UNIFYING REGULARISATION METHODS FOR CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Continual Learning addresses the challenge of learning a number of different distributions sequentially. The goal of maintaining knowledge of earlier distributions without re-accessing them starkly conflicts with standard SGD training for artificial neural networks. An influential method to tackle this are so-called regularisation approaches. They measure the importance of each parameter for modelling a given distribution and subsequently protect important parameters from large changes. In the literature, three ways to measure parameter importance have been put forward and they have inspired a large body of follow-up work. Here, we present strong theoretical and empirical evidence that these three methods, Elastic Weight Consolidation (EWC), Synaptic Intelligence (SI) and Memory Aware Synapses (MAS), all approximate the Fisher Information. Only EWC intentionally relies on the Fisher, while the other two methods stem from rather different motivations. We find that for SI the relation to the Fisher – and in fact its performance – is due to a previously unknown bias. Altogether, this unifies a large body of regularisation approaches. It also provides the first theoretical explanation for the effectiveness of SI- and MAS-based algorithms and offers theoretically justified versions of these algorithms. From a practical viewpoint, our insights offer computational speed-ups and uncover conditions needed for different algorithms to work.

1 INTRODUCTION

Despite considerable progress, many gaps between biological and machine intelligence remain. Animals, for example, flexibly learn new tasks throughout their lives, while at the same time maintaining robust memories of previous knowledge. This ability conflicts with traditional training procedures for artificial neural networks, which overwrite previous skills when optimizing new tasks (McCloskey & Cohen, 1989; Goodfellow et al., 2013). The field of continual learning is dedicated to mitigate this crucial shortcoming of machine learning. It exposes neural networks to a sequence of distinct tasks. During training new tasks, the algorithm is not allowed to revisit old data, but should retain previous skills while remaining flexible enough to also acquire new knowledge.

An influential line of work to approach this challenge was introduced by Kirkpatrick et al. (2017), who proposed the first regularisation-algorithm, Elastic Weight Consolidation (EWC). After training a task, EWC measures the importance of each parameter and introduces an auxiliary loss penalising large changes in important parameters. Naturally, this raises the question of how to measure this 'importance'. While EWC uses the diagonal Fisher Information, two alternatives have been proposed: Synaptic Intelligence (SI, Zenke et al. (2017)) aims to attribute the decrease in loss during training to individual parameters and Memory Aware Synapses (MAS, Aljundi et al. (2018)) introduces a heuristic measure of output sensitivity. Together, these three approaches have inspired many further regularisation-based approaches, including combinations of them (Chaudhry et al., 2018), refinements (Huszár, 2018; Ritter et al., 2018; Chaudhry et al., 2018; Yin et al., 2020), extensions (Schwarz et al., 2018; Liu et al., 2018; Park et al., 2019; Lee et al., 2020) and applications in different continual learning settings (Aljundi et al., 2019) as well as different domains of machine learning (Lan et al., 2019). Almost every new continual learning method compares to at least one of EWC, SI and MAS.

Despite their popularity and influence, basic practical and theoretical questions regarding these algorithms had previously been unanswered. Notably, it was unkown how similar these importance measures are. Additionally, for SI and MAS (and their follow-ups) there was no solid theoretical

understanding of their effectiveness. Here, we close both these gaps through a theoretical analysis confirmed by a series of carefully designed experiments on standard continual learning benchmarks. Our main findings can be summarised as follows:

(a) We show that MAS is almost identical to a natural variant of the Fisher Information.

(b) We show that SI's importance approximation is biased, that the bias is responsible for SI's performance and that the bias, like MAS, is tightly linked to the Fisher Information.

(c) Our fine-grained analysis leads to new baselines, including theoretically justified versions of MAS and SI as well as a significantly faster version of EWC with equal performance.

(d) Together, (a) and (b) show that all three regularisation approaches (and their follow-ups) rely on the Fisher Information. This unifies a large body of regularisation literature. It also gives the first solid theoretical justification for the effectiveness of SI- and MAS-based algorithms.

2 Related Work

The problem of catastrophic forgetting in neural networks has been studied for many decades (McCloskey & Cohen, 1989; Ratcliff, 1990; French, 1999). In the context of deep learning, it received more attention again (Goodfellow et al., 2013; Srivastava et al., 2013). All previous work we are aware of proposes new or modified algorithms to tackle continual learning. No attention has been directed towards understanding or unifying existing methods and we hope that our work will not remain the only effort of this kind. We now review the broad body of continual learning algorithms. Following (Parisi et al., 2019), they are often categorised into regularisation-, replay- and architectural approaches. Most *regularisation methods* will be reviewed more closely in the next section. There we cover regularisation for standard neural nets, but the same idea has also been applied to bayesian neural networks (Nguyen et al., 2017; Swaroop et al., 2019). An alternative approach (Mirzadeh et al., 2020) does not penalize deviating from important parameters, but rather modifies training so that the network is robust to such deviations. *Replay methods* refer to algorithms which either store a small sample or generate data of old distributions and use this data while training on new methods (Rebuffi et al., 2017; Lopez-Paz & Ranzato, 2017; Shin et al., 2017; Kemker & Kanan, 2017). These approaches can be seen as investigating how far standard i.i.d.-training can be relaxed towards the (highly non-i.i.d.) continual learning setting without losing too much performance. They are interesting, but usually circumvent the original motivation of continual learning to maintain knowledge *without* accessing old distributions. Intriguingly, the most effective way to use old data appears to be simply replaying it, i.e. mimicking training with i.i.d. batches sampled from all tasks simultaneously (Chaudhry et al., 2019). *Architectural methods* extend the network as new tasks arrive (Fernando et al., 2017; Li et al., 2019; Schwarz et al., 2018; Golkar et al., 2019; von Oswald et al., 2019). This can be seen as a study of how old parts of the network can be effectively used to solve new tasks and touches upon transfer learning. Typically, it avoids the challenge of integrating new knowledge into an existing networks. Finally, van de Ven & Tolias (2019); Hsu et al. (2018); Farquhar & Gal (2018) point out that different continual learning scenarios and assumptions with varying difficulty were used across the literature.[1]

3 Review of Existing Regularisation Methods

3.1 Formal Description of Continual Learning

In continual learning we are given K datasets $\mathcal{D}_1, \ldots, \mathcal{D}_K$ sequentially. When training a neural net with N parameters $\theta \in \mathbb{R}^N$ on dataset $\mathcal{D}_k$, we have no access to the previously seen datasets $\mathcal{D}_{1:k-1}$. However, at test time the algorithms is tested on all K tasks, usually the average accuracy is taken as measure of the algorithms performance, but see also Chaudhry et al. (2018).

3.2 Common Framework for Regularisation Methods

Regularisation based approaches introduced in Kirkpatrick et al. (2017); Huszár (2018) protect previous memories by modifying the loss function L_k related to dataset $\mathcal{D}_k$. Let us denote the

[1] In the appendix, we critically review and question some of their experimental results.

parameters obtained after finishing training on task k by $\theta^{(k)}$ and let $\omega^{(k)} \in \mathbb{R}^N$ be the parameters' *importances*. When training on task k, regularisation methods use the loss

$$\tilde{L}_k = L_k + c \cdot \sum_{i=1}^{N} \omega_i^{(k-1)} \left(\theta_i - \theta_i^{(k-1)}\right)^2$$

where $c > 0$ is a hyperparameter. The first term L_k is the standard (e.g. cross entropy) loss of task k. The second term ensures that the parameters do not move away too far from their previous values. Usually, $\omega_i^{(k)} = \omega_i^{(k-1)} + \omega_i$, where ω_i is the importance for the most recently learned task k.

3.3 Review of EWC, MAS, SI and follow-ups

Elastic Weight Consolidation (Kirkpatrick et al., 2017) uses the diagonal of the Fisher Information as importance measure. It is evaluated at the end of training a given task. To define the *Fisher Information Matrix* F, we omit the task index k and simply consider a dataset $\mathcal{X}$. For each datapoint $X \in \mathcal{X}$ the network predicts a probability distribution $\mathbf{q}_X$ over the set of labels L, where we suppress dependence of q_X on θ for simplicity of notation. We denote the predicted probability of class y by $\mathbf{q}_X(y)$ for each $y \in L$. Let us assume that we minimize the negative log-likelihood $\log \mathbf{q}_X(y)$ and write $g(X, y) = -\frac{\partial \log \mathbf{q}_X(y)}{\partial \theta}$ for its gradient. Then the Fisher Information F is given by

$$F = \mathbb{E}_{X \sim \mathcal{X}} \mathbb{E}_{y \sim \mathbf{q}_X} \left[g(X, y) g(X, y)^T \right] = \mathbb{E}_{X \sim \mathcal{X}} \sum_{y \in L} \mathbf{q}_X(y) \cdot g(X, y) g(X, y)^T \tag{1}$$

Taking only the **diagonal Fisher** means $\omega_i(\text{EWC}) = \mathbb{E}_{X \sim \mathcal{X}} \mathbb{E}_{y \sim \mathbf{q}_X} \left[g_i(X, y)^2 \right]$.
The *Empirical Fisher Information* replaces the expectation over $y \sim \mathbf{q}_X$ by the (deterministic) label y given by the labels of the dataset. We also define the *Predicted Fisher Information* as taking the maximum-likelihood predicted label, i.e. the argmax of $\mathbf{q}_X(\cdot)$. When the network classifies images correctly and confidently, these three versions clearly become increasingly similar.
Under the assumption that the learned label distribution $\mathbf{q}_X$ is the real label distribution the Fisher equals the Hessian of the loss (Martens, 2014; Pascanu & Bengio, 2013). Its use for continual learning has a clean, bayesian, theoretical interpretation (Kirkpatrick et al., 2017; Huszár, 2018).

Memory Aware Synapses (Aljundi et al., 2018) heuristically argues that the sensitivity of the final layer with respect to a parameter should be used as the parameter's importance. This sensitivity is evaluated after training a given task. Denoting, as before, the final layer of learned probabilities by $\mathbf{q}_X$, this is implemented as (using a euclidean norm)

$$\omega_i(\text{MAS}) = \mathbb{E}_{X \sim \mathcal{X}} \left[\left| \frac{\partial \|\mathbf{q}_X\|^2}{\partial \theta_i} \right| \right],$$

Synaptic Intelligence (Zenke et al., 2017) approximates the contribution of each parameter to the decrease in loss and uses this contribution as importance. To formalise the 'contribution of a parameter', let us denote the parameters at time t by $\theta(t)$ and the loss by $L(t)$. If the parameters follow a smooth trajectory in parameter space, we can write the *decrease* in loss from time 0 to T as

$$L(0) - L(T) = -\int_{\theta(0)}^{\theta(T)} \frac{\partial L(t)}{\partial \theta} \theta'(t) dt = -\sum_{i=1}^{N} \int_{\theta_i(0)}^{\theta_i(T)} \frac{\partial L(t)}{\partial \theta_i} \theta_i'(t) dt. \tag{2}$$

The i-th summand in (2) can be interpreted as the contribution of parameter θ_i to the decrease in loss. While we cannot evaluate the integral precisely, we can use a first order approximation to obtain the importances. To do so, we write $\Delta_i(t) = (\theta_i(t+1) - \theta_i(t))$ for an approximation of $\theta_i'(t)dt$ and get

$$\tilde{\omega}_i(\text{SI}) = \sum_{t=0}^{T-1} \frac{\partial L(t)}{\partial \theta_i} \cdot \Delta_i(t). \tag{3}$$

Thus, we have $\sum_i \tilde{\omega}_i(SI) \approx L(0) - L(T)$. In addition, SI rescales its importances as follows[2]

$$\omega_i(\text{SI}) = \frac{\max\{0, \tilde{\omega}_i(\text{SI})\}}{(\theta_i(T) - \theta_i(0))^2 + \xi}. \tag{4}$$

[2]Note that the $\max(0, \cdot)$ is not part of the description in (Zenke et al., 2017). However, we needed to include it to reproduce their results. A similar mechanism can be found in the official SI code.

Zenke et al. (2017) use additional assumptions to justify this importance. One of the them – using full batch gradient descent – is violated in practice and we will show that this has an important effect.

Follow up work. We presented a version of EWC due to Huszár (2018) and tested in (Chaudhry et al., 2018; Schwarz et al., 2018). It is theoretically more sound and was shown to perform better. Chaudhry et al. (2018) combine EWC with a 'KL-rescaled'-SI. Ritter et al. (2018) use a block-diagonal (rather than diagonal) approximation of the Fisher; Yin et al. (2020) use the full Hessian matrix for small networks. Liu et al. (2018) rotate the network to diagonalise the most recent Fisher Matrix, Park et al. (2019) modify the loss of SI and Lee et al. (2020) aim to account for batch-normalisation.

4 Memory Aware Synapses (MAS) approximates Absolute Fisher

Here, we explain why – despite its different motivation – MAS is almost linearly dependent of the **diagonal Absolute Fisher**, a variant of the diagonal Fisher taking absolute values of gradients rather than squares. We first present the theory and then carry out careful empirical confirmations of the underlying assumptions. For a tabular summary of algorithms and baselines see Appendix B.1.

4.1 Theoretical Relation of MAS and Absolute Fisher

We take a closer look at the definition of the importance of MAS. We apply linearity of derivatives, the chain rule and write $y_0 = \arg\max \mathbf{q}_X$ to obtain (omitting expectation over $X \sim \mathcal{X}$ for brevity)

$$\textbf{MAS:} \quad \left|\frac{\partial \|\mathbf{q}\|^2}{\partial \theta}\right| = 2\left|\sum_{y \in Y} \mathbf{q}(y) \frac{\partial \mathbf{q}(y)}{\partial \theta}\right| \approx 2\left|\mathbf{q}(y_0) \frac{\partial \mathbf{q}(y_0)}{\partial \theta}\right| = 2\mathbf{q}^2(y_0)\left|\frac{\partial \log \mathbf{q}(y_0)}{\partial \theta}\right| \quad (5)$$

Here we made the assumption that the sum is dominated by its maximum-likelihood label y_0, which should be the case if the network classifies its images confidently, i.e. $\mathbf{q}(y_0) \gg \mathbf{q}(y)$ for $y \neq y_0$. Recall that the importance is measured at the end of training, so that our assumption is justified if the task has been learned successfully. Using the same assumption for the Absolute Fisher we obtain

$$\textbf{Abs. Fisher:} \quad \mathbb{E}_{y \sim \mathbf{q}_X}\left[|g(X, y)|\right] \approx \mathbf{q}(y_0)|g(X, y_0)| = \mathbf{q}(y_0)\left|\frac{\partial \log \mathbf{q}(y_0)}{\partial \theta}\right|. \quad (6)$$

This shows the similarity between MAS and the Absolute Fisher. The only difference is a factor of $2\mathbf{q}_X(y_0)$, which we can assume to be approximately constant as the model learned to classify training images confidently. Even with a pessimistic guess that $\mathbf{q}_X(y_0)$ is in a range of 0.5 (rather inconfident) and 1.0 (absolutely confident), the two measures would be highly correlated.

4.2 Empirical Relation of MAS and Absolute Fisher

Our theoretical derivation used two approximations, both of which we now explicitly check empirically on standard benchmarks; see Section 6 for a summary of our experimental setup.

Max-Likeilhood Approximation First, in equation 5 we assumed that sum over the labels is dominated by its maximum likelihood label y_0. To assess the validity of this, we directly compare the entire sum (MAS) with the part only using the max-likelihood label (MAS-MAX). The results in Figure 1 show scatter plots and a summary of the pearson correlations between MAS and MAS-MAX on each task. On P-MNIST the correlations range from 0.95 to 0.99, on CIFAR they are always above 0.99. This shows that MAS and MAS-MAX are nearly equal as claimed.

Absolute Fisher Approximation Second, we assumed that the factor $2\mathbf{q}_X(y_0)$ distinguishing MAS-MAX (RHS of equation 5) and Absolute Fisher is roughly constant. Recall that even if $\mathbf{q}_X(y_0)$ varied between 0.5 and 1.0, the two measures would have large correlations. The data in Figure 1 shows that we are far above this pessimistic regime with correlations between MAS-MAX and AF always above 0.98, showing that MAS-MAX indeed approximates AF. The data also shows that the resulting correlation between MAS and AF is large (between 0.9-0.97 on MNIST, >0.99 on CIFAR), confirming that MAS is an approximation of AF.

Continual Learning Performance Experiment As an additional check of our hypothesis, we implemented continual learning baselines based on MAS, MAS-MAX, AF and compared their performances. The results in Table 1(1) show that all baselines have the same performance, further supporting the claim that MAS is effective because it approximates the Absolute Fisher.

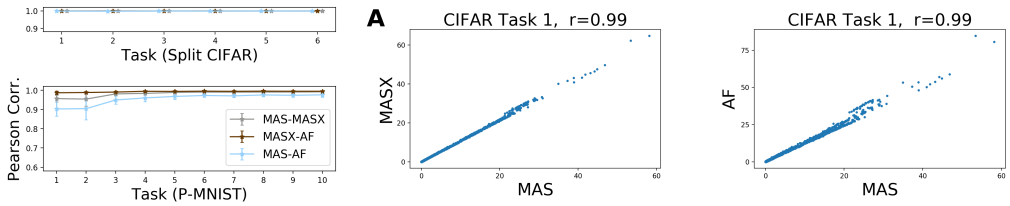

Figure 1: **Empirical Relation between MAS and baselines MASX, Absolute Fisher. Left**: Summary of Pearson Correlations (top: CIFAR, bottom: MNIST), error bars represent std over 4 runs. **Mid & Right**: Example scatter plots of importance measures. Each point in the scatter plot corresponds to one weight of the net. A straight line through the origin corresponds to two importance measures being multiples of each other as was predicted by our theoretical analysis.

5 Synaptic Intelligence approximates (Online) Absolute Fisher

Here, we explain why the importance of SI is related to the Absolute Fisher, despite the apparent contrast between SI's path integral and the Fisher Information. The theoretical part first identifies the bias of SI when approximating the path integral. It then relates the bias to the Absolute Fisher. As before, we carefully design baselines to validate each assumption of our analysis empirically.

5.1 Bias of Synaptic Intelligence

To calculate $\omega(\text{SI})$ (equation 3), we need to calculate the product $p = \frac{\partial L(t)}{\partial \theta} \cdot \Delta(t)$ at each step t. Since evaluating the full gradient $\frac{\partial L(t)}{\partial \theta}$ is prohibitively expensive, SI uses a stochastic mini-batch gradient. The resulting estimate of p is biased since the same mini-batch is used for the parameter update $\Delta(t)$ and the estimate of $\frac{\partial L(t)}{\partial \theta}$.

We now give the calculations detailing the argument above. For ease of exposition, let us assume that the network is optimized using vanilla SGD with learning rate 1. Given a mini-batch, denote its gradient estimate by $g + \sigma$, where $g = \frac{\partial L(t)}{\partial \theta}$ denotes the real gradient and σ the mini-batch noise. The parameter update equals $\Delta(t) = g + \sigma$. Thus, our product p should be $p = g \cdot (g + \sigma)$. However, using $g + \sigma$, which was used for the parameter update, to estimate $\frac{\partial L(t)}{\partial \theta_i}$ results in $p_{biased} = (g + \sigma)^2$. Thus, the gradient noise introduces a bias of $\mathbb{E}[\sigma^2 + \sigma g] = \mathbb{E}[\sigma^2]$.

Unbiased Synaptic Intelligence. Having understood the bias, we can design an unbiased estimate by using two independent mini-batches to calculate the parameter update and to estimate g. This way we get $\Delta(t) = g + \sigma$ and an estimate $g + \sigma'$ for g from an independent mini-batch with independent noise σ'. We obtain $p_{unbiased} = (g + \sigma') \cdot (g + \sigma)$ which in expectation equals $p = g \cdot (g + \sigma)$. Based on this we define an unbiased importance measure

$$\tilde{\omega_i}(\text{SIU}) = \sum_{t=0}^{T-1} (g_t + \sigma'_t) \cdot \Delta(t).$$

Table 1: **Test accuracies on Permuted-MNIST and Split CIFAR 10/100.** (Mean and std of average accuracy over 3 resp. 10 runs for MNIST resp. CIFAR) Each part compares known algorithms to baselines, explaining algorithms' performances (1-3), or showing speed-ups at equal performance (4).

No.	Algo.	P-MNIST	CIFAR	No.	Algo.	P-MNIST	CIFAR
(1)	MAS	97.3±0.1	73.7±0.5	(3)	SI	97.2±0.1	74.4±0.5
	MAS-MAX	97.4±0.2	73.7±0.2		OnAF	97.3±0.1	74.4±0.6
	AF	97.4±0.1	73.4±0.4		AF	97.4±0.1	73.4±0.4
(2)	SI	97.2±0.1	74.4±0.5	(4)	Fisher (EWC)	97.1±0.3	73.1±0.7
	SI-Bias	97.2±0.1	75.1±0.4		Emp. Fisher (EF)	97.1±0.1	73.0±0.5
	SI-Unbiased	96.3±0.1	72.5±0.8		Batch-EF	97.1±0.1	73.0±0.8

Bias-Only version of SI. To isolate the bias, we can take the difference between biased and unbiased estimate. Concretely, this gives an importance which only measures the bias of SI

$$\tilde{\omega}_i(\text{SIB}) = \sum_{t=0}^{T-1} ((g+\sigma) - (g+\sigma'_t)) \cdot \Delta(t).$$

Observe that this estimate multiplies the parameter-update $\Delta(t)$ with nothing but stochastic gradient noise. From the perspective of the SI path-integral, this should be meaningless and perform poorly.

5.2 Relation of Bias of SI to Absolute Fisher

The bias of SI depends on the optimizer used. The original SI-paper (and we) uses Adam (Kingma & Ba, 2014) and we now analyse the influence of this choice in detail. Recall that $\omega(SI)$ is a sum over terms $\frac{\partial L(t)}{\partial \theta} \cdot \Delta(t)$, where $\Delta(t) = \theta(t+1) - \theta(t)$ is the parameter update at time t. Both terms, $\frac{\partial L(t)}{\partial \theta}$ as well as $\Delta(t)$, are computed using the same mini-batch. Given a stochastic gradient $g_t + \sigma_t$, Adam keeps an exponential average of the gradient $m_t = (1-\beta_1)(g_t+\sigma_t) + \beta_1 m_{t-1}$ as well as the squared gradient $v_t = (1-\beta_2)(g_t+\sigma_t)^2 + \beta_2 v_{t-1}$. Ignoring minor normalisations and the learning rate, the parameter update is $\Delta(t) = m_t/(\sqrt{v_t}+\epsilon)$, with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Thus,

$$\frac{\partial L(t)}{\partial \theta}\Delta(t) = \frac{(1-\beta_1)(g_t+\sigma_t)^2}{\sqrt{v_t}+\epsilon} + \frac{\beta_1(g_t+\sigma_t)m_{t-1}}{\sqrt{v_t}+\epsilon} \approx (1-\beta_1)\frac{(g_t+\sigma_t)^2}{\sqrt{v_t}+\epsilon} \tag{7}$$

Here, we made **Assumption 1** that the gradient noise is larger than the gradient, or more precisely: $(1-\beta_1)\sigma_t^2 \gg \beta_1 m_{t-1} g_t$ (we ignore $\sigma_t m_{t-1}$ since $\mathbb{E}[\sigma_t m_{t-1}] = 0$ and since we average SI over many time steps). Assumption 1 is equivalent to the bias of SI being larger than its unbiased part as detailed in Appendix D.1. Experiments in Section 5.3 provide strong support for this assumption.

Next, consider v_t. It is a slowly moving exponential average of $(g_t+\sigma_t)^2$ and is thus approximately $\mathbb{E}[(g_t+\sigma_t)^2]$. It is reasonable to expect that $\sqrt{v_t}$ is roughly equal to $|g_t+\sigma_t|$. This clearly does not hold for all possible distributions of $g_t+\sigma_t$, but is unlikely to be violated by non adversarially chosen, realistic noise distribution. With equation 7 and **Assumption 2** that $\sqrt{v_t} \approx |g_t+\sigma_t|$, we get

$$\tilde{\omega}(SI) = \sum_t \frac{\partial L(t)}{\partial \theta}\Delta(t) \approx (1-\beta_1)\sum_t \frac{(g_t+\sigma_t)^2}{\sqrt{v_t}} \approx (1-\beta_1)\sum_t |g_t+\sigma_t|. \tag{8}$$

An alternative way to arrive at equation 8 is following the analysis of Balles & Hennig (2018), who show that the update of Adam often depends on the sign of the moment m_t rather than its magnitude.

Summary. Equation 8 shows that SI approximates the *(Empirical) Online Absolute Fisher*, which averages the (empirical) Absolute Fisher 'online' along the training trajectory of the parameters rather than evaluating it only at the final parameter-point. We point out one subtle difference between equation 8 and AF, namely that equation 8 takes absolute values of mini-batch gradients rather than single-image gradients (like AF). In Appendix D.2 we show why this is negligible under our Assumption 1 of large gradient noise. (In fact, we use this insight to significantly speed-up EWC.)

Relation of OnAF to AF. To see if SI, like MAS, is related to the Absolute Fisher (AF), it remains to explore the relation of OnAF and AF. If the latter two are similar, it is clear from the above that SI is similar to AF. Theoretically, in the limit of small learning rates or small Hessian (of the loss), OnAF and AF become equal. Moreover, we point out that the fact that Adam's second moment estimate usually works well with a very slowly moving average ($\beta_2 = 0.999$) indicates that the squared gradient changes slowly in many settings, suggesting that the Absolute Fisher also changes slowly, so that OnAF and AF are similar. Therefore, we hypothesise that OnAF and AF are similar and empirically confirm this claim in Section 5.3.

Influence of Regularisation The derivation above assumes that the parameter update $\Delta(t)$ is given by the gradients of the current task, ignoring the auxiliary regularisation loss. The latter will change update direction and momentum of the optimizer. Thus, strong regularisation will make the relation between SI and OnAF noisy, but not abolish it. This will be confirmed empirically in Section 5.3.

5.3 Emprical Investigation of SI, its bias and Absolute Fisher

Bias dominates SI & checking Assumption 1. According to the motivation of SI the sum of importances (over parameters) $\sum_i \tilde{\omega}_i(\text{SI})$ should track the decrease in loss $L(0) - L(T)$, see equation 2.

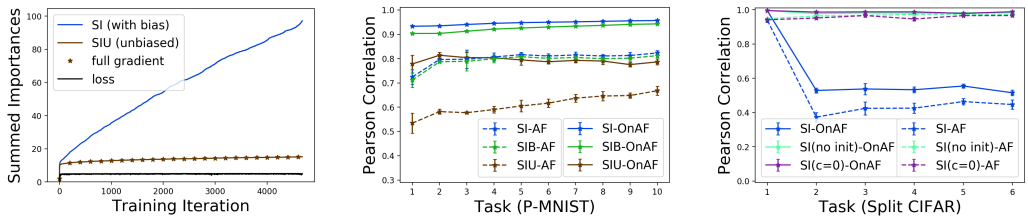

Figure 2: **Effect of Bias and Regularisation on SI. Left:** Summed Importances for SI and its unbiased version, showing that the bias dominates SI and that Assumption 1 holds. **Mid:** Pearson Correlations of SI, its bias (SIB), and unbiased version (SIU) on MNIST, showing that relation between SI and (On)AF is due to bias. **Right:** Relation between standard-SI (blue) and OnAF and two SI-controls: 'no init' (turquoise) does not re-initialise network weights after each task; '$c = 0$' (purple) has regularisation strength 0. This shows that strong regularisation weakens the tie between SI and (On)AF as explained by our theory.

Therefore, we investigated the summed importances for SI, its unbiased version SIU and for an approximation of the path integral based on the full training set gradient (rather than approximating it). The results, Figure 2 (left) show: (1) the bias is 5-6 times larger than the unbiased part; (2) using an unbiased gradient estimate and the full gradient gives almost identical sums, supporting the validity of the unbiased estimator. Note that even the unbiased first order approximation of the path integral overestimates the decrease in loss. This is consistent with previous studies showing that the loss has positive curvature (Jastrzebski et al., 2018; Lan et al., 2019). We emphasise that the bias (difference between SI and SIU) in Figure 2 (left) is due to the term $(1 - \beta_1)\sigma_t^2$ and that this bias is considerably larger than the unbiased part. This is direct, strong evidence of Assumption 1 (see Appendix D.1 for full calculation). Additionally, a detailed analysis of the gradient noise in Appendix E gives further support for Assumption 1. Results on CIFAR are analogous (Appendix Figure G.6).

Bias explains SI's performance. We saw that the bias is much larger than the unbiased part. But how does it influence SI's performance? To quantify this, we compared SI to its unbiased version SIU and the bias-only version SIB. Note that SIB is completely independent of the path integral motivating SI, only measures gradient noise and therefore should perform poorly according to SI's original motivation. However, the results in Table 1(2), reveal the opposite: Removing the bias reduces performance of SI (SIU is worse), whereas isolating the bias does not affect or slightly improve performance (difference between SIB and SI on CIFAR is statistically significant, $p < 0.005$ two-sided t-test). This demonstrates that SI relies on its bias for its continual learning performance.

Bias and Fisher & Checking Assumption 2. As additional experiment, we checked how much the relation of SI to the Fisher depends on the bias. Our theory predicts that the relation between SI and OnAF is mostly based on the bias. Figure 2 (mid) confirms this, showing a strong correlation between SI, SIB and OnAF and a weaker correlation between SIU and OnAF. Note that the correlation of OnAF and SI clearly supports Assumption 2; results on CIFAR are similar (c.f. Figure G.1), but with a stronger influence of regularisation as seen in Figure 2 (right) and full text below.

Relation of SI, OnAF to AF. Next, we investigated the relation between OnAF and AF, finding strong correlations: between 0.73-0.87 on MNIST and 0.86-0.95 on CIFAR, see Figure G.3 in the appendix. Figure 2 (mid panel for MNIST, right panel for CIFAR, blue) shows that this carries over to relating SI and AF. An additional control in Appendix G.2.1 shows that this observation is not an artefact of long training times. These results confirm that SI is closely related to AF.

Effect of Regularisation on SI. Lastly, we assessed our derivation that large regularisation will diminish the correlation between SI and OnAF. To this end, we compared standard SI to two variants of it which have less strong regularisation. The first control simply sets regularisation strength to $c = 0$. The second control refrains from re-initialising the network weights after each task (exactly as in the original SI paper, albeit with slightly worse validation performance than our version). In the second setting the current parameters θ never move too far from their old value $\theta^{(k-1)}$, implying smaller gradients from the quadratic regularisation loss, and also meaning that a smaller value of $c = 0.5$ is optimal. We see that for both controls with weak regularisation the relation to OnAF is larger than for standard-SI with strong regularisation (Figure 2 (right)), verifying our theory.

Efficient version of EWC. Our analysis of SI shows that the Fisher can be approximated by the Empirical Fisher and that the gradient computation for the Empirical Fisher can be averaged across mini-batches (before it is squared) rather than carried out for each image individually (see Appendix D.2 for theory). This is important as it offers large computational savings. First, we checked that the approximations described above do not harm performance. To this end we compared EWC (using the 'real' Fisher) to the Empirical Fisher and a variant calculating the Empirical Fisher in mini-batches (Batch-EF), c.f. C.1. The results (Table 1 (4)) show that the approximations have the same performance as EWC. At the same time, **our final algorithm Batch-EF is more than 800x faster than EWC at approximating the Fisher** (Figure 3); on Split CIFAR the speed up is >300x. Note also that EWC is dominated by the time it needs to approximate the Fisher, while for our sped-up Batch-EF this time has become negligible compared to the training time.

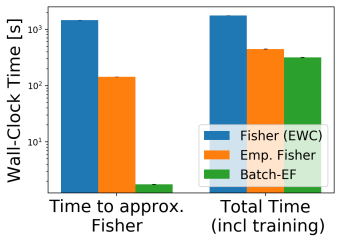

Figure 3: Wall-Clock Time (log sale) for EWC and new versions measured on P-MNIST. Batch-EF (ours) approximates the Fisher more than 800x faster. Raw data in Appendix, Table G.1.

6 Experimental Setup

Above, we carried out experimental validations of our assumptions. Here, we outline our setup, see Appendix C for full details. We closely follow SI's setting (Zenke et al., 2017): In domain-incremental **Permuted MNIST** (Goodfellow et al., 2013) each of 10 tasks consists of a random (but fixed) pixel-permutation of MNIST and a **fully connected ReLU network** is used. The task-incremental **Split CIFAR 10/100** benchmark (Zenke et al., 2017) consists of six 10-way classification tasks, the first is CIFAR 10, and the other ones are extracted from CIFAR 100. The **keras default** CIFAR 10 **convolutional architecture** (with dropout and max-pooling) is used keras. The only difference to the setup of Zenke et al. (2017) is that like Swaroop et al. (2019), we usually re-initialise the network weights after each task, observing performance improvements. Code is available on github. For brevity, we sometimes only showed plots based on one benchmark. The analogous plot based on the other benchmark can then be found in the appendix.

7 Discussion

We have investigated regularisation approaches for continual learning, which are the method of choice for continual learning without replaying old data or expanding the model. We have provided strong theoretical and experimental evidence that the three most influential methods in this domain are surprisingly similar. This does not only unify these three methods, but also their many follow ups. Additionally, relating SI and MAS to the Fisher Information has the benefit of putting these algorithms on a solid theoretical underpinning. Kirkpatrick et al. (2017); Huszár (2018) give a theoretical, bayesian explanation of why the Fisher Information with a quadratic weight penalty is a principled approach to continual learning and recently, Yin et al. (2020) provided theoretical guarantees as well as a frequentist interpretation of this approach. Our results extend these insights from EWC to all MAS- and SI-based algorithms providing them with a solid theoretical basis.

Our thorough analysis and carefully designed baselines yielded several insights. Among other things, they showed that: our baseline OnAF is a theoretically more sound version of SI; our (predictive) AF algorithm is more sound than MAS; EWC can be replaced by our Batch-EF offering large computational speed-ups. Moreover, we point out that our algorithm SIU to approximate SI's path integral opens up the possibility to apply the (non continual learning related) algorithm LCA (Lan et al., 2019) to larger datasets and architectures, since it drastically reduces its computational costs.

Finally, we hope that more future work will focus on unifying and explaining the ever growing body of continual learning literature, advancing our understanding of these methods and allowing to discover new, efficient algorithms.

REFERENCES

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 139–154, 2018.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11254–11263, 2019.

Lukas Balles and Philipp Hennig. Dissecting adam: The sign, magnitude and variance of stochastic gradients. In *International Conference on Machine Learning*, pp. 404–413, 2018.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 532–547, 2018.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. Continual learning with tiny episodic memories. *arXiv preprint arXiv:1902.10486*, 2019.

Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

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

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Siavash Golkar, Michael Kagan, and Kyunghyun Cho. Continual learning via neural pruning. *arXiv preprint arXiv:1903.04476*, 2019.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

Yen-Chang Hsu, Yen-Cheng Liu, Anita Ramasamy, and Zsolt Kira. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018.

Ferenc Huszár. Note on the quadratic penalties in elastic weight consolidation. *Proceedings of the National Academy of Sciences*, pp. 201717042, 2018.

Stanislaw Jastrzebski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. On the relation between the sharpest directions of dnn loss and the sgd step length. *arXiv preprint arXiv:1807.05031*, 2018.

Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. *arXiv preprint arXiv:1711.10563*, 2017.

keras. Official keras cnn example. `https://github.com/keras-team/keras/blob/master/examples/cifar10_cnn.py`. Accessed: 2020-09-25.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114 (13):3521–3526, 2017.

Frederik Kunstner, Lukas Balles, and Philipp Hennig. Limitations of the empirical fisher approximation. *arXiv preprint arXiv:1905.12558*, 2019.

Janice Lan, Rosanne Liu, Hattie Zhou, and Jason Yosinski. Lca: Loss change allocation for neural network training. In *Advances in Neural Information Processing Systems*, pp. 3614–3624, 2019.

Janghyeon Lee, Hyeong Gwon Hong, Donggyu Joo, and Junmo Kim. Continual learning with extended kronecker-factored approximate curvature. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9001–9010, 2020.

Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. *arXiv preprint arXiv:1904.00310*, 2019.

Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 2262–2268. IEEE, 2018.

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

James Martens. New insights and perspectives on the natural gradient method. *arXiv preprint arXiv:1412.1193*, 2014.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation - Advances in Research and Theory*, 24(C):109–165, January 1989. ISSN 0079-7421. doi: 10.1016/S0079-7421(08)60536-8.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning, 2020.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.

Dongmin Park, Seokil Hong, Bohyung Han, and Kyoung Mu Lee. Continual learning by asymmetric loss approximation with single-side overestimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3335–3344, 2019.

Razvan Pascanu and Yoshua Bengio. Revisiting natural gradient for deep networks. *arXiv preprint arXiv:1301.3584*, 2013.

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

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, pp. 3738–3748, 2018.

Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*, 2018.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, pp. 2990–2999, 2017.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Rupesh K Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to compute. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26*, pp. 2310–2318. Curran Associates, Inc., 2013. URL http://papers.nips.cc/paper/5059-compete-to-compute.pdf.

Siddharth Swaroop, Cuong V Nguyen, Thang D Bui, and Richard E Turner. Improving and understanding variational continual learning. *arXiv preprint arXiv:1905.02099*, 2019.

Gido M van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

Johannes von Oswald, Christian Henning, João Sacramento, and Benjamin F Grewe. Continual learning with hypernetworks. *arXiv preprint arXiv:1906.00695*, 2019.

Dong Yin, Mehrdad Farajtabar, and Ang Li. Sola: Continual learning with second-order loss approximation, 2020.

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

A APPENDIX

This appendix has several components:

1. We provide a tabular overview over algorithms and baselines described in the paper in B
2. We give full experimental details including hyperparameter seraches and values, as well as full details about the variants and speed-ups of EWC in Appendix C.
3. We give details for two calculations omitted in the main paper in Appendix D.
4. We empirically investigate the gradient noise in Appendix E. This experiment is not specific to continual learning.
5. We critically review experimental claims from previous work about regularisation methods in Appendix F.
6. We include additional experiments and plots in Appendix G.

B OVERVIEW OVER ALGORITHMS AND BASELINES

Table B.1: **Summary of Regularisation Methods and Related Baselines.** *Details:* Algorithms on the left calculate importance 'online' along the parameter trajectory during training. Algorithms on the right calculate importance at the end of training a task by going through (part of) the training set again. Thus, the sum is over timesteps t (left) or datapoints X (right). N is the number of images over which is summed. All the algorithms on the left rescale their final importances as in equation equation 4 for fair comparison. $\Delta(t) = \theta(t+1) - \theta(t)$ refers to the parameter update at time t, which depends on both the current task's loss and the auxiliary regularisation loss. Moreover, $(g_t + \sigma_t)$ refers to the stochastic gradient estimate of the current task's loss (where g_t is the full gradient and σ_t the noise) given to the optimizer to update parameters. In contrast, $(g_t + \sigma'_t)$ refers to an independent stochastic gradient estimate. For a datapoint X, $\mathbf{q}_X$ denotes the predicted label distribution and $g(X, y)$ refers to the gradient of the negative log-likelihood of (X, y).

Name	**Paramater importance** $\omega(\cdot)$	**Name**	**Paramater importance** $\omega(\cdot)$
SI	$\sum_t (g_t + \sigma_t)\Delta(t)$	Fisher (EWC)	$\frac{1}{N}\sum_X \mathbb{E}_{y\sim\mathbf{q}_X}\left[g(X,y)^2\right]$
SIU (SI-Unbiased)	$\sum_t (g_t + \sigma'_t)\Delta(t)$	AF (Absolute Fisher)	$\frac{1}{N}\sum_X \mathbb{E}_{y\sim\mathbf{q}_X}\left[\lvert g(X,y)\rvert\right]$
SIB (SI Bias-only)	$\sum_t (\sigma_t - \sigma'_t)\Delta(t)$	MAS	$\frac{1}{N}\sum_X \left\lvert\frac{\partial \lvert\lvert \mathbf{q}_X \rvert\rvert^2}{\partial\theta}\right\rvert$
OnAF (Online AF)	$\sum_t \lvert g_t + \sigma_t \rvert$	MASX (MAS-Max)	$\frac{1}{N}\sum_X \left\lvert\frac{\partial (\max \mathbf{q}_X)^2}{\partial\theta}\right\rvert$

C EXPERIMENTAL DETAILS

C.1 DETAILS, VARIANTS AND SPEED-UPS OF EWC. DETAILS OF MAS AND AF.

For EWC we calculate the 'real' Fisher Information as defined in the main article.[3] For P-MNIST, we randomly sample 2000 training images (rather than iterating through the entire training set, which is prohibitively expensive). For Split CIFAR we use 500 random training images. Preliminary experiments with more than 2000 (500) images gave no performance improvement.

Similarly, the Empirical Fisher is based on 2000 (500) samples.

[3]We use a 'naïve' implementation to calculate the Fisher, which simply iterates over all potential labels. Theoretically, this implementation could be improved, but in practice this is infeasible since the necessary computations are currently not well-supported by standard deep-learning libraries; see Kunstner et al. (2019), Appendix B for details.

For Batch-EF, we also use 2000 (resp 500) samples and split them into 20 (resp 5) mini-batches of size 100. We compute the average gradient for each mini-batch, square it and average the squares over the mini-batches. We also multiply the result by the mini-batch size (100) for normalisation. For a theoretical justification see D.2. Note that this requires 100 times less forward passes than calculating the Empirical Fisher (albeit with larger batch-size) and makes more efficient use of the deep-learning libraries optimised for parallel-batch computation.

MAS, MAS-X and AF algorithms are based on 1000 resp 500 random samples. When comparing these algorithms, we use the same set of samples for each algorithm.

When calculating the corelation between SI and AF, we use 30000 (MNIST) resp 500 (CIFAR) samples to evaluate AF. This is because we observed a higher variance on MNIST for AF (but as pointed out above taking more samples did not improve performance of AF).

C.2 Benchmarks

In **Permuted MNIST** (Goodfellow et al., 2013) each task consists of predicting the label of a random (but fixed) pixel-permutation of MNIST. We use a total of 10 tasks. As noted, we use a domain incremental setting, i.e. a single output head shared by all tasks.

In **Split CIFAR10/100** the network first has to classify CIFAR10 and then is successively given groups of 10 (consecutive) classes of CIFAR100. As in (Zenke et al., 2017), we use 6 tasks in total. Preliminary experiments on the maximum of 11 tasks showed very little difference. We use a task-incremental setting, i.e. each task has its output head and task-identity is known during training and testing.

C.3 Pre-processing

Following Zenke et al. (2017), we normalise all pixel values to be in the interval $[0, 1]$ (i.e. we divide by 255) for MNIST and CIFAR datasets and use no data augmentation. We point out that pre-processing can affect performance, e.g. differences in pre-processing are part of the reason why results for SI on Permuted-MNIST in (Hsu et al., 2018) are considerably worse than reported in (Zenke et al., 2017) (the other two reasons being an unusually small learning rate of 10^{-4} for Adam in (Hsu et al., 2018) and a slightly smaller architecture).

C.4 Architectures and Initialization

For P-MNIST we use a fully connected net with ReLu activations and two hidden layers of 2000 units each. For Split CIFAR10/100 we use the default Keras CNN for CIFAR 10 with 4 convolutional layers and two fully connected layers, dropout (Srivastava et al., 2014) and maxpooling, see Table C.1. We use Glorot-uniform (Glorot & Bengio, 2010) initialization for both architectures.

C.5 Optimization

We use the Adam Optimizer (Kingma & Ba, 2014) with tensorflow (Abadi et al., 2016) default settings ($lr = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$) and batchsize 256 for both tasks like Zenke et al. (2017). On Permuted-MNIST we train each task for 20 epochs, on Split CIFAR we train each task for 60 epochs, also following Zenke et al. (2017). We reset the optimizer variables (momentum and such) after each task.

C.6 SI & OnAF details

Recall that we applied the operation $\max(0, \cdot)$ (i.e. a ReLU activation) to the importance measure of each individual task (equation 4 of main paper), before adding it to the overall importance. In the original SI implementation, this seems to be replaced by applying the same operation to the overall importances (after adding potentially negative values from a given task). No description of either of these operations is given in the SI-paper. In light of our findings, our version seems more justified.

Somewhat naturally, the gradient $\frac{\partial L}{\partial \theta}$ usually refers to the cross-entropy loss of the current task and not the total loss including regularisation. For CIFAR we evaluated this gradient without dropout (but the parameter update was of course evaluated with dropout).[4]

For OnAF, similarly to SI, we used the gradient evaluated without dropout on the cross-entropy loss of the current task for our importance measure.

When evaluated on benchmarks, SI, SIU, SIB and OnAF are all rescaled according to equation 4 to make the comparison as fair as possible. Scatter plots and correlations use values before this rescaling.

Table C.1: CIFAR 10/100 architecture. Following Zenke et al. (2017) we use the keras default architecture for CIFAR 10. Below, 'Filt.' refers to the number of filters of a convolutional layer, or respectively the number of neurons in a fully connected layer. 'Drop.' refers to the dropout rate. 'Non-Lin.' refers to the type of non-linearity used.
Table reproduced and slightly adapted from (Zenke et al., 2017).

Layer	Kernel	Stride	Filt.	Drop.	Non-Lin.
3x32x32 input					
Convolution	3x3	1x1	32		ReLU
Convolution	3x3	1x1	32		ReLU
MaxPool	2x2	2x2		0.25	
Convolution	3x3	1x1	64		ReLU
Convolution	3x3	1x1	64		ReLU
MaxPool	2x2	2x2		0.25	
FC			512	0.5	ReLU
Task 1: FC			10		softmax
...:FC			10		softmax
Task 6: FC			10		softmax

C.7 HYPERPARAMETERS

For all methods and benchmarks, we performed grid searches for the hyperparameter c and over the choice whether or not to re-initialise model variables after each task.

The grid of c included values $a \cdot 10^i$ where $a \in \{1, 2, 5\}$ and i was chosen in a suitable range (if a hyperparameter close to a boundary of our grid performed best, we extended the grid).

For CIFAR, we measured validation set performance based on at least three repetitions for good hyperparameters. We then picked the best HP and ran 10 repetitions on the test set. For MNIST, we measured HP-quality on the test set based on at least 3 runs for good HPs.

Additionally, SI and consequently SIU, SIB as well as OnAF have rescaled importance (c.f. equation equation 4 from main paper). The damping term ξ in this rescaling was set to 0.1 for MNIST and to 0.001 for CIFAR following Zenke et al. (2017) without further HP search.

All results shown are based on the same hyperparameters obtained – individually for each method – as described above. They can be found in Table C.2. We note that the difference between the HPs for MAS and MASX might seem to contradict our claims that the two measures are almost identical (they should require the same c in this case), but this is most likely due to similar performance for different HPs and random fluctuations. For example on CIFAR, MAS had almost identical validation performance for $c = 200$ (best for MAS) and $c = 1000$ (best for MASX) (74.2 ± 0.7 vs 74.1 ± 0.5).

Also for the other methods, we observed that usually there were two or more HP-configurations which performed very similarly. The precise 'best' values as found by the HP search and reported in Table C.2 are therefore subject to random fluctuations in the grid search.

[4]We did not check the original SI code for what's done there and the paper does not describe which version is used.

Table C.2: Hyperparameter values for our experiments. See also maintext.

Algorithm	MNIST		CIFAR	
	c	re-init	c	re-init
SI	0.2	√	5.0	√
SIU	2.0	√	2.0	×
SIB	0.5	√	5.0	√
OnAF	5e-5	√	5e-4	√
MAS	500	×	200	√
MASX	200	×	1e3	√
EWC	1e3	×	5e4	√
AF	200	×	1e3	√
Emp. Fisher	1e6	×	2e7	√
Batch-EF	5e5	×	2e7	√

C.8 Scatter Plot Data Collection and Intensity Plots

The scatter plots are based on a run of SI, where we in parallel to evaluating the SI importance measure also evaluated all other importance measures. Figures 1 and 2 show data based on 4 repetitions of this experiment and confirm that the correlations observed in the scatter plots are representative, it also shows the effect of sample sizes on AF.

Intensity Plots (as used for some figures in the appendix) Show same kind of data as scatter plots, but each weight is binned into one of 80x50 equispaced bins. The number of weights per bin is divided by the maximum number of weights per bin in that column and the resulting value is shown on a gray scale. Normalisation per column was performed since for 'global normalisation' (divide by total number of weights) only weights in the bottom left are visible (as most weights have small importances); the number of bins was chosen to match aspect ratio of plots.

C.9 Computing Infrastructure

We ran experiments on one NVIDIA GeForce GTX 980 and on one NVIDIA GeForce GTX 1080 Ti GPU. All wall-clock time experiments were run on the latter GPU. We used tensorflow 1.4 for all experiments.

D Detailed Calculations for the bias of SI, and for calculating Fisher in Batches

Here we give the details for two calculations omitted in the main text.

D.1 Formula of Bias of SI with Adam

Here, we show that the bias, i.e. the expected difference between SI and SIU, is equal to $(1-\beta_1)\sigma_t^2/(\sqrt{v_t}+\epsilon)$ and that Assumption 1 is equivalent to the bias being larger than the unbiased part of SI.

Recall that for SI, we approximate $\frac{\partial L(t)}{\partial \theta}$ by $g_t + \sigma_t$, which is the same gradient estimate given to Adam. So we get

$$\textbf{SI:} \qquad \frac{\partial L(t)}{\partial \theta}\Delta(t) = \frac{(1-\beta_1)(g_t+\sigma_t)^2}{\sqrt{v_t}+\epsilon} + \frac{\beta_1(g_t+\sigma_t)m_{t-1}}{\sqrt{v_t}+\epsilon}.$$

For SIU, we use an independent mini-batch estimate $g_t + \sigma_t'$ for $\frac{\partial L(t)}{\partial \theta}$ and therefore obtain

$$\textbf{SIU:} \qquad \frac{\partial L(t)}{\partial \theta}\Delta(t) = \frac{(1-\beta_1)(g_t+\sigma_t')(g_t+\sigma_t)}{\sqrt{v_t}+\epsilon} + \frac{\beta_1(g_t+\sigma_t')m_{t-1}}{\sqrt{v_t}+\epsilon}.$$

Taking the difference between these two and ignoring all terms which have expectation zero (note that $\mathbb{E}[\sigma_t] = \mathbb{E}[\sigma'_t] = 0$ and that σ_t, σ'_t are independent of m_{t-1} and g_t) gives

$$\textbf{SI} - \textbf{SIU:} \qquad (1-\beta_1)\frac{\sigma_t^2}{\sqrt{v_t}+\epsilon}$$

as claimed.

To see that Assumption 1 is equivalent to the bias being larger than the unbiased part, note that in expectation SIU equals

$$(1-\beta_1)\frac{g_t^2}{\sqrt{v_t}+\epsilon} + \beta_1\frac{g_t m_{t-1}}{\sqrt{v_t}+\epsilon} \approx \beta_1\frac{m_{t-1}g_t}{\sqrt{v_t}+\epsilon}$$

The last approximation here is valid because $\beta_1 m_{t-1} \gg (1-\beta_1)g_t$ which holds since (1) $\beta_1 \gg (1-\beta_1)$ and (2) $\mathbb{E}[|m_{t-1}|] \geq \mathbb{E}[|g_t|]$. (1) follows from $\beta_1 = 0.9$, and (2) holds since $\mathbb{E}[m_{t-1}] \approx g_t$ and since m_{t-1} has a noise component (due to gradient noise), while g_t is a deterministic quantity, implying $\mathbb{E}[|m_{t-1}|] \geq \mathbb{E}[|g_t|]$ by elementary considerations.

In particular, the two calculations for the bias and the expectation of SIU above imply that Assumption 1, which is $(1-\beta_1)\sigma_t^2 \gg \beta_1 m_{t-1} g_t$, is equivalent to the bias of SI being larger than its unbiased part.

D.2 Effect of Calculating Fisher in Mini-Batches: Analysis of Batch-EF

For this subsection, let us slightly change notation and denote the images by $X_1, \ldots, X_D$ and the gradients (with respect to their labels and the cross entropy loss) by $g+\sigma_1, \ldots g+\sigma_D$. Here, again g is the overall training set gradient and σ_i is the noise (i.e. $\sum_{i=1}^D \sigma_i = 0$). Then the Empirical Fisher is given by

$$\text{EF} = \frac{1}{D}\sum_{i=1}^{D}(g+\sigma_i)^2$$

We want to compare this to evaluating the squared gradient over a batch. Let $i_1, \ldots, i_b$ denote uniformly random, independent indices from $\{1, \ldots, D\}$, so that $X_{i_1}, \ldots, X_{i_b}$ is a random mini-batch of size b. Let $g+\sigma$ be the gradient on this mini-batch. We then have, taking expectations over the random indices,

$$\begin{aligned}
\mathbb{E}[(g+\sigma)^2] &= \mathbb{E}\left[\frac{1}{b^2}\sum_{r,s=1}^{b}(g+\sigma_{i_r})(g+\sigma_{i_s})\right] \\
&= \frac{b(b-1)}{b^2}\mathbb{E}[(g+\sigma_{i_1})(g+\sigma_{i_2})] + \frac{b}{b^2}\mathbb{E}[(g+\sigma_{i_1})^2] \\
&= \frac{b-1}{b}g^2 + \frac{1}{b}\text{EF} \\
&\approx \frac{1}{b}\text{EF}
\end{aligned}$$

This conclusion may seem surprising, but recall that it is only based on one assumption, namely that the gradient noise is considerably bigger than the gradient itself - an assumption for which we have collected ample evidence.

Concretely, we assume $\mathbb{E}[(g+\sigma_i)^2] \gg (b-1)g^2$. Note that unlike in other sections of this manuscript, here σ_i refers to the gradient noise of individual images (rather than mini-batch noise). We have seen that even with a batch-size of 256 (i.e. when we reduce then noise by a factor of 256) the noise is still much bigger than the gradient (recall Figures G.5, G.6, E.1, E.2), so that our assumption is true with room to spare.

E Gradient Noise

Here, we quantitatively assess the noise magnitude outside the continual learning context. Recall that Figure 2 (left) from the main paper, as well as Figures G.5 and G.6 already show that the noise

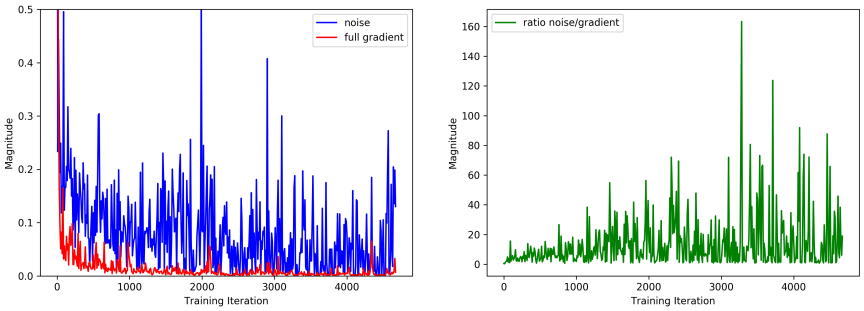

Figure E.1: Gradient noise, measured as squared ℓ_2 distance between full training set gradient and the stochastic mini-batch gradient with a batch size of 256. 'Full gradient' magnitude is also measured as squared ℓ_2 norm.
Data obtained by training a ReLu network with 2 hidden layers of 2000 hidden units for 20 epochs with default Adam settings. Only every 20-th datapoint shown for better visualisation.

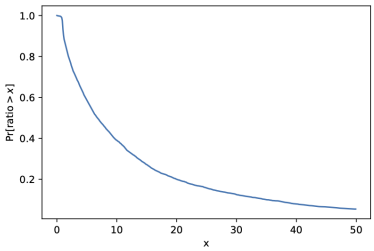

Figure E.2: y-value shows fraction of training iterations in which the ratio between mini-batch noise and full training set gradient was at least x-value. Data obtained as described in main text / Figure E.1, in particular the batch-size was 256. 'Ratio' refers to the ratio of squared ℓ_2 norms of the respective values.

dominates the SI importance measure, which indicates that the noise is considerably larger than the gradient itself.

To obtain an assessment independent of the SI continual learning importance measure, we trained our network on MNIST as described before, i.e. a ReLu network with 2 hidden layers of 2000 units each, trained for 20 epochs with batch size 256 and default Adam settings. At each training iteration, on top of calculating the stochastic mini-batch gradient used for optimization, we also computed the full gradient on the entire training set and computed the noise – which refers to the squared ℓ_2 distance between the stochastic mini-batch gradient and the full gradient – as well as the ratio between noise and gradient, measured as the ratio of squared ℓ_2 norms. The results are shown in Figure E.1. In addition, we computed the fraction of iterations in which the ratio between noise and squared gradient norm is above a certain threshold, see Figure E.2.

F Relative Performance of Regularisation Methods

Several papers (van de Ven & Tolias, 2019; Hsu et al., 2018; Farquhar & Gal, 2018) played an important role in recognising that different continual learning algorithms had been evaluated in settings with varying difficulties. These papers also greatly clarified these setting, making an important contribution for future work.

Here, we want to point out that some of the results (and thus conclusions) of the experiments in (Hsu et al., 2018) (these were the only ones we investigated more closely) are specific to suboptimal settings and hyperparameters used there.

For example, it is found there that EWC, MAS, SI perform worse than an importance measure which assigns the same importance to all parameters (called 'L2 regularisation'). This is in contrast to results reported in the EWC paper (Kirkpatrick et al., 2017), who found the L2 baseline to perform worse than EWC. We also attempted to reproduce the L2 results and tried two versions of this approach, one with constant importance for all tasks and one with importance increasing linearly with the number of tasks. The latter approach worked better, but still considerably underperformed EWC, MAS, SI on Permuted-MNIST (88% vs 97%). We didn't run these experiments on CIFAR. An alternative importance measure based on the weights' magnitudes after training was better, but still could not compete with SI, EWC, MAS (<95% vs >97%) showing that these importance measure are more use- and meaningful than naive baselines.

Additionally, in other work we found that for domain incremental settings (again, these were the only ones we investigated), results of baselines EWC and SI, but also of fine-tune (a baseline which takes no measures to prevent catastrophic forgetting) can be improved, often by more than 10% compared to previously mentioned reports (Hsu et al., 2018; van de Ven & Tolias, 2019). This also implies that the difference between well set-up regularisation approaches and for example repaly-methods is not nearly as big as previously thought. The differences between these results are explained by different factors (we explicitly tested eeach factor): hyperparameters (for example using Adam with the standard learning rate 0.001 improves performance over the learning rate 0.0001 used previously), initialisation (Glorot Uniform Initialisation works considerably better for SI than pytorch standard initialisation for linear layers, which reveals unexpected defaults upon close investigation), training duration (regularisation approaches usually notably benefit from longer training, this is reported in Swaroop et al. (2019), and we also found that performance on P-MNIST can be improved to 98% by training for 200 rather than 20 epochs; Hsu et al. (2018) use very short training times), data-preprocessing (depending on the setting different normalisations have different, non-negligible impacts).

G More Experiments and Plots

G.1 Plots, Data analogous to Figures/Tables in main paper

Here, we show data for experiments already partly reported in the main paper, including both benchmarks and each task of each benchmark.

We start with the magnitude of the bias of SI (Figure 2, left panel): The observation that most of the SI importance is due to its bias is consistent across datasets and tasks, see Figures G.5 and G.6. Intriguingly, on CIFAR we find that the unbiased approximation of SI slightly underestimates the decrease in loss in the last task, suggesting that strong regularisation pushes the parameters in places, where the cross-entropy of the current task has negative curvature.

We show correlations between SI, SIB, SIU and AF, OnAF on CIFAR (analogous to Figure 2, middle) in Figure G.1. The findings are in accordance with the ones on MNIST reported in the main paper. Note that only the first task of CIFAR (i.e. CIFAR10) shows slightly different behaviour, as SIU and OnAF are more similar than on other tasks: correlation with OnAF is 0.994 ± 0.001 for SI; 0.991 ± 0.001 for SIU; 0.979 ± 0.003 for SIB. The similarity between SIU and OnAF in this situation is explained by the weights with largest OnAF importance: If we remove the 5% of weights which have highest OnAF importance, correlation between OnAF, SIU drops to 0.599 ± 0.027, but for SI resp. SIB the same procedure yields 0.977 ± 0.002 resp. 0.965 ± 0.003. This indicates that for the first task of CIFAR there is a small fraction of weights with large OnAF importances, which are dominated by the gradient rather than the noise. The remainder of weights is in accordance with our previous observations and dominated by noise, recall also Figure G.6.

We did not conduct the analogue of experiment corresponding to Figure 2, right panel, on MNIST since there the influence of regularisation was already weak there.

We also compare wall-clock time between EWC and our proposed speed-ups, see Figure G.2 for an analogue of Figure 3 on CIFAR rather than MNIST and see Table G.1 for raw data.
Additional Details: 'Time to approx. Fisher' refers to the accumulated time used to calculate the Fisher on each task (10 resp 6 tasks on MNIST/CIFAR). 'Total Time' refers to the accumulated time the algorithm needs to both train on each task and evaluate the Fisher on each task. It excludes

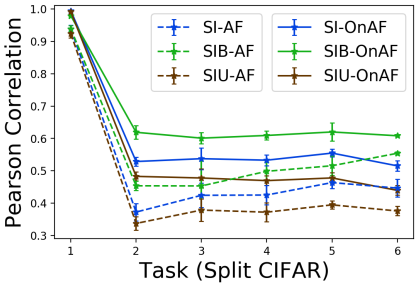

Figure G.1: **Effect of Bias on SI.** Analogous to Figure 2 (middle) but on CIFAR rather than MNIST. Drop of correlations for tasks 2-6 is due to strong correlation as predicted in Section 5.2 and confirmed in Figure 2 (right).

Table G.1: **Wall-Clock Times of EWC and suggested variants.** We report mean and std over 3 repetitions of wall-clock times measured in seconds.

Algorithm	P-MNIST		Split CIFAR	
	Fisher Time	Total Time	Fisher Time	Total Time
EWC	1459.0 ± 1.7	1763.6 ± 2.8	174.0 ± 1.0	461.9 ± 2.5
Emp. FIsher	142.9 ± 0.4	447.1 ± 1.4	17.8 ± 0.1	310.6 ± 7.7
Batch-EF	1.8 ± 0.02	318.0 ± 4.9	0.5 ± 0.04	291.4 ± 3.6

model-evaluation (which is part of the code provided for the other experiments). Note that for EWC on CIFAR (probably due to some peculiarities of tensorflow and our specific implementation), the forward pass to calculate the predicted label distribution for the Fisher was more than an order of magnitude slower than comparable forward passes. Since this model-evaluation is not part of Emp. Fisher and Batch-EF (and since we are confident that more tf-compatible implementations exists), we excluded it from the time-measurements of EWC. Observe that this modification only makes EWC faster, and without it the speed-up of our algorithms compared to EWC would be even more drastic.

G.2 Relation of OnAF and AF

We show summaries of the correlations between OnAF and AF in Figure G.3. This includes comparisons of AF based on different sample sizes. We also investigated the influence of training duration on AF, OnAF, see next section.

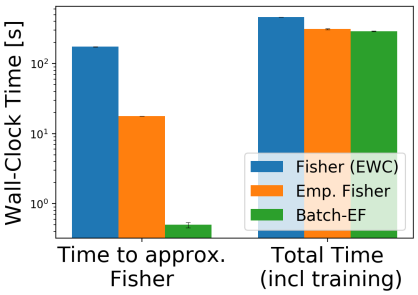

Figure G.2: Wall-Clock Time (log sale) for EWC and new versions measured on CIFAR. Batch-EF (ours) approximates the Fisher more than 300x faster.

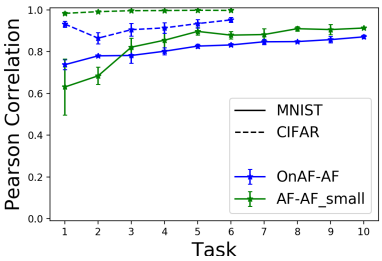

Figure G.3: **Correlations between AF and OnAF.** Solid lines show correlations on MNIST, dotted lines show correlations on CIFAR. 'AF' is based on 30000 (MNIST) resp. 5000 (CIFAR) samples, while 'AF_small' is based on 1000 resp. 500 samples.

G.2.1 Influence of Training Duration on AF, OnAF

It seems plausible that the correlation between AF and OnAF we observed is due to training on tasks for a long time – the model might converge quickly so that the sum of OnAF is dominated by summands which approximate AF close to the final point in training. To test this, we trained our networks for varying numbers of epochs on the first task of P-MNIST and Split CIFAR10/100 and measured the correlation between OnAF and AF. The results in Tables G.2, G.3 show that the correlation between AF, OnAF does not rely on long training duration. On MNIST, we even see decreasing correlation with longer training time, this may be due to either an actual decrease in correlation or due to higher variance when estimating AF using 10000 samples (or both).

Table G.2: **Correlation between AF and OnAF on CIFAR10 depending on training time.** We show mean and standard deviation of Pearson correlation over 3 runs. AF is based on 500 samples. If the standard deviation is below 0.005 it is shown as 0.00.

Number of Epochs	Correlation
1	0.94± 0.01
2	0.95± 0.00
4	0.96± 0.01
8	0.96± 0.01
15	0.97± 0.00
30	0.97± 0.00
60	0.95± 0.01

Table G.3: **Correlation between AF and OnAF on MNIST depending on training time.** We show mean and standard deviation of Pearson correlation over 3 runs. AF is based on 10000 samples. Recall that for 30000 samples and 20 epochs correlation is larger.

Number of Epochs	Correlation
1	0.90± 0.01
2	0.90± 0.01
5	0.88± 0.01
10	0.78± 0.04
20	0.68± 0.05

G.3 Additional Scatter Plots and Intensity Plots

We show additional scatter and intensity plots in Figure G.7. Note that for both architectures there are a few million weights. The scatter plots are overcrowded and show the whole range of dependencies between two measure that can possibly occur. Intensity plots show the dependencies that the majority of weights adheres to.

G.4 ALTERNATIVE COMPARISON OF SI, SIU, SIB TO AF, ONAF

Finally, we point out that our comparisons between SI, SIU, SIB and AF, OnAF are usually obtained after applying a $\max(0, \cdot)$ (c.f. equation equation 4 of main paper) to the SI (and SIU, SIB) importances as the algorithms diverge without this operation. We present results without this max operation in G.4, further showing that (1) correlation between OnAF and SI is weakened by strong regularisation and (2) the correlation is due to the bias of SI.

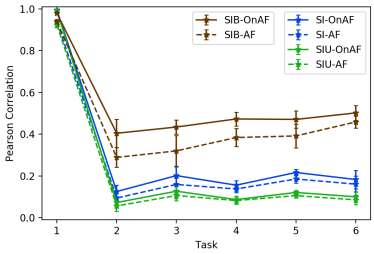

Figure G.4: Analogous to Figure 2 (middle) from main paper, comparing SI, SIU, SIB but before the operation $\max(\cdot, 0)$ is applied to importances of SI, SIU, SIB. See fulltext. Data from CIFAR tasks. On P-MNIST the operation $\max(\cdot, 0)$ has close to no effect.

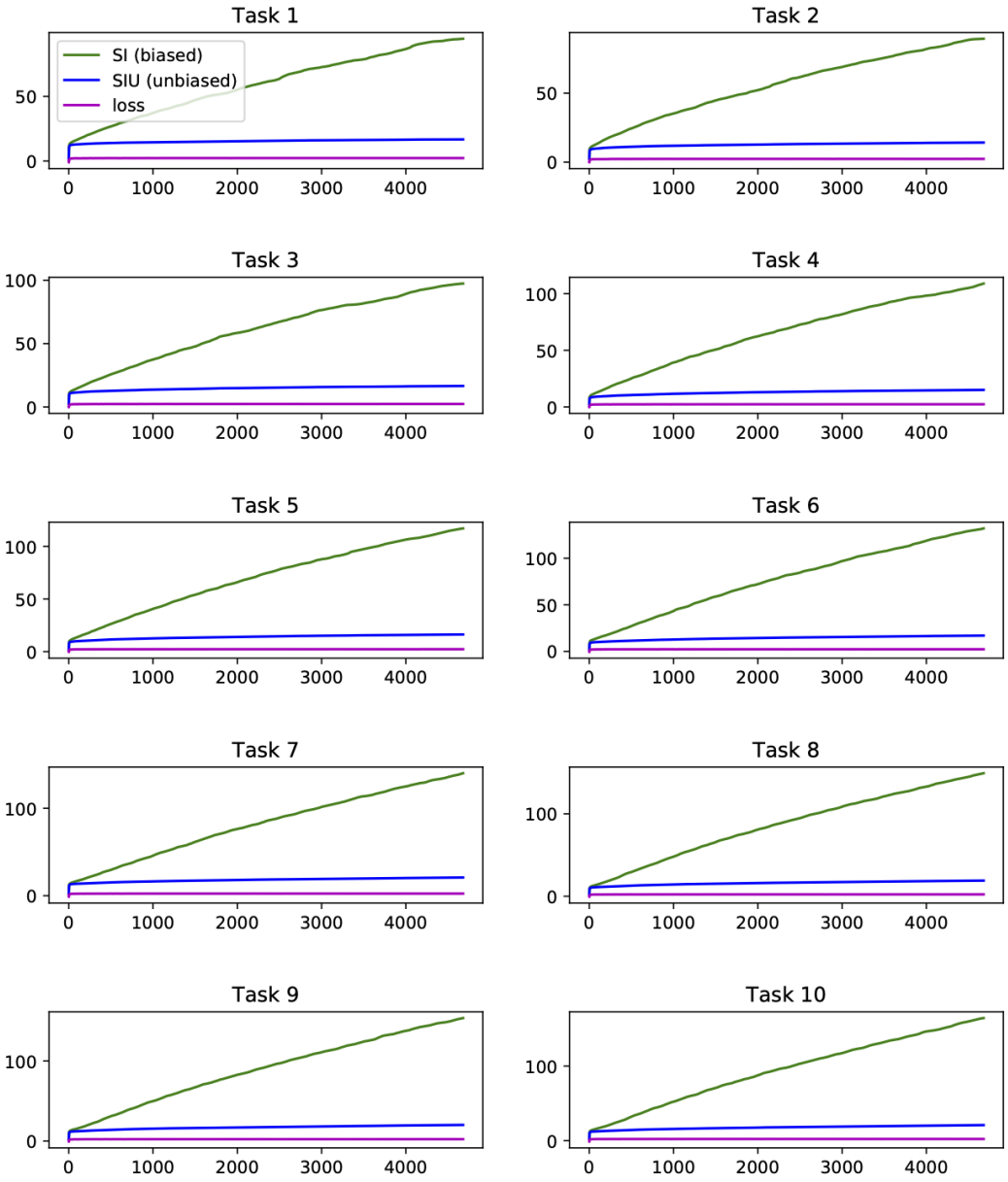

Figure G.5: Summed importances for all P-MNIST tasks for SI and its unbiased version. Analogous to Figure 2 (left) from main paper.

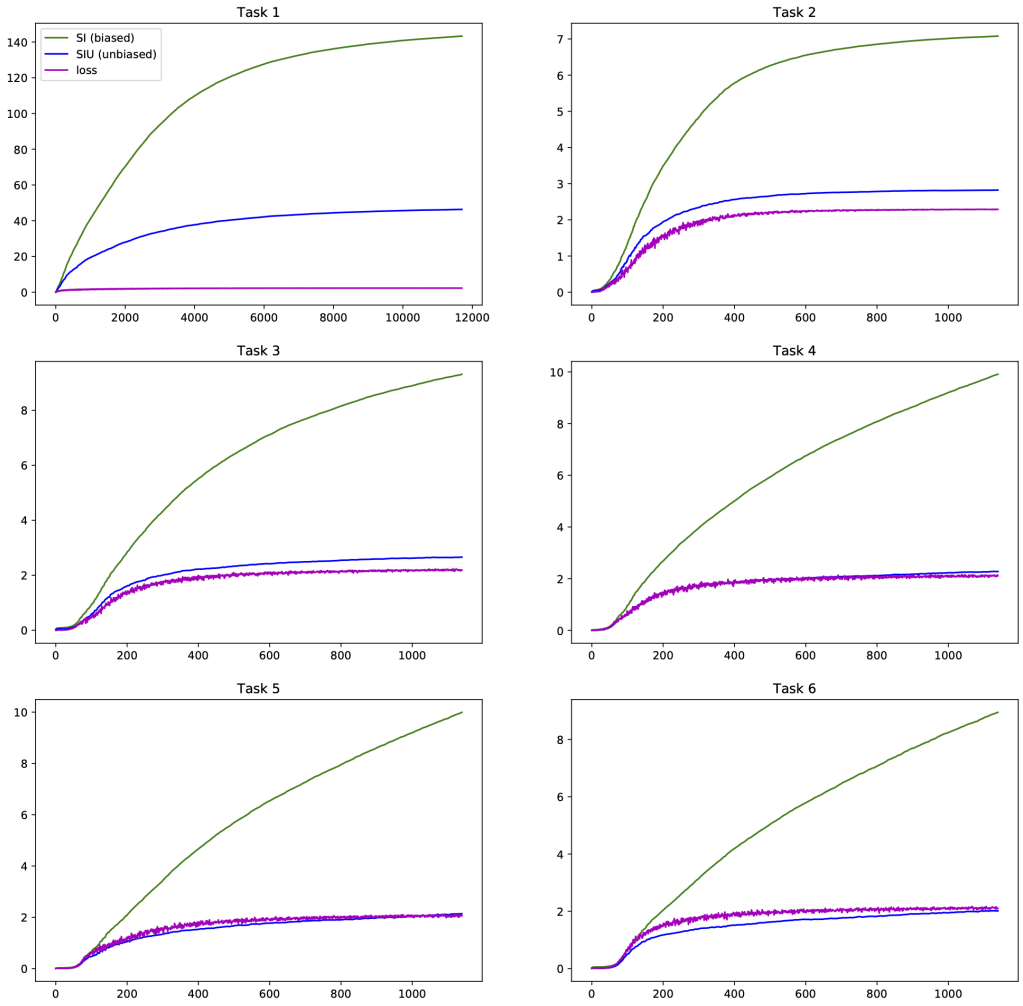

Figure G.6: Summed importances for all Split CIFAR 10/100 tasks for SI and its unbiased version. Analogous to Figure 2 (left) from main paper.

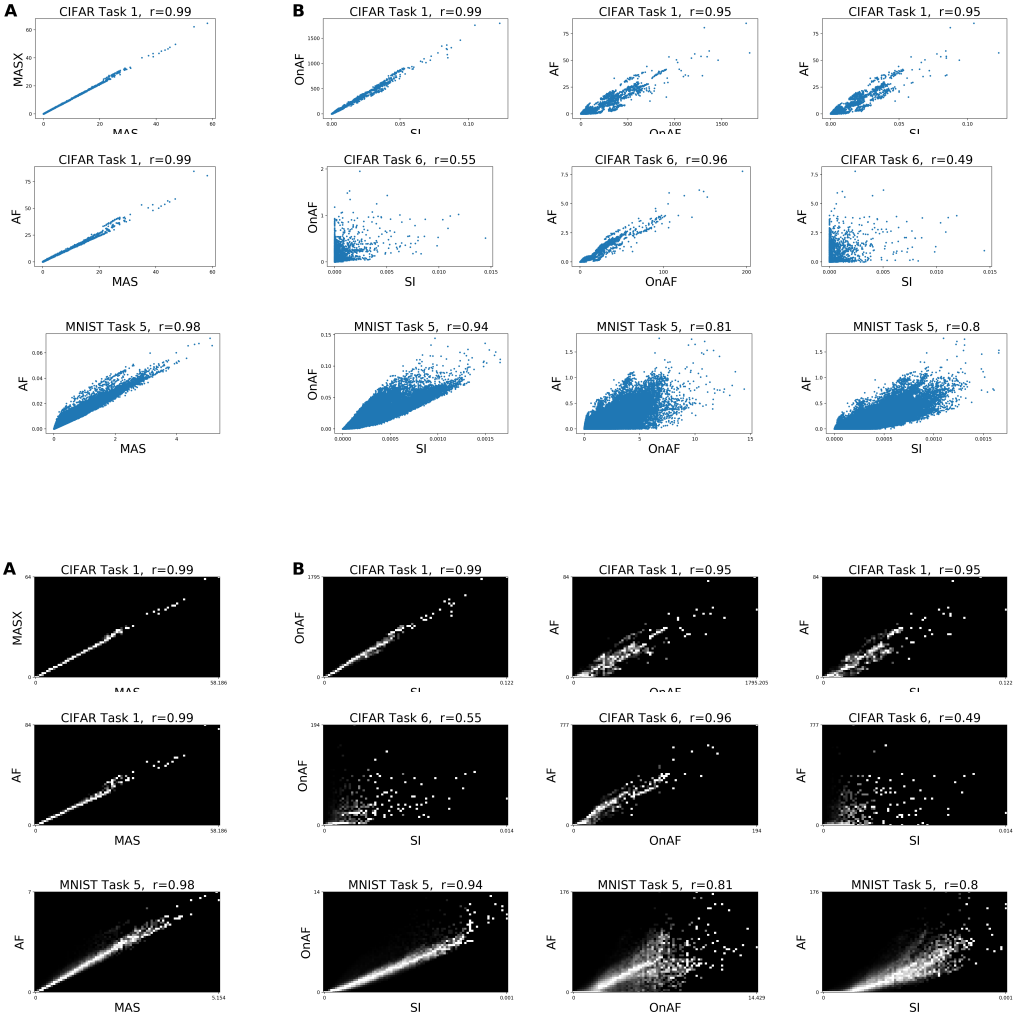

Figure G.7: More scatter and intensity plots. Refer to the title of each panel for details.