Regularizing Trajectories to Mitigate Catastrophic Forgetting

Paul Michel
Language Technologies Institute
Carnegie Mellon University
pmichel1@cs.cmu.edu

Elizabeth Salesky
Center for Language and Speech Processing
Johns Hopkins University
esalesky@jhu.edu

Graham Neubig
Language Technologies Institute
Carnegie Mellon University
gneubig@cs.cmu.edu

Abstract

Regularization-based continual learning approaches generally prevent catastrophic forgetting by augmenting the training loss with an auxiliary objective. However in practical optimization scenarios with noisy data and/or gradients, it is possible that stochastic gradient descent can inadvertently change critical parameters. In this paper, we argue for the importance of regularizing optimization trajectories directly. We derive a new *co-natural* gradient update rule for continual learning whereby the new task gradients are preconditioned with the empirical Fisher information of previously learnt tasks. We show that using the co-natural gradient systematically reduces forgetting in continual learning. Moreover, it helps combat overfitting when learning a new task in a low-resource scenario.

1 Introduction

Endowing machine learning models with the capability to learn a variety of tasks in a sequential manner is critical to obtain agents that are both versatile and persistent. However, continual learning of multiple tasks is hampered by catastrophic forgetting [31, 39], the tendency of previously acquired knowledge to be overwritten when learning a new task.

Modern techniques to mitigate catastrophic forgetting can be roughly categorized into 3 lines of work (see [35] for a comprehensive overview): 1. regularization-based approaches, where forgetting is mitigated by the addition of a penalty term in the learning objective ([24, 8], *inter alia*), 2. dynamic architectures approaches, which incrementally increase the model's capacity to accomodate the new tasks [41], and 3. memory-based approaches, which retain data from learned tasks for later reuse [29, 9, 10]. Among these, regularization-based approaches are particularly appealing because they do not increase the model size and do not require access to past data. This is particularly relevant to real-world scenarios where keeping data from previous training tasks may be impractical because of infrastructural or privacy-related reasons. Moreover, they are of independent intellectual interest because of their biological inspiration rooted in the idea of synaptic consolidation [24].

In these regularization-based approaches, a good regularizer will ensure that, when learning a new task, gradient descent will ultimately converge to parameters that yield good results on the new task while preserving performance on previously learned tasks. Critically, this is predicated upon successful optimization of the regularized objective, a fact that has been largely taken for granted in previous work. Non-convexity of the loss function, along with noise in the data (due to small

Preprint. Under review.

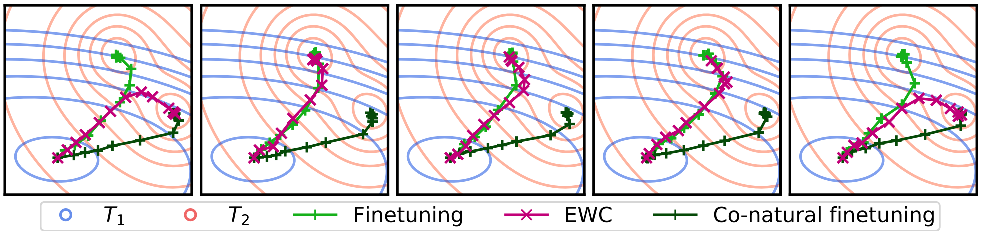


Figure 1: **On the importance of trajectories: an example with 2-dimensional logistic regression.** Having learned task T_1, the model is trained on T_2 with two different objectives: minimizing the loss on T_2 (Finetuning) and a regularized objective (EWC; [24]). We add a small amount of Gaussian noise to gradients in order to simulate the stochasticity of the trajectory. Plain finetuning and EWC often converge to a solution with high loss for T_1, but the co-natural optimization trajectory *consistently* converges towards the optimum with lowest loss for T_1.

or biased datasets) or in the gradients (due to stochastic gradient descent), can yield optimization trajectories — and ultimately convergence points — that are highly non-deterministic, even for the same starting parameters. As we demonstrate in this paper, this can cause unintended catastrophic forgetting along the optimization path. This is illustrated in a toy setting in Figure 1: a two parameter model is trained to perform task T_2 (an arbitrary bi-modal loss function) after having learned task T_1 (a logistic regression task). Standard finetuning, even in the presence of a regularized objective (EWC; [24]), quickly changes the loss of T_1 and tends to converge to a solution with high T_1 loss.

We propose to remedy this issue by regularizing not the objective function but *the optimization trajectory itself*, specifically by preconditioning gradient descent with the empirical Fisher information of previously learned tasks (§3). This yields what we refer to as a *co-natural* gradient, an update rule inspired by the natural gradient [4], but taking the Fisher information of *previous tasks* as a natural Riemannian metric[1] of the parameter space, instead of the Fisher information of the task being optimized for. When we introduce our proposed co-natural gradient for the toy example of Figure 1, the learning trajectory follows a path that changes the loss on T_1 much more slowly, and tends to converges to the optimum that incurs the lowest performance degradation on T_1.

We test the validity of our approach in a continual learning scenario (§4). We show that the co-natural gradient consistently reduces forgetting in a variety of existing continual learning approaches by a large factor, and greatly improves performance over simple finetuning, without modification to the training objective. As an additional contribution, we assemble a new collection of 13 tasks for evaluating continual learning on text classification.

We further investigate the special case of transfer learning in a two-task, low-resource scenario. In this specific case, control over the optimization trajectory is particularly useful because the optimizer has to rely on early stopping to prevent overfitting to the meager amount of training data in the target task. We show that the co-natural gradient yields the best trade-offs between source and target domain performance over a variety of hyper-parameters (§5).

2 Background and Notations

We first give a brief overview of the continual learning paradigm and existing approaches for overcoming catastrophic forgetting.

2.1 Notation

Let us define a task as a triplet containing an input space $\mathcal{X}$ and an output space $\mathcal{Y}$, both measurable spaces, as well as a distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. In general, learning a task will consist of training a model to approximate the conditional distribution $p(y \mid x)$ induced by $\mathcal{D}$.

Consider a probabilistic model p_θ parametrized by $\theta \in \mathbb{R}^d$ where d is the size of the model, trained to perform a *source* task $S = \langle \mathcal{X}_S, \mathcal{Y}_S, \mathcal{D}_S \rangle$ to some level of performance, yielding parameters θ_S.

[1] Informally, the reader can think of a Riemannian metric as a function that assigns an inner product $u, v \mapsto g_{\mathbf{x}}(u, v)$ to each point x in the space, thus inducing a localized notion of distance and curvature.

In the most simple instance of continual learning, we are tasked with learning a second *target* task $T = \langle \mathcal{X}_T, \mathcal{Y}_T, \mathcal{D}_T \rangle$. In general in a multitask setting, it is not the case that the input or output spaces are the same. The discrepancy between input/output space can be addressed in various ways, *e.g.* by adding a minimal number of task-specific parameters (for example, different softmax layers for different label sets). To simplify exposition, we set these more specific considerations aside for now, and assume that $\mathcal{X}_S = \mathcal{X}_T$ and $\mathcal{Y}_S = \mathcal{Y}_T$.

At any given point during training for task T, our objective will be to minimize the loss function $\mathcal{L}_T(\theta)$ – generally the expected negative log-likelihood $\mathbb{E}_{x,y\sim\mathcal{D}_T}[-\log p_\theta(y \mid x)]$. Typically, this will be performed by iteratively adding incremental update vectors $\delta \in \mathbb{R}^d$ to the parameters $\theta \longleftarrow \theta + \delta$.

2.2 Existing Approaches for Continual Learning

In this paper, we focus on those models that have a fixed architecture over the course of continual learning. The study of continual learning for models of fixed capacity can be split into two distinct (but often overlapping) streams of work:

Regularization-based approaches introduce a penalty in the loss function $\mathcal{L}_T$, often quadratic, pulling the weights θ back towards θ_S:

$$\mathcal{L}_T(\theta) = \underbrace{\mathbb{E}_{\mathcal{D}_T} - \log p_\theta(y \mid x)}_{\text{NLL on task } T} + \underbrace{\lambda(\theta - \theta_S)^T \Omega_S (\theta - \theta_S)}_{\text{Regularization term}} \tag{1}$$

where Ω_S is a matrix, generally diagonal, that encodes the respective importance of each parameter with respect to task S, and λ is a regularization strength hyper-parameter. Various choices have been proposed for Ω_S; the diagonal empirical Fisher information matrix [24], or path-integral based importance measures [54, 8]. More elaborate regularizers have been proposed based on *e.g.* a Bayesian formulation of continual learning [33, 1] or a distillation term [28, 16]. The main advantage of these approaches is that they do not rely on having access to training data of previous tasks.

Memory-based approaches store data from previously seen tasks for re-use in continual learning, either as a form of constraint, by *e.g.* ensuring that training on the new task doesn't increase the loss on previous tasks [29, 9], or for replay *i.e.* by retraining on instances from previous tasks [40, 10, 3, 2]. Various techniques have been proposed for the selection of samples to store in the memory [10, 3] or for retrieval of the samples to be used for replay [2].

All of these methods rely on stochastic gradient descent to optimize their regularized objective or to perform experience replay, with the notable exception of GEM [29, 9], where the gradients are projected onto the orthogonal complement of previous task's gradients. However, this method has been shown to perform poorly in comparison with simple replay [10], and it still necessitates access to data from previous tasks.

3 Regularizing the Trajectory

After briefly recalling derivation of the usual update in gradient descent, we derive a new, *co-natural* update designed to better preserve the distribution induced by the model over previous tasks.

3.1 Warm up: the Standard Gradient Descent Update

At point θ in the parameter space, gradient descent finds the optimal update δ that is (1) small and (2) locally minimizes the difference in loss $\mathcal{L}(\theta + \delta) - \mathcal{L}(\theta)$ ($\approx \delta^\intercal \nabla_\theta \mathcal{L}$ at the first order). Traditionally this can be formulated as minimizing the Lagrangian:

$$\mathbb{L}(\delta) = \underbrace{\delta^\intercal \nabla_\theta \mathcal{L}_T}_{\substack{\text{first order} \\ \text{loss minimization term}}} + \underbrace{\mu \|\delta\|^2}_{\text{“small update” term}} \tag{2}$$

with Lagrangian multiplier $\mu > 0$. Minimizing $\mathbb{L}$ for δ yields the well-known optimal update $\delta^* = -\frac{1}{2\mu}\nabla_\theta \mathcal{L}_T$, where $\frac{1}{2\mu}$ corresponds to the learning rate (see Appendix A.1 for the full derivation).

3.2 KL Regularization of Trajectories

The $\|\delta\|^2$ term in $\mathbb{L}$ implicitly expresses the underlying assumption that the best measure of distance between parameters θ and $\theta + \delta$ is the Euclidean distance. In a continual learning setting however, the quantity we are most interested in preserving is the probability distribution that θ models on the source task S, $p_\theta^S(x, y) = p_\theta(y \mid x)p^S(x)$

Therefore, a more *natural* distance between θ and $\theta + \delta$ is the Kullback-Leibler divergence $\mathrm{KL}(p_\theta^S \| p_{\theta+\delta}^S)$ [25]. For preventing catastrophic forgetting along the optimization path, we incorporate incorporate this KL term into the Lagrangian $\mathbb{L}$ itself:

$$\mathbb{L}(\delta) = \delta^\intercal \nabla_\theta \mathcal{L}_T + \mu\|\delta\|^2 + \nu \mathrm{KL}(p_\theta^S \| p_{\theta+\delta}^S) \tag{3}$$

Doing so means that the optimization trajectory will tend to follow the direction that changes the distribution of the model the least. Notably, this is not a function of the previous objective $\mathcal{L}_S$, so knowledge of the original training objective is not necessary during continual learning (which is typically the case in path-integral based regularization methods [54] or experience replay [10]).

3.3 Co-natural Gradient Optimization

Presuming that δ is small, we can perform a second order Taylor approximation of the function $\delta \mapsto \mathrm{KL}(p_\theta^S \| p_{\theta+\delta}^S)$ around 0. Considering that both the zeroeth and first order terms are null because 0 is a global minimizer of $\delta \mapsto \mathrm{KL}(p_\theta^S \| p_{\theta+\delta}^S)$, this reduces the Lagrangian to a quadratic optimization problem (we refer the reader to [36] for a more detailed derivation):

$$\mathbb{L}(\delta) = \delta^\intercal \nabla_\theta \mathcal{L}_T + \mu\|\delta\|^2 + \frac{1}{2}\nu \delta^\intercal F_\theta^S \delta \tag{4}$$

where F_θ^S is the Hessian of the KL divergence around θ. A crucial, well-known property of this matrix is that it coincides with the Fisher information matrix[2] $\mathbb{E}_{x,y\sim p_\theta}[(\nabla \log p_\theta^S)(\nabla \log p_\theta^S)^T]$ (the expectation being taken over the model's distribution p_θ; see Appendix A.1 for details). This is appealing from a computational perspective because the Fisher can be computed by means of first order derivatives only.

Minimizing for δ yields the following optimal update:

$$\delta^* = -\lambda \left[F_\theta^S + \alpha I\right]^{-1} \nabla_\theta \mathcal{L}_T \tag{5}$$

where coefficients μ and ν are folded into two hyper-parameters: the learning rate λ and a damping coefficient α (the step-by-step derivation can be found in Appendix A.1). In practice, especially with low damping coefficients, it is common to obtain updates that are too large (typically when some parameters have no effect on the KL divergence). To address this, we re-normalize δ^* to have the same ∞ norm as the original gradient, $\|\nabla \mathcal{L}_T\|_\infty$. Moreover, to improve numerical stability in the case of degenerate Fisher matrices, we add a very small damping term of 10^{-12} to the Fisher in all experiments.

For computational reasons, we will make 3 key practical approximations to the Fisher:

1. $F_\theta^S \approx F_{\theta_S}^S$: we maintain the Fisher computed at θ_S, instead of recomputing F_S at every step of training. This relieves us of the computational burden of updating the Fisher for every new value of θ. This approximation (shared by previous work, *e.g.* [24];[8]) is only valid insofar as θ_S and θ are close. While there is no theoretical ground for this approximation to hold beyond this neighborhood, we observe empirically (in §4.3) that this still allows for reduced forgetting.
2. F^S is diagonal: this is a common approximation in practice with two appealing properties. First, this makes it realistic to store the d diagonal Fisher coefficients in memory. Second, this trivializes the inverse operation (simply invert the diagonal elements).
3. Empirical Fisher: this common approximation replaces the expectation under the model's distribution by the expected log-likelihood of the *true* distribution: $E_{x,y\sim p^S}[(\nabla \log p_\theta^S)(\nabla \log p_\theta^S)^T]$ (mind the subscript). This is particularly useful in tasks with a large or unbounded number of classes (*e.g.* structured prediction), where summing over all possible outputs is intractable. We can then compute the diagonal of the empirical Fisher using Monte Carlo sampling: $\frac{1}{N}\sum_{i=1}^N [\nabla \log p_\theta^S(y_i \mid x_i)]^2$ with (x_i, y_i) sampled from $\mathcal{D}_S$ (we use $N = 1000$ for all experiments).

[2] Hence our use of the letter F to designate the Hessian

This formulation bears many similarities with the natural gradient from [4], which also uses the KL divergence as a metric for choosing the optimal update δ^* in gradient descent. There is a however a crucial difference, both in execution and purpose: where the natural gradient uses knowledge of the curvature of the KL divergence of $\mathcal{D}_T$ to *speed-up convergence*, our proposed method leverages the curvature of the KL divergence on $\mathcal{D}_S$ to *slow-down divergence* from $p^S_{\theta_S}$. To highlight the resemblance and complementarity between these two concepts, we refer to the new update as the *co-natural* gradient.

3.4 Beyond Two Tasks

In a continual learning scenario, we are confronted with a large number of tasks $T_1 \dots T_n$ presented in sequential order. When learning T_n, we can change the Lagrangian $\mathbb{L}$ from 3 to incorporate the constraints for all previous tasks $T_1 \dots T_{n-1}$:

$$\mathbb{L}(\delta) = \delta^\intercal \nabla_\theta \mathcal{L}_{T_n} + \mu \|\delta\|^2 + \sum_{i=1}^{n-1} \nu_i \mathrm{KL}(p^{T_i}_\theta \| p^{T_i}_{\theta+\delta}) \tag{6}$$

This in turn changes the Fisher in Eq. 6 to $\tilde{F}_{n-1} := \frac{1}{2}\sum_{i=1}^{n-1} \nu_i F^{T_i}$. The choice of the coefficients ν_i is crucial. Setting all ν_i to the same value, *i.e.* assigning the same importance to all tasks is suboptimal for a few reasons. First and foremost, it is unreasonable to expect of a model with finite capacity to remember an unbounded number of tasks (as tasks "fill-up" the model capacity, $\tilde{F}_{n-1}$ is likely to become more "homogeneous"). Second, as training progresses and θ changes, our approximation that $F^{T_i}_\theta \approx F^{T_i}_{\theta_{T_i}}$ is less and less likely to hold.

We address this issue in the same fashion as [42], by keeping a rolling exponential average of the Fisher matrices $\tilde{F}^\gamma_n = \gamma F_{T_n} + (1-\gamma)\tilde{F}^\gamma_{n-1}$. In this case, previous tasks are gracefully forgotten at an exponential rate controlled by γ. We account for the damping α term in Eq. 5 by setting $\tilde{F}_0 := \frac{\alpha}{\gamma} I$. In preliminary experiments, we have found $\gamma = 0.9$ to yield consistently good results, and use this value in all presented experiments.

4 Continual Learning Experiments

4.1 Experimental setting

To examine our hypothesis that controlling the optimization trajectory with the co-natural gradient reduces catastrophic forgetting, we follow the experimental procedure from [10]: given a collection of tasks, we create a "validation set" of 3 tasks used to select the best hyper-parameters, and keep the remaining tasks for evaluation. This split is chosen at random and kept the same across all experiments. In most settings, the nature and possibly the number of classes changes from task to task. We account for this by training a separate "task head" for each task: an affine transform projecting the features onto the number of classes, followed by a softmax layer. We apply continual learning only to the remaining, "feature-extraction" part of the model.

We use the four following task suites in our experiments:

- **Split CIFAR**: The CIFAR100 dataset, split into 20 independent 5-way classification tasks. Similarly to [9], we use a smaller version of the ResNet architecture [22]. We train on each task for 10 epochs with batch size 32.
- **Omniglot**: the Omniglot dataset [26] consists of 50 independent character recognition datasets on different alphabet. We adopt the setting of [42] and consider each alphabet as a separate task.[3] On this dataset we use the same small CNN architecture as [42]. We augment the training data by randomly shifting and rotating images and train on each task for 2500 steps (120 to 417 epochs depending on the alphabet) with batch size 32.
- **Split MiniImageNet**: The MiniImageNet dataset, a subset of the popular ImageNet [14] dataset[4]; [46], is split into 20 independent 5-way classification tasks, similarly to Split CIFAR. We use the same smaller ResNet and train on each task for 1500 steps with batch size 32 ($\approx$ 20 epochs).

[3] Note that this is a different setting than the usual meta-learning scenario that Omniglot is used for.

[4] As MiniImageNet was developed as a meta-learning benchmark, its canonical train/test split consists of disjoint classes. We perform a custom transversal split so that the dataset can be used as a 100-way classification task.

Table 1: Average final accuracies and forgetting, with and without the co-natural gradient. Results are reported in percentages (± standard deviation over 5 re-runs). We indicate the best of standard/co-natural in bold and denote statistical significance by underlining ($p < 0.05$).

(a) **Split CIFAR 1.4 long**

	Finetuning	EWC	ER
	Average accuracy ↑		
Standard	33.41 ±1.48	58.97 ±0.95	62.47 ±0.65
Co-natural	**63.94** ±1.78	**61.41** ±0.82	**68.08** ±0.87
	Forgetting ↓		
Standard	44.30 ±1.67	8.73 ±1.14	14.77 ±0.69
Co-natural	**7.92** ±2.16	**4.66** ±0.81	**3.18** ±0.50

(b) **Omniglot 1.4**

Finetuning	EWC	ER
Average accuracy ↑		
21.37 ±3.58	71.06 ±2.16	70.40 ±0.80
75.48 ±0.37	**73.64** ±2.54	**80.68** ±1.71
Forgetting ↓		
71.04 ±3.47	10.47 ±2.37	22.04 ±0.86
3.04 ±1.16	**2.76** ±0.72	**1.70** ±0.30

(c) **Split MiniImageNet 1.4 long**

	Finetuning	EWC	ER
	Average accuracy ↑		
Standard	35.50 ±3.47	63.42 ±0.48	65.47 ±0.88
Co-natural	**65.82** ±3.22	**65.25** ±2.35	**71.16** ±0.59
	Forgetting ↓		
Standard	44.95 ±3.36	8.91 ±0.79	14.21 ±0.64
Co-natural	**9.53** ±4.10	**6.85** ±2.64	**4.91** ±1.09

(d) **Text Classification 1.4**

Finetuning	EWC	ER
Average accuracy ↑		
53.80 ±5.18	62.94 ±1.63	**69.13** ±0.34
62.69 ±1.71	**63.72** ±1.02	64.62 ±1.04
Forgetting ↓		
18.27 ±5.27	1.68 ±0.97	3.11 ±0.51
0.82 ±0.42	**0.30** ±0.12	**0.70** ±0.51

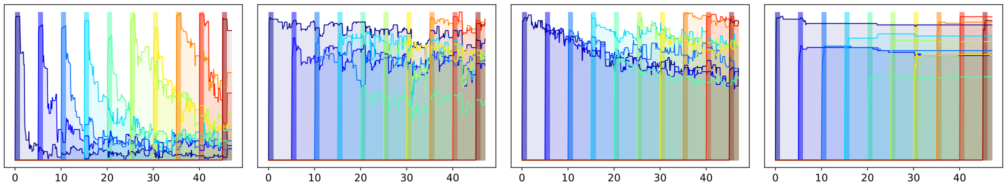
- **Text Classification**: The most recent work on continual learning of language tasks [13] relies on a relatively small set of tasks (5 classification tasks from [55]). This small number of tasks is not amenable to our experimental setup described above, where 3 tasks are reserved for validation. Therefore we assemble a larger collection of text classification datasets from three sources: the GLUE benchmark [48], the SuperGLUE benchmark [47] and the text classification datasets used in [13]. To keep things relatively simple, we only keep tasks that 1. are single sentence or sentence pair classification tasks and 2. have more than 1000 training examples. The exact list of tasks can be found in Appendix A.2. Instructions to download and code to preprocess the data will be made available at `anonymized_url` to facilitate reproduction of our results and future work on continual learning for text classification. Following recent practice in text classification, we use a large model that has already been pre-trained in an unsupervised fashion (specifically `bert-base-uncased`[5] from [15]), and fine-tune the model on the supervised classification tasks. We train on each task for 1000 steps with batch size 16 (from 7 to < 1 epochs due to the large variance in dataset sizes).

We report results using two common metrics for continual learning: **average accuracy**, the accuracy at the end of training averaged over all tasks, and **forgetting**. Forgetting is defined in [8] as the difference in performance on a task between the current model and the best performing model on this task. Formally if A_t^T represents the accuracy on task T at step t of training, the forgetting F_t^T at step t is defined as $F_t^T = \max_{\tau<t} A_\tau^T - A_t^T$. Low forgetting means that the model tend to keep the same level of performance on a task it has previously learned.

We implement the proposed co-natural update rule on top of 3 baselines:

- **Finetuning**: Simply train the model on the task at hand, without any form of regularization.
- **EWC**: Proposed by [24], it is a simple but effective quadratic regularization approach. While neither the most recent nor sophisticate regularization technique, it is a natural baseline for us to compare to in that it also consists in a Fisher-based penalty — albeit in the loss function instead of the optimization dynamics. We also use the rolling Fisher described in §3.4, making our EWC baseline equivalent to the superior online EWC introduced by [42].
- **ER**: Experience replay with a fixed sized episodic memory proposed by [10]. While not directly comparable to EWC in that it presupposes access to data from previous tasks, ER is a simple approach that boasts the best performances on a variety of benchmarks [10]. In all experiments, we use a memory of size 1,000 populated with reservoir sampling.

[5] We use the open-source implementation from [52] with the default configuration.



(a) Finetuning (b) EWC (c) ER (d) Co-natural Finetuning

Figure 2: Evolution of task performance over the course of continual learning on one ordering of Omniglot. For visibility we only show accuracies for every fifth task. The rectangular shaded regions delineate the period during which each task is being trained upon.

Training proceeds as follows: we perform exhaustive search on all possible hyper-parameter configurations using the validation tasks. Every configuration is reran 5 times (3 for text classification) with a different random seed and assigned a score based on the average task score at the end of training (averaged over reruns). We then evaluate the best hyper-parameters by continual training on the evaluation tasks. Results are reported over 5 random restarts, and we control for statistical significance using a paired t-test (we pair together runs with the same task ordering). For all experiments we train with plain stochastic gradient descent. We refer to Appendix A.3 for more details regarding fine-grained design choices.

4.2 Results

The upper half of Tables 1a, 1b, 1c and 1d reports the average accuracy of all the tasks at the end of training (higher is better). We observe that the co-natural gradient always improves greatly over simple finetuning, and occasionally over EWC and ER. We note that simple co-natural finetuning (without any other form of regularization) sometimes matches or exceeds the performance of EWC even though it requires strictly fewer resources (there is no need to store the previous parameters as in EWC, or data in ER).

Even more appreciable is the effect of the co-natural trajectories on forgetting, as shown in the lower half of Table 1. As evidenced by the results in the lowest rows, using the co-natural gradient on top of finetuning and ER consistently results in large drops in forgetting across all datasets. With EWC, the conclusion is more nuanced. While the co-natural gradient systematically reduces forgetting, the effect is not always statistically significant. Finally, we refer to Appendix A.4 for a detailed analysis of the effect of the damping coefficient on forgetting.

To get a qualitative assessment of the learning trajectories that yield such results, we visualize the accuracy curves of 10 out of the 47 evaluation tasks of Omniglot in Figure 2. We observe that previous approaches do poorly at keeping stable levels of performance over a long period of time (especially for tasks learned early in training), a problem that is largely alleviated by the co-natural preconditioning. This seems to come at the cost of more intransigence [8], *i.e.* some of the later tasks are not being learnt properly. In models of fixed capacity, there is a natural trade-off between intransigence and forgetting (see also the "stability-plasticity" dilemma in neuroscience [19]). Our results position the co-natural gradient as a strong low-forgetting/moderate intransigence basis for future work.

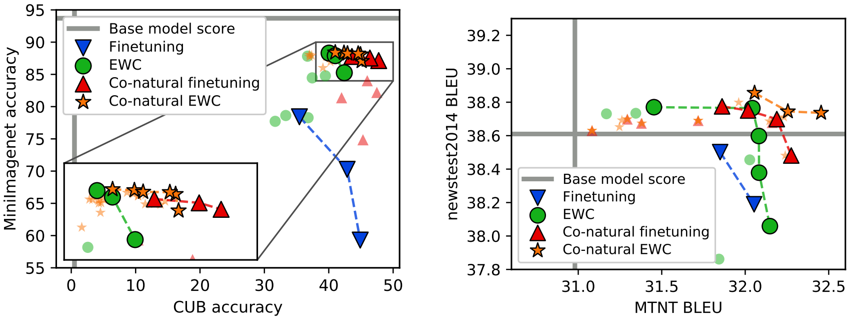
4.3 On the Importance of Re-estimating the Fisher

In our experiments, we make a critical approximation by computing the Fisher information of each task only once. Indeed, there is no theoretical reason for the Fisher to stay the same as the model is fine-tuned, and by keeping a possibly stale Fisher we are losing the local geometric information highlighted in Equation 3.

In order to examine how much this approximation hurts the co-natural gradient, we perform a series of experiments on the split CIFAR dataset, specifically we only look at the performance of the first task and compare three versions of co-natural fine-tuning: one where the Fisher is re-estimated every epoch and one where the Fisher is computed only once (which is the version we use in our experiments above). As evidenced by results in Table 2 (averaged over 10 reruns), while there is a small (but statistically significant) decrease in performance by not re-computing the Fisher, we

Table 2: Degradation in forgetting and accuracy when the Fisher is computed less frequently. All improvements are statistically significant ($p < 0.05$).

	Regular fine-tuning	Computed once	Re-computed every epoch
Final accuracy (↑)	21.88 ± 5.97	67.72 ± 7.17	71.82 ± 4.13
Forgetting (↓)	51.46 ± 6.48	7.84 ± 4.32	3.78 ± 2.10



(a) MiniImageNet to CUB adaptation

(b) WMT to MTNT fine-tuning

Figure 3: Low-resource adaptation results. Source and target task performance are represented on the y and x axes respectively. Pareto optimal points for each method are highlighted and the frontier is represented with dashed lines. Solid gray lines indicate the original source-task-trained model.

are still able to achieve $\approx$ 91% of the forgetting reduction over regular fine-tuning compared to re-computing the Fisher every epoch, at a drastically lower computation cost (and without the need to retain data to re-estimate the Fisher).

5 Low-Resource Adaptation Experiments

In this section we take a closer look at the specific case of adapting a model from a single task to another, when we only have access to a minimal amount of data in the target task. In this case, controlling the learning trajectory is particularly important because the model is being trained on an unreliable sample of the true distribution of the target task, and we have to rely on early-stopping to prevent overfitting. We show that using the co-natural gradient during adaptation helps both at preserving source task performance and reach higher overall target task performance. We perform experiments on two different scenarios:

Image classification We take MiniImagenet as a source task and CUB (a 200-way birds species classification dataset; [50]) as a target task. To guarantee a strong base model despite the small size of MiniImageNet, we start off from a ResNet18 model [22] pretrained on full ImageNet, which we retrofit to MiniImageNet by replacing the last fully connected layer with a linear layer regressed over the MiniImageNet training data. To simulate a low-resource setting, we sub-sample the CUB training set to 2000 images ($\approx$ 10 per class). Scores for these tasks are reported in terms of accuracy.

Machine translation We consider adaptation of an English to French model trained on WMT15 (a dataset of parallel sentences crawled from parliamentary proceedings, news commentary and web page crawls; [6]) to MTNT (a dataset of Reddit comments; [32]). Our model is a Transformer [45] pretrained on WMT15. Similarly to CUB, we simulate a low-resource setting by sub-sampling 1000 sentence pairs as a training set. Scores are reported in terms of BLEU score [34].[6]

Here we do not allow any access to data in the source task when training on the target task. We compare four methods **Finetuning** (our baseline), **Co-natural finetuning**, **EWC** (which has been proven effective for domain adaptation [44]) and **Co-natural EWC**.

Given that different methods might lead to different trade-offs between source and target task performance, with some variation depending on the hyper-parameters (*e.g.* learning rate, regularization

[6] We use sacrebleu [37] with `-tok intl` as recommended by [32].

strength…), we take inspiration from [44] and graphically report results for all hyper-parameter configuration of each method on the 2 dimensional space defined by the score on source and target tasks.[7] Additionally, we highlight the Pareto frontier of each method *i.e.* the set of configurations that are not strictly worse than any other configuration for the same model.

The adaptation results for both scenarios are reported in Figure 3. We find that in both cases, the co-natural gradient not only helps preserving the source task performance, but to some extent it also allows the model to reach better performance on the target task as well. We take this to corroborate our starting hypothesis: while introducing a regularizer does help, controlling the optimization dynamics actively helps counteract overfitting to the very small amount of training data, because the co-natural pre-conditioning makes it harder for stochastic gradient descent to push the model towards directions that would also hurt the source task.

6 Conclusion

We have presented the co-natural gradient, a technique that regularizes the optimization trajectory of models trained in a continual setting. We have shown that the co-natural gradient stands on its own as an efficient approach for overcoming catastrophic forgetting, and effectively complements and stabilizes other existing techniques at a minimal cost. We believe that the co-natural gradient — and more generally, trajectory regularization — can serve as a solid bedrock for building agents that learn without forgetting.

Broader Impact

The work presented in this paper consists of an algorithmic improvement that inscribes itself in a long line of work in preventing catastrophic forgetting in neural models (see Section 2.2 for a brief overview of the relevant literature), and as such its impact inherits from that of this entire area of the literature. In particular, reducing catastrophic forgetting makes it feasible to keep models over larger periods of time without re-training from scratch, with potential computation and energy benefits. More specifically, our work makes it possible to keep high levels of performance without retaining training data on previous tasks or domains, which is attractive from the point of view of preserving privacy (as it is not necessary to keep training data around after training). This advantage however should be tempered by the possibility of extracting training data from trained models (as explored in [7] for example): by preventing forgetting, it is unclear how much of the training data is in fact preserved and recoverable in the model.

References

[1] H. Ahn, D. Lee, S. Cha, and T. Moon. Uncertainty-based continual learning with adaptive regularization. In *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2019.

[2] R. Aljundi, L. Caccia, E. Belilovsky, M. Caccia, L. Charlin, and T. Tuytelaars. Online continual learning with maximally interfered retrieval. In *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2019.

[3] R. Aljundi, M. Lin, B. Goujaud, and Y. Bengio. Gradient based sample selection for online continual learning. In *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2019.

[4] S.-i. Amari. Neural learning in structured parameter spaces-natural riemannian gradient. In *Proceedings of the 9th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 127–133, 1997.

[5] L. Bentivogli, P. Clark, I. Dagan, and D. Giampiccolo. The fifth pascal recognizing textual entailment challenge. In *TAC*, 2009.

[6] O. Bojar, R. Chatterjee, C. Federmann, B. Haddow, M. Huck, C. Hokamp, P. Koehn, V. Logacheva, C. Monz, M. Negri, M. Post, C. Scarton, L. Specia, and M. Turchi. Findings of the

[7] For CUB we report the average accuracy of every configuration over 5 runs, each with a different subset.

2015 workshop on statistical machine translation. In *Proceedings of the 10th Workshop on Statistical Machine Translation (WMT)*, pages 1–46, 2015.

[7] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 267–284, 2019.

[8] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[9] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny. Efficient lifelong learning with a-gem. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[10] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. K. Dokania, P. H. Torr, and M. Ranzato. Continual learning with tiny episodic memories. *arXiv preprint arXiv:1902.10486*, 2019.

[11] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. 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 (NAACL-HLT)*, pages 2924–2936, 2019.

[12] I. Dagan, O. Glickman, and B. Magnini. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer, 2005.

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

[14] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 22nd IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 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 (NAACL-HLT)*, 2018.

[16] P. Dhar, R. V. Singh, K.-C. Peng, Z. Wu, and R. Chellappa. Learning without memorizing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5138–5146, 2019.

[17] W. B. Dolan and C. Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the The 3rd International Workshop on Paraphrasing (IWP)*, 2005. URL `http://aclweb.org/anthology/I05-5002`.

[18] D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics, 2007.

[19] S. T. Grossberg. *Studies of Mind and Brain: Neural Principles of Learning, Perception, Development, Cognition, and Motor Control*, volume 70. Springer Science & Business Media, 1982.

[20] A. Gulli. Ag's corpus of news articles. `http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html`, 2005.

[21] R. B. Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, 2006.

[22] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[23] S. Iyer, N. Dandekar, and K. Csernai. First quora dataset release: Question pairs. `https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs`, 2017.

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

[25] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[26] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350:1332–1338, 2015.

[27] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web – Interoperability, Usability, Applicability*, 6(2): 167–195, 2015.

[28] Z. Li and D. Hoiem. Learning without forgetting. *Proceedings of the 14th European Conference on Computer Vision (ECCV)*, 2016.

[29] D. Lopez-Paz and M. Ranzato. Gradient episodic memory for continual learning. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 6467–6476, 2017.

[30] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys)*, pages 165–172, 2013.

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

[32] P. Michel and G. Neubig. MTNT: A testbed for machine translation of noisy text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 543–553, 2018.

[33] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner. Variational continual learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[34] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, 2002.

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

[36] R. Pascanu and Y. Bengio. Revisiting natural gradient for deep networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2013.

[37] M. Post. A call for clarity in reporting BLEU scores. In *Proceedings of the 3rd Conference on Machine Translation (WMT)*, pages 186–191, 2018.

[38] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392, 2016.

[39] R. Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

[40] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2001–2010, 2017.

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

[42] J. Schwarz, W. Czarnecki, J. Luketina, A. Grabska-Barwinska, Y. W. Teh, R. Pascanu, and R. Hadsell. Progress & compress: A scalable framework for continual learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4535–4544, 2018.

[43] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the*

2013 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1631–1642, 2013.

[44] B. Thompson, J. Gwinnup, H. Khayrallah, K. Duh, and P. Koehn. Overcoming catastrophic forgetting during domain adaptation of neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017.

[46] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 3630–3638, 2016.

[47] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NIPS)*, pages 3261–3275, 2019.

[48] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[49] A. Warstadt, A. Singh, and S. R. Bowman. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*, 2018.

[50] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[51] A. Williams, N. Nangia, and S. Bowman. 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 (NAACL-HLT)*, pages 1112–1122, 2018.

[52] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv preprint arXiv:1910.03771*, 2019.

[53] Yelp. Yelp dataset challenge. `https://www.yelp.com/dataset/challenge`, 2015.

[54] F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3987–3995, 2017.

[55] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 649–657, 2015.

A Appendix

A.1 Derivations

A.1.1 The Standard Gradient Update

We derive the standard gradient update by solving the Lagrangian $\mathbb{L}$ in Eq. 2 for δ. Given that its first and second derivatives with respect to δ are:

$$\begin{aligned}\nabla\mathbb{L} &= \nabla\mathcal{L}_T + 2\mu\delta \\ \nabla^2\mathbb{L} &= 2\mu I\end{aligned}$$

the problem is trivially strictly convex and its global minimizer δ^* satisfies:

$$\nabla\mathbb{L}\big|_{\delta^*} = 0 \iff \delta^* = -\frac{1}{2\mu}\nabla\mathcal{L}_T$$

□

A.1.2 Equivalence of the Hessian of the KL Divergence and the Fisher Information Matrix

To simplify notation, let us perform the change of variables $\theta+\delta \to x$. We show that the Hessian of the KL coincides with the Fisher on θ: in other words, $\nabla^2\mathrm{KL}(p_\theta\|p_x)\big|_{x=\theta} = \mathbb{E}_{p_\theta}[(\nabla\log p_\theta)(\nabla\log p_\theta)^\intercal]$. Under mild regularity assumptions[8], we can write (all derivatives are taken with respect to variable x):

$$\begin{aligned}\nabla^2\mathrm{KL}(p_\theta\|p_x) &= \underbrace{\nabla^2\mathbb{E}_{p_\theta}[\log p_\theta]}_{=0} - \nabla^2\mathbb{E}_{p_\theta}[\log p_x] \\ &= -\mathbb{E}_{p_\theta}[\nabla^2\log p_x]\end{aligned}$$

Now note that $\nabla^2\log p_x$ can be rewritten via standard derivatives manipulations as $\frac{\nabla^2 p_x}{p_x} - \frac{(\nabla p_x)(\nabla p_x)^\intercal}{p_x^2}$. This leads to:

$$\nabla^2\mathrm{KL}(p_\theta\|p_x) = -\mathbb{E}_{p_\theta}\left[\frac{\nabla^2 p_x}{p_x}\right] + \mathbb{E}_{p_\theta}\left[\left(\frac{\nabla p_x}{p_x}\right)\left(\frac{\nabla p_x}{p_x}\right)^\intercal\right]$$

When taken at θ, the first term evaluates to[9] :

$$\begin{aligned}\mathbb{E}_{p_\theta}\left[\frac{\nabla^2 p_x}{p_x}\right]\Bigg|_{x=\theta} &= \left[\int p_\theta(z)\frac{\nabla^2 p_x(z)}{p_x(z)}dz\right]\Bigg|_{x=\theta} \\ &= \int p_\theta(z)\frac{\nabla^2 p_\theta(z)}{p_\theta(z)}dz \\ &= \int \nabla^2 p_\theta(z)dz \\ &= \nabla^2\underbrace{\int p_\theta(z)dz}_{=1} = 0\end{aligned}$$

By using the identity $\frac{\nabla p_x}{p_x} = \nabla\log p_x$ and evaluating at $x=\theta$, the second term gives us:

[8] Essentially allowing us to interchange derivatives and integrals.

[9] We abuse notation and write $\nabla p_x\big|_{x=\theta}$ as ∇p_θ

$$\nabla^2 \mathrm{KL}(p_\theta \| p_x)\big|_{x=\theta} = \mathbb{E}_{p_\theta}[(\nabla \log p_\theta)(\nabla \log p_\theta)^\intercal]$$

□

A.1.3 Obtaining the Co-natural Update (Equation 5)

We solve the Lagrangian from Eq. 4 in a similar fashion as in A.1.1. First we compute its gradient and Hessian with respect to δ:

$$\begin{aligned} \nabla \mathbb{L} &= \nabla \mathcal{L}_T + 2\mu\delta + \nu F_\theta^S \delta \\ &= \nabla \mathcal{L}_T + (\nu F_\theta^S + 2\mu I)\delta \\ \nabla^2 \mathbb{L} &= (\nu F_\theta^S + 2\mu I) \end{aligned}$$

While not as straightforwardly as the one in A.1.1, this problem is also strongly convex: indeed F_θ^S is positive semi-definite (as an expectation of PSD matrices) and the addition of μI ensures that $\nabla^2\mathbb{L}$ is positive definite. We find the unique solution by solving:

$$\begin{aligned} \nabla \mathbb{L}\big|_{\delta^*} = 0 &\iff \nabla \mathcal{L}_T + (\nu F_\theta^S + 2\mu I)\delta^* = 0 \\ &\iff \delta^* = -[\nu F_\theta^S + 2\mu I]^{-1} \nabla \mathcal{L}_T \end{aligned}$$

Set $\lambda := \frac{1}{\nu}$ and $\alpha := \frac{\mu}{\nu}$ to get Eq. 5 □

A.2 Detailed Description of the Text Classification Task Suite

As mentioned in Section 4, we build a collection of text classification tasks by selecting sentence or sentence pair classification tasks with training datasets of size greater than 1,000 samples from three sources: GLUE [48], SuperGLUE [47] and the datasets used in [13] (originally used in [55]). Specifically, we use the following tasks from each dataset (reported with the training data size):

- **GLUE**
  - CoLA [49]: 8.6K
  - MultiNLI [51]: 392.7K
  - MRPC [17]: 3.7K
  - QNLI [38]: 108.4K
  - QQP [23]: 363.8K
  - RTE [12, 21, 18, 5]: 2.5K
  - SST-2 [43]: 67.3K
- **SuperGLUE**
  - BoolQ [11]: 9.4K
- [13]
  - AG News [20] (115.0K)
  - Amazon Reviews Full [30] (115.0K)
  - DBPedia [27] (115.0K)
  - Yahoo Answers [55] (115.0K)
  - Yelp Reviews Full [53] (115.0K)

Note that the RTE dataset from SuperGLUE also satisfies our task type and dataset size constraint, however according to [47] it is an exact duplicate of GLUE's RTE dataset, therefore we don't include it. [13] performed some mild preprocessing and subsampling of the datasets in [55], however they did not release the final data. We perform a similar preprocessing step: for datasets where each sample consists in several pieces of text (typically review title and body in the Amazon and Yelp datasets), we concatenate all into one utterance, joined with a period and a space ("`. `"). We subsample the datasets to 115K training samples, 5K validation samples and 7.6K test samples.

We randomly select three out of these 13 tasks to serve as the validation split upon which to perform hyper-parameter search, namely: **BoolQ**, **MRPC** and **SST-2**.

Since test sets are not available for the GLUE and SuperGLUE benchmark, we compute the final scores on the validation datasets. Note that in all our experiments, validation data is not used during training, therefore there is no leakage of the ultimate evaluation dataset in the training procedure

A.3 Additional Experimental Settings for Continual Learning

This section is intended to facilitate the reproduction of our results. The full details can be found with our code at `anonymized_url`.

A.3.1 Split CIFAR

We split the dataset into 20 disjoint sub-tasks with each 5 classes, 2500 training examples and 500 test examples. This split, performed at random, is kept the same across all experiments, only the order of these tasks is changed. During continual training, we train the model for one epoch on each task with batch size 10, following the setup in [9].

A.3.2 Omniglot

We consider each alphabet as a separate task, and split each task such that every character is present 12, 4 and 4 times in the training, validation and test set respectively (out of the 20 images for each character). During continual training, we train for 2500 steps with batch size 32 (in keeping with [42]). We ignore the validation data and simply evaluate on the test set at the end of training.

A.3.3 MiniImageNet

We split the dataset into 20 disjoint sub-tasks with each 5 classes. During continual training, we train for 500 steps with batch size 32 (in keeping with [42]). We ignore the validation data and simply evaluate on the test set at the end of training.

A.3.4 Grid-search parameters

For each all image-related tasks, we perform grid-search over the following parameter values:

- Learning rate (all methods): 0.1, 0.03, 0.01
- EWC regularization strength (EWC, Co-natural EWC): 0.5, 1, 5
- Fisher damping coefficient (Co-natural finetuning, Co-natural EWC, Co-natural ER): 0, 1, 0.1

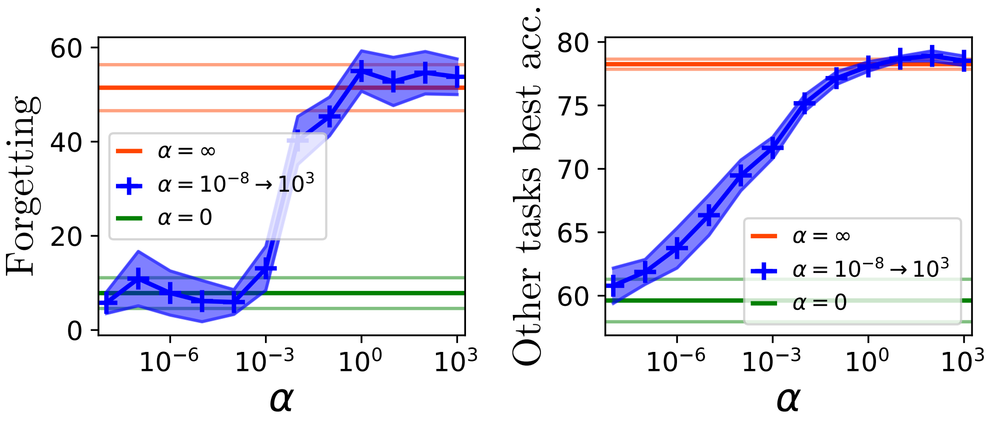
For text classification with BERT specifically, preliminary experiments showed that all methods benefitted from lower learning rate as well as lower regularization (likely due to the fact that optimization is starting from a pretrained model). We therefore use the following values:

- Learning rate (all methods): 0.05, 0.01, 0.005
- EWC regularization strength (EWC, Co-natural EWC): 0.5, 0.1, 0.05
- Fisher damping coefficient (Co-natural finetuning, Co-natural EWC, Co-natural ER): 1, 10, 100

For ER, we simply set the replay batch size to the same value as standard training (10 and 32 for Split CIFAR and Omniglot respectively). Note that whenever applicable, we re-normalize the diagonal Fisher so that the sum of its weights is equal to the number of parameters in the model. This is so that the hyper-parameter choice is less dependent on the size of the model. In particular this means that the magnitude of each diagonal element is much bigger, which is why we do grid-search over smaller regularization parameters for EWC than is common in the literature.

A.4 Sensitivity to damping

The main hyper-parameter for the co-natural gradient is the damping parameter α from Eq. 5. In our previous experiments, the value of α is chosen according to grid search on a small number of tasks. While this is a realistic setting for practical applications, in this section we perform a smaller, targeted experiment to examine the effect of α on catastrophic forgetting.



(a) Forgetting (b) Intransigence

Figure 4: Effect of damping on forgetting and intransigence. The horizontal axis (measuring α) follows a logarithmic scale. The width of the bands on each side of the curves represent the 95% confidence intervals over the 10 re-runs.

We focus on the 17 evaluation tasks of the Split CIFAR dataset and set the learning rate to 0.1. We observe how much the model "forgets" about the first task it has observed after training on all others. Specifically, we train on the first task, compute its Fisher F_1, and then proceed to train on the 16 remaining tasks with the co-natural gradient using $F_1 + \alpha$ (in particular we do not regularize for the other tasks). We observe how the value of alpha affects the final forgetting of the first task at the end of training, as well as the model's ability to learn new tasks (sometimes referred to as "intransigence" in the literature [8]). As a measure of the latter, we report the maximum accuracy achieved by the model averaged over the 16 remaining tasks.

We evaluate values of α from 10^{-7} to 10^{2} (following a geometric progression), as well as the two extremal values $\alpha = 0$ ("pure" co-natural gradient[10]) and $\alpha = \infty$ (simple finetuning). All results are averaged over 10 random restarts (with different model initialization and task order).

We observe in Figure 4a that forgetting monotonically increases with the damping coefficient. Similarly, Figure 4b shows how increasing α results in lower intransigence. While these two observations are expected, it is interesting to note the existence of a "sweet spot" around $\alpha \in [10^{-7}, 10^{-4}]$ where damped co-natural gradient is significantly less intransigent than the un-damped co-natural gradient while simultaneously not being significantly less robust to forgetting (all hypotheses are tested for with $p < 0.05$).

[10] As mentioned in Section 3, we do actually add a small damping term $\varepsilon = 10^{-12}$ to all experiments.