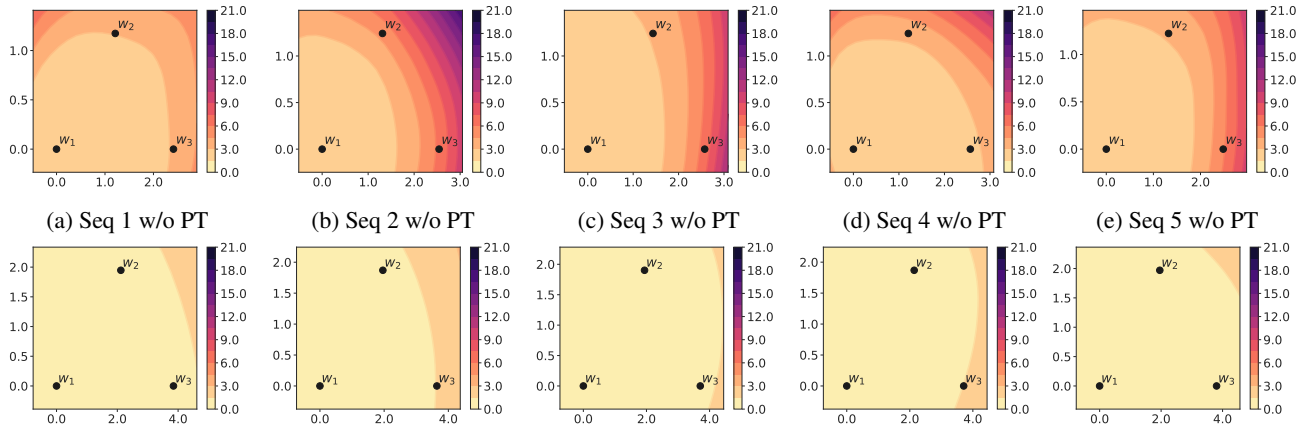


(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 8: Loss contours for **Task 2** on 5 task sequences of **Split YahooQA**. Each contour shows the location of the model parameters after training sequentially on **Task 2 ($\mathbf{w}_2$), Task 3 ($\mathbf{w}_3$), Task 4 ($\mathbf{w}_4$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

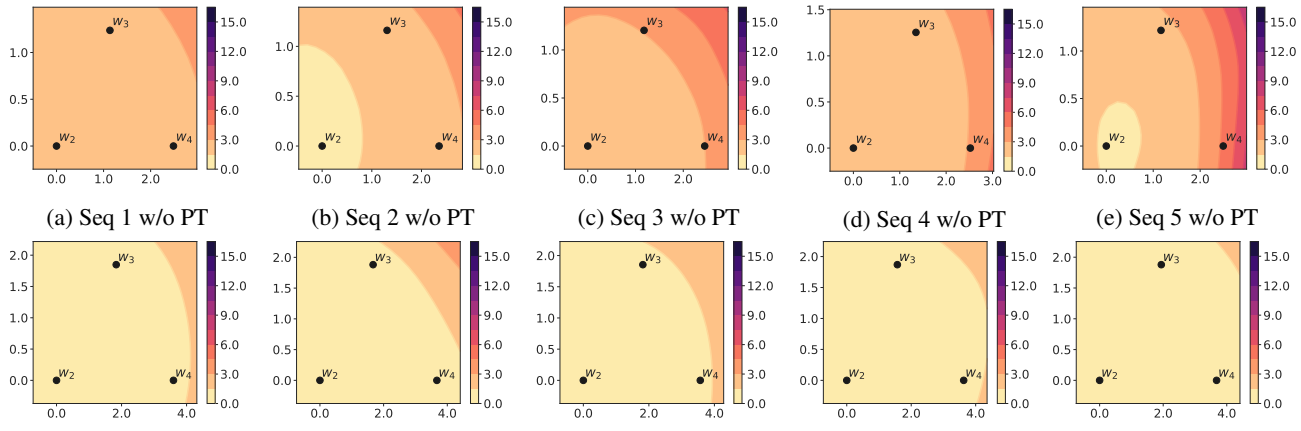
C.2.3. SPLIT CIFAR-50 (1 EPOCH)



(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 9: Loss contours for **Task 1** on 5 task sequences of **Split CIFAR-50** with 1 epoch of training on each task. Each contour shows the location of the model parameters after training sequentially on **Task 1 ($\mathbf{w}_1$), Task 2 ($\mathbf{w}_2$), and Task 3 ($\mathbf{w}_3$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

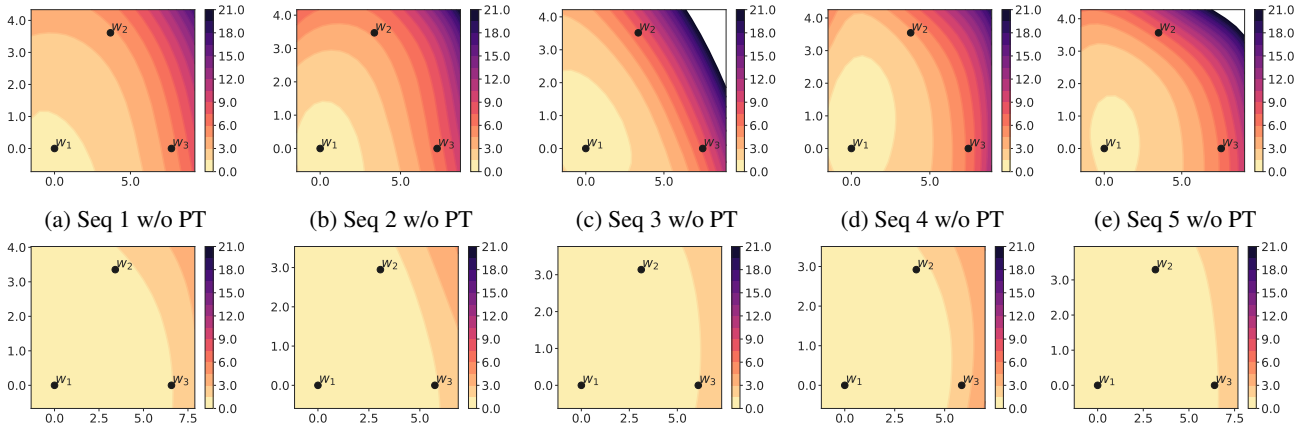


(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 10: Loss contours for **Task 2** on 5 task sequences of **Split CIFAR-50** with 1 epoch of training on each task. Each contour shows the location of the model parameters after training sequentially on **Task 2 ($\mathbf{w}_2$), Task 3 ($\mathbf{w}_3$), and Task 4 ($\mathbf{w}_4$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

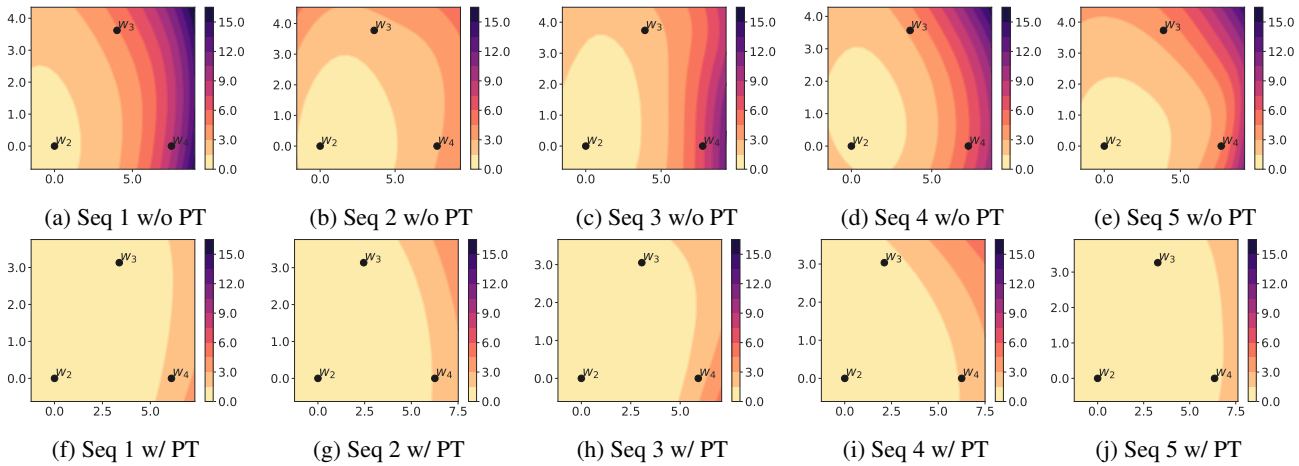
C.2.4. SPLIT CIFAR-50 (5 EPOCHS)



(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 11: Loss contours for **Task 1** on 5 task sequences of **Split CIFAR-50** with 5 epochs of training on each task. Each contour shows the location of the model parameters after training sequentially on **Task 1 ($\mathbf{w}_1$), Task 2 ($\mathbf{w}_2$), and Task 3 ($\mathbf{w}_3$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

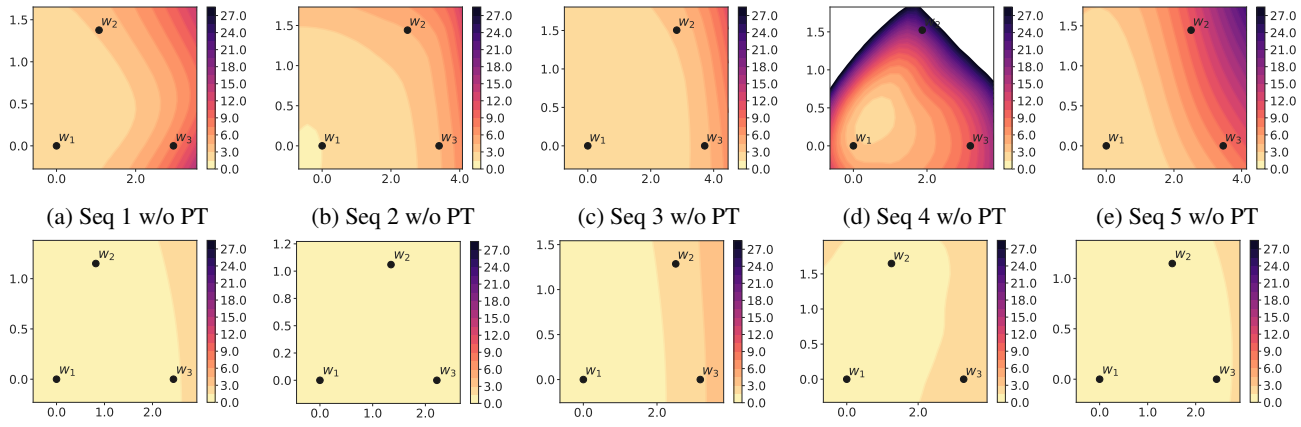


(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 12: Loss contours for **Task 2** on 5 task sequences of **Split CIFAR-50** with 5 epochs of training on each task. Each contour shows the location of the model parameters after training sequentially on **Task 2 ($\mathbf{w}_2$), Task 3 ($\mathbf{w}_3$), and Task 4 ($\mathbf{w}_4$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

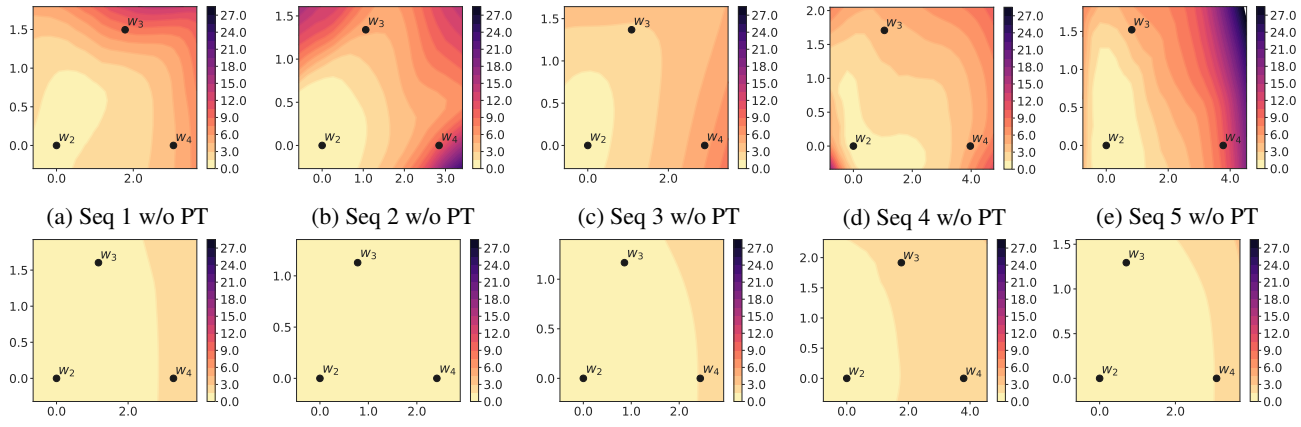
C.2.5. 5-DATASET (1 EPOCH)



(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 13: Loss contours for **Task 1** on 5 task sequences of **5-dataset** with 1 epoch of training on each task. Each contour shows the location of the model parameters after training sequentially on **Task 1 ($\mathbf{w}_1$), Task 2 ($\mathbf{w}_2$), and Task 3 ($\mathbf{w}_3$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

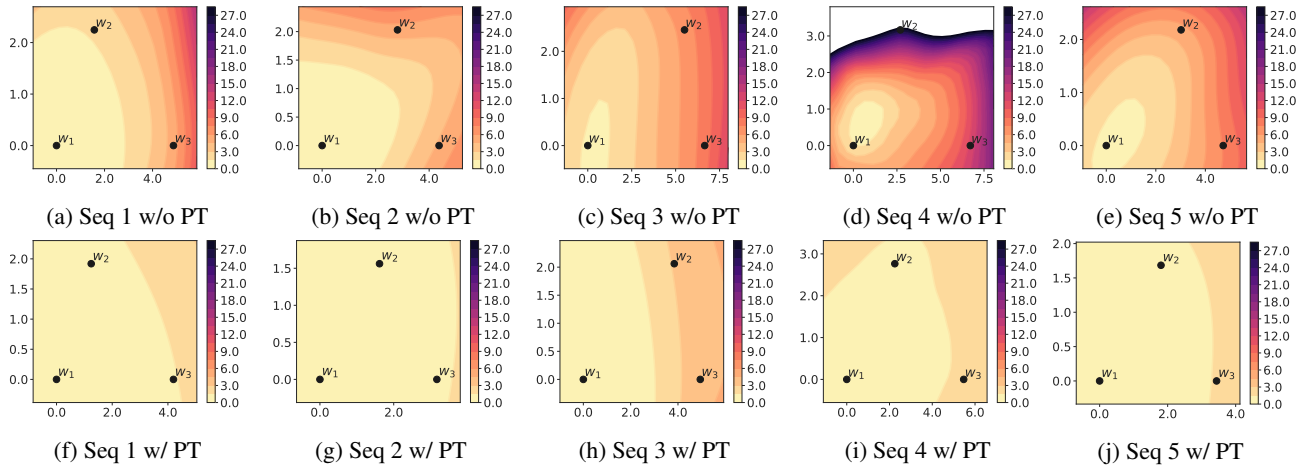


(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 14: Loss contours for **Task 2** on 5 task sequences of **Split CIFAR-50** with 1 epoch of training on each task. Each contour shows the location of the model parameters after training sequentially on **Task 2 ($\mathbf{w}_2$), Task 3 ($\mathbf{w}_3$), and Task 4 ($\mathbf{w}_4$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).

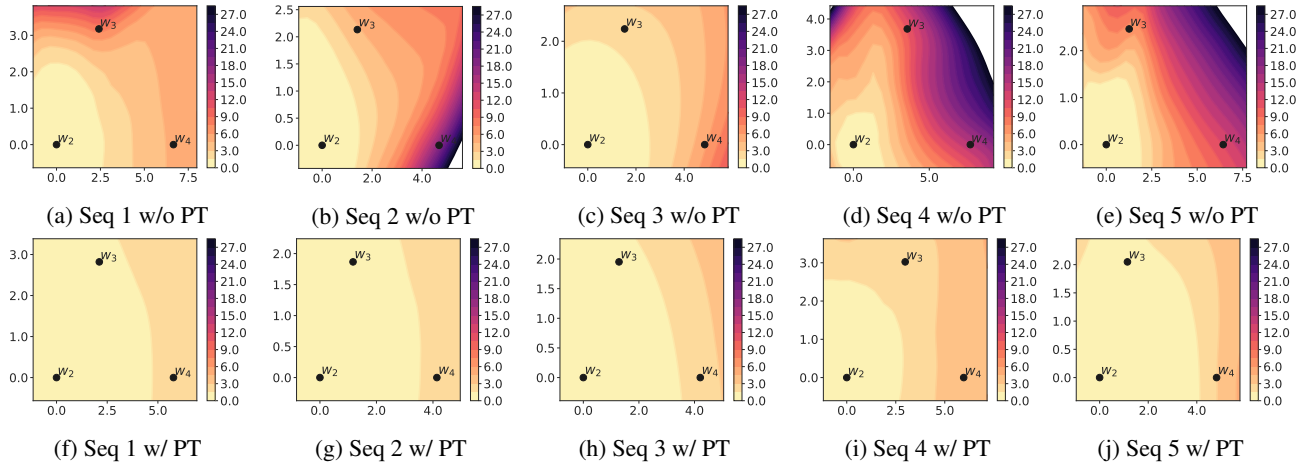
C.2.6. 5-DATASET (5 EPOCHS)



(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 15: Loss contours for **Task 1** on 5 task sequences of **Split CIFAR-50** with 5 epochs of training on each task. Each contour shows the location of the model parameters after training sequentially on **Task 1 ($\mathbf{w}_1$), Task 2 ($\mathbf{w}_2$), and Task 3 ($\mathbf{w}_3$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).



(a) Seq 1 w/o PT (b) Seq 2 w/o PT (c) Seq 3 w/o PT (d) Seq 4 w/o PT (e) Seq 5 w/o PT

(f) Seq 1 w/ PT (g) Seq 2 w/ PT (h) Seq 3 w/ PT (i) Seq 4 w/ PT (j) Seq 5 w/ PT

Figure 16: Loss contours for **Task 2** on 5 task sequences of **5-dataset** with 5 epochs of training on each task. Each contour shows the location of the model parameters after training sequentially on **Task 2 ($\mathbf{w}_2$), Task 3 ($\mathbf{w}_3$), and Task 4 ($\mathbf{w}_4$)**. The top row shows contours for randomly initialized models (w/o PT) and the bottom row shows contours for pre-trained initialized models (w/ PT).