Learning to Rewind via Iterative Prediction of Past Weights for Practical Unlearning

Jinhyeok Jang[1,2], **Jaehong Kim**[1], **Chan-Hyun Youn**[2*]
[1]ETRI
[2]KAIST
jjh6297@etri.re.kr, jhkim504@etri.re.kr, chyoun@kaist.ac.kr

Abstract

In artificial intelligence (AI), many legal conflicts have arisen, especially concerning privacy and copyright associated with training data. When an AI model's training data incurs privacy concerns, it becomes imperative to develop a new model devoid of influences from such contentious data. However, retraining from scratch is often not viable due to the extensive data requirements and heavy computational costs. Machine unlearning presents a promising solution by enabling the selective erasure of specific knowledge from models. Despite its potential, many existing approaches in machine unlearning are based on scenarios that are either impractical or could lead to unintended degradation of model performance. We utilize the concept of weight prediction to approximate the less-learned weights based on observations about further training. By repetition of 1) finetuning on specific data and 2) weight prediction, our work gradually eliminates knowledge about the specific data. We verify its ability to eliminate side effects caused by problematic data and show its effectiveness across various architectures, datasets, and tasks.

Code — https://github.com/jjh6297/InvWNN

Introduction

In recent decades, artificial intelligence (AI) has undergone remarkable advancements, primarily driven by the exponential growth of datasets. Historically, AI development has prioritized performance enhancement and generalization capabilities. However, this narrow focus on performance has led to various challenges. For example, many face recognition studies have neglected privacy considerations, igniting numerous legal disputes. In response, regulatory bodies like the European Union (EU) have implemented stringent privacy regulations (EU 2023).

A significant challenge arises when training data for an AI model is flagged for privacy concerns. Under such circumstances, it is crucial to develop a new model that is devoid of knowledge derived from the problematic data. Retraining a model from scratch, however, poses practical difficulties due to the extensive computational resources required and the necessity for complete training datasets. Machine unlearning emerges as a solution, targeting the selective forgetting of specific knowledge within trained models (Cao and Yang 2015). Despite its promise, several issues within the current unlearning methodologies remain unresolved.

First, many existing unlearning approaches require access to the entire training dataset, which is often impractical due to accessibility constraints or the heavy computational cost to process large-scale datasets. If the complete dataset is available and manageable, retraining from scratch is typically more effective than attempting to unlearn. Or, with large-scale datasets, the computational burden becomes untenable. Moreover, this challenge escalates with the scale of the dataset. As a result, it's essential to devise unlearning methods that depend solely on the specific data that needs to be forgotten. Second, some unlearning strategies inadvertently introduce errors by fine-tuning models using incorrect ground truths for the data targeted for forgetting. This approach introduces incorrect knowledge rather than erase the knowledge about the specific data, and may corrupt the model's accuracy on unrelated data. Additionally, these methods are predominantly applicable to categorical classification tasks and do not easily extend to other types of AI applications, such as image synthesis. Third, methods like NegGrad (Golatkar, Achille, and Soatto 2020a) and Task Vector (Ilharco et al. 2022), which approximate less-learned weights via gradient ascent or linear extrapolation, often underperform on complex deep neural network (DNN) architectures or intricate tasks, underscoring their limitations.

In this paper, we propose a novel approach of machine unlearning by leveraging the concept of weight prediction—a method typically used for acceleration of training. Specifically, we introduce a method of DNN-based past weight prediction by inverting the typical use of weight prediction to estimate less informed past weights from future weights. This method involves two key steps: 1) fine-tuning on specific data to track future weight trajectories, and 2) predicting past weights based on these observations. Through iterative application of these steps, our method progressively eliminates knowledge associated with the problematic data. Our contributions are threefold: 1) introducing a new machine unlearning method based on weight prediction without necessity of the entire training data, 2) establishing an evaluation protocol measuring the removal of side effects, and 3) providing extensive analyses that demonstrate general applicability and visualize the unlearning process.

*Corresponding Author

Related Works

In this section, we delve into 1) objective: machine unlearning with only forget data, and 2) method: weight prediction.

Machine Unlearning

Machine unlearning is a task to force a trained neural network to forget some selected data. Consider a scenario where we have two datasets: the remaining dataset ($\mathcal{D}_r$) and forget dataset ($\mathcal{D}_f$). Suppose a model $\mathcal{F}$ with pretrained weights Θ_{PT} has been trained on the $\mathcal{D}_r \cup \mathcal{D}_f$. Then, machine unlearning aims to force $\mathcal{F}$ to forget $\mathcal{D}_f$, while remaining the knowledge about $\mathcal{D}_r$.

Unlearning with both $\mathcal{D}_f$ and $\mathcal{D}_r$ Numerous studies have utilized the remaining dataset, $\mathcal{D}_r$, for their unlearning processes. We can categorize them into two primary strategies:
Finetuning & Distillation. A foundational strategy involves finetuning the model $\mathcal{F}$ on $\mathcal{D}_r$, which facilitates the gradual forgetting of $\mathcal{D}_f$ via catastrophic forgetting. Certified removal (Guo et al. 2020) utilizes a Newton update coupled with loss perturbation to achieve unlearning. Similarly, SCRUB (Kurmanji et al. 2023) implements selective distillation by employing both $\mathcal{D}_r$ and $\mathcal{D}_f$, using KL-divergence.
Weight Noising. Some methods analyze $\mathcal{F}$ to identify weight parameters sensitive to $\mathcal{D}_r$ and inject random noise inversely proportional to their sensitivity. (Golatkar, Achille, and Soatto 2020a) were pioneers in this approach, employing the Fisher Information Matrix (FIM) to gauge weight-parameter sensitivity throughout $\mathcal{D}_r$. However, due to the high computational demands, subsequent studies sought simplifications. For instance, SSSE (Peste, Alistarh, and Lampert 2021) used a single-shot FIM, and Neural Tangent Kernels (Golatkar, Achille, and Soatto 2020b) have approximated $\mathcal{F}$ as a linear model to reduce complexity.

Unlearning with only $\mathcal{D}_f$ The unlearning with $\mathcal{D}_r$ presents practical challenges primarily because $\mathcal{D}_r$ is often unattainable or too extensive for effective computation. Moreover, if $\mathcal{D}_r$ is available, retraining from scratch is typically more feasible and effective than attempting to unlearn. Thus, there is a pressing need to develop unlearning methods that operate effectively with only the problematic subset $\mathcal{D}_f$. Existing approaches in this domain fall into two categories:

Introducing incorrect knowledge. Some studies have attempted to overwrite specific knowledge about $\mathcal{D}_f$ with incorrect information. For example, the method of random label (Hayase, Yasutomi, and Katoh 2020) involves finetuning only on $\mathcal{D}_f$ with randomized labels. Similarly, boundary unlearning (Chen et al. 2023) employs the most appealing incorrect label identified via adversarial attacks like FGSM (Goodfellow, Shlens, and Szegedy 2014). These methods effectively confuse the model $\mathcal{F}$ concerning $\mathcal{D}_f$, but rather than erasing the knowledge, they verge on "learning to malfunction." While these concepts are similar, "learning to malfunction" may cause the model $\mathcal{F}$ to incorrectly process other data similar to $\mathcal{D}_f$, spreading misknowledge beyond the intended scope. In addition, SalUn (Fan et al. 2023) is a add-on that selects some weights sensitive to $\mathcal{D}_f$. Basically, it can be applied various unlearning methods, but its baseline approach is ad-hoc on random label.

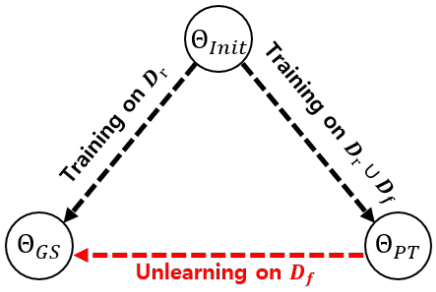


Figure 1: Ideal relations among Θ_{GS} and Θ_{PT}, and Θ_{Init}.

Rewinding toward past weights. This approach involves reverting to a state prior to training on $\mathcal{D}_f$. NegGrad (Golatkar, Achille, and Soatto 2020a), for example, uses gradient ascent on $\mathcal{D}_f$, which theoretically is the inverse of gradient descent and approximate the past weights before training on $\mathcal{D}_f$. Also, Task Vector (Ilharco et al. 2022) finetunes model $\mathcal{F}$ on $\mathcal{D}_f$ and then approximates past weights through linear extrapolation, using weight states before and after finetuning. This method reflects a refinement of NegGrad by considering the longer-term training dynamics and was the first to propose a two-step process: 1) forward finetuning on $\mathcal{D}_f$, followed by 2) an inversion of the updates to approximate less informed past weights.

Learning to Weight Prediction

Weight update prediction involves learning the changes in weights during training through meta-learning. *Learning to Optimize* (L2O) (Andrychowicz et al. 2016; Lv, Jiang, and Li 2017; Wichrowska et al. 2017; Huang et al. 2022) focuses on predicting more effective updates for back-propagation based on prior training experiences. L2O replaces traditional optimizers like SGD with a DNN-based optimizer, aiming for more efficient training updates. In a different approach, Introspection (Sinha et al. 2017) and WNN (Jang et al. 2023) forecast far future weights, enabling the skipping of multiple training epochs, and thus serving as ad-hoc boosters in conjunction with analytic optimizers. This is achieved by training small DNNs on historical weight data from simpler image classification tasks. Introspection was the pioneer in weight forecasting, and WNN expanded the concept with periodic nowcasting, applying it across various architectures, datasets, and tasks. Generally, weight prediction aims at enhancing training efficiency.

Problems

Unlearning involves approximating unlearned weights (Θ_{UL}) to function as gold standard weights (Θ_{GS}), which are trained on only $\mathcal{D}_r$. This process starts from pretrained weights (Θ_{PT}) that were trained on $\mathcal{D}_r \cup \mathcal{D}_f$. The relations among Θ_{GS}, Θ_{PT}, and initial weights (Θ_{Init}) are depicted in Fig. 1. Many prior studies (Tarun et al. 2023; Liu et al. 2023; Foster, Schoepf, and Brintrup 2024; Kurmanji et al. 2023; Golatkar, Achille, and Soatto 2020a) of unlearning has **required access to both the remaining set $\mathcal{D}_r$ and forget set $\mathcal{D}_f$.** However, such approaches often face practical

challenges: if $\mathcal{D}_r$ is small and manageable, it is more feasible to retrain the model from scratch rather than to unlearn, which offers a superior solution. Conversely, if $\mathcal{D}_r$ is large, the computational costs become prohibitive, escalating with the dataset's size, making it impractical for efficient unlearning. Given that $|\mathcal{D}_r|$ is typically much larger than $|\mathcal{D}_f|$, unlearning specific knowledge using the entire training dataset is computationally burdensome. To address these practical limitations, two primary unlearning approaches have been developed that do not require $\mathcal{D}_f$: 1) Introducing Incorrect Knowledge, and 2) Rewinding.

The approaches of introducing incorrect knowledge (i.e., random label, SalUn, boundary unlearning) involves fine-tuning the model $\mathcal{F}$ on $\mathcal{D}_f$ with incorrect labels. Such fine-tuning leads to $\mathcal{F}$ malfunctioning for $\mathcal{D}_f$, but it inadvertently introduces false knowledge rather than facilitating forgetting. This can **degrade the model's performance on non-target data** and is generally **only applicable to categorical classification tasks**. In other domains, such as image synthesis, assigning incorrect labels is not feasible, thus limiting the applicability of these approaches. In contrast, rewinding strategies (i.e., NegGrad, and Task vector) do not encourage the learning of false information. **These approaches depend significantly on the complexity of the search space.** NegGrad works well in smoother search spaces by essentially reversing the learning process. However, as DNN architectures and tasks grow in complexity, simple methods like NegGrad become inadequate. Task Vector attempts to address this by considering the long-term training tendency. It employs a linear operation to estimate less informed past weights by weight deletion ($\Theta^t - \lambda(\Theta_{FT} - \Theta^t)$), where Θ^t indicates the weights at current timestep t, Θ_{FT} means the finetuned weights from Θ^t, and λ is a weighing factor. While this approach approximates less learned weights about $\mathcal{D}_f$, its reliance on linear operations limits its effectiveness in complex search spaces where non-linear dynamics dominate. Thus, it is required to develop a method considering non-linear dynamics in long-term weight trajectory.

Method

In this paper, we enhance the Task Vector approach by accounting for non-linearity in the long-term weight trajectory from Θ^t to Θ_{FT}. We introduce a DNN-based past weight predictor, termed InvWNN, which reverses the conventional application of weight prediction—traditionally used to improve training convergence—into a "learning to rewind" mechanism. InvWNN is trained on extensive training histories sourced from diverse datasets, architectures, and training recipes. This training enables InvWNN to capture the general tendencies of weight trajectories, equipping it with the ability to predict past weights that are less learned about $\mathcal{D}_f$. Through this approach, we extend the linear operation of the Task Vector into a non-linear one, significantly enhancing the model's ability to unlearn specific data effectively. Please note that the evaluation setting (dataset, architecture, training recipe, and initialization) was not included in training data of InvWNN for fair evaluation.

Past Weight Prediction

Numerous studies have explored the relationship between past and future weights in DNNs, aiming to harness this understanding to enhance the training process (Andrychowicz et al. 2016; Sinha et al. 2017; Jang et al. 2023). One notable approach is the Weight Nowcasting Network (WNN) (Jang et al. 2023), which is designed as a distinct module to comprehend and predict the trajectories of weights during training. WNN accelerates the training process by predicting future weights based on current trends. We extend the WNN by reversing its original function to predict **past weights** to serve unlearning as depicted in Fig. 2. This approach, termed Inverse WNN (InvWNN), is based on the premise that past weights are less trained by given data, making them ideal for unlearning. We have developed InvWNN to estimate the discrepancy between current weights ($\hat{\Theta}^t$) and weights from U epochs prior ($\hat{\Theta}^{t-U}$). This estimation relies on observed weight changes over the subsequent K epochs ($[\Theta^t, \Theta^{t+1}, \Theta^{t+2}, \ldots, \Theta^{t+K-1}]$), as:

$$\hat{\Theta}^{t-U} = PastPredict([(\Theta^t, \Theta^{t+1}, \Theta^{t+2}, \ldots, \Theta^{t+K-1})]). \tag{1}$$

Then, the $PastPredict(\cdot)$ can be described as:

$$\Theta = \{\theta_i\}_{i=1,2,\ldots,N}, \tag{2}$$

$$W_i^t = [\theta_i^t, \theta_i^{t+1}, \theta_i^{t+2}, ..., \theta_i^{t+K-1}], \tag{3}$$

$$dW_i^t = [(\theta_i^{t+1} - \theta_i^t), ..., (\theta_i^{t+K-1} - \theta_i^{t+K-2})], \tag{4}$$

where i means the index of the weight parameter, and N represents the total number of weight parameters in the target network. We then input W_i^t and dW_i^t into our InvWNN to predict the residual in the direction of the past weight. InvWNN-based past weights prediction can be given as:

$$\hat{\theta}_i^{t-U} = \theta_i^t + InvWNN(W_i^t, dW_i^t), \tag{5}$$

$$\hat{\Theta} = \{\hat{\theta}_i\}_{i=1,2,\ldots,N} \tag{6}$$

Subsequently, we trained our InvWNN with the objective of minimizing the ℓ_1 residual error as :

$$\left| (\theta_i^t + InvWNN(W^t, dW^t)) - \theta_i^{t-U} \right|_1. \tag{7}$$

For training InvWNN, we utilized the dataset provided by Jang et al. (Jang et al. 2023) comprising weight histories collected across diverse conditions, including different architectures, datasets, and training recipes.

Past Weight Prediction Targeting only Forget Data

To unlearn solely from $\mathcal{D}_f$, we first fine-tune the target model on this dataset for K epochs and capture the weight changes at each epoch. This sequential weight history is then fed into InvWNN to guide the determination of the optimal unlearning discrepancy. InvWNN strategically considers the trajectory of forward fine-tuning. Further, this approach is versatile across various loss functions. This is similar to NegGrad and Task vector, but differs in consideration of non-linearity in long-term forward trajectory, and employing extra knowledge from numerous training experiences.

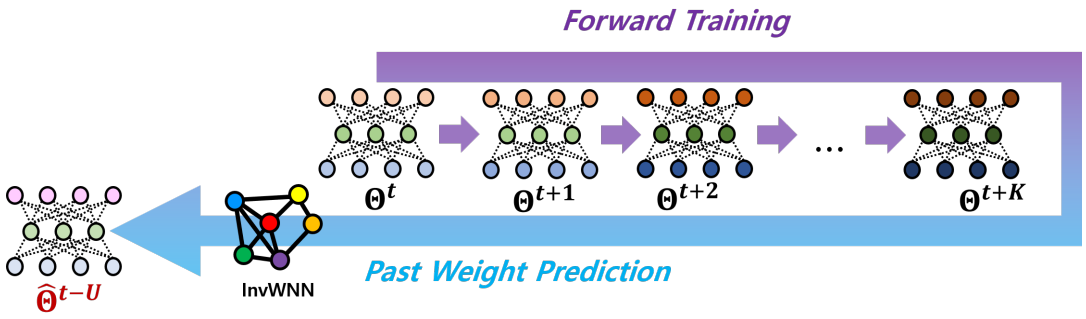


Figure 2: Schematic of past weight prediction via InvWNN.

Our approach yields three key benefits: 1) identifying a reasonable direction for unlearning, 2) enabling efficient unlearning without the need for $\mathcal{D}_r$, and 3) broad applicability to tasks beyond just classification.

Iterative Past Weight Prediction

Weight prediction can selectively remove knowledge about $\mathcal{D}_f$, but it only approximates the weights from U epochs prior to the current epoch. However, machine unlearning aims to erase the entire knowledge specifically about $\mathcal{D}_f$. To fully erase, we introduce **iterative past weight prediction**.

Our approach is based on the concept that Θ_{PT} is positioned along the trajectory of forward training from Θ_{GS}. To apply InvWNN to approximate Θ_{GS} starting from Θ_{PT}, we use procedure as described in Algorithm 1. Note that Θ^t denotes the weights at timestep t, $\mathcal{L}(\cdot)$ refers to the loss function. Our approach focuses on 1) finetuning on $\mathcal{D}_f$ for K epochs to stack weight trajectory as input of InvWNN, and 2) iterative application of InvWNN for gradual unlearning.

Algorithm 1: Iterative Past Weight Prediction

$\Theta^T_{Ours} \leftarrow \Theta_{PT}$
for t=T, $T-1$,..., 1 **do**
  $\Phi^t \leftarrow \Theta^t_{Ours}$
  for k=0,1,...,$K-2$ **do**
    $\Phi^{t+k+1} \leftarrow \Phi^{t+k} - \frac{\partial \mathcal{L}(\mathcal{D}_f;\Phi^{t+k})}{\partial \Phi^{t+k}}$
  end for
  $\Theta^{t-1}_{Ours} \leftarrow PastPredict([\Phi_t, \Phi_{t+1}, ..., \Phi^{t+k-1}])$
end for
return Θ^0_{Ours}

Experiments

Standard Unlearning Experiment Using Only $\mathcal{D}_f$

We conducted a standard experiment in unlearning. For the pretraining, we trained ResNet18 (He et al. 2016) on CIFAR10 dataset (Krizhevsky, Hinton et al. 2009) for 200 epochs from scratch. Also, we used Adam optimizer with learning rate decay from 1e-3, and spatial transformation as data augmentation. We randomly selected 50% of the training data as $\mathcal{D}_f$ to apply our work and various prior methods:

Random Label (Hayase, Yasutomi, and Katoh 2020): Finetuning on $\mathcal{D}_f$ with random labels.

Random Label+SalUn (Fan et al. 2023): Applying Random Label on only selected weights sensitive to $\mathcal{D}_f$.

Boundary Unlearning (Chen et al. 2023): Finetuning on $\mathcal{D}_f$ with attractive incorrect labels obtained by FGSM (Goodfellow, Shlens, and Szegedy 2014).

NegGrad (Golatkar, Achille, and Soatto 2020a): Finetuning using *gradient ascent* on $\mathcal{D}_f$.

Task Vector (Ilharco et al. 2022): A two-step process involving finetuning on $\mathcal{D}_f$ from Θ^t to Θ_{FT}, followed by weight-level deletion using the formula $\Theta^t - \lambda(\Theta_{FT} - \Theta^t)$.

All methods in comparison can work with only $\mathcal{D}_f$. Notably, 1) we utilized a pretrained InvWNN *without any task-specific finetuning*, 2) ResNet18 was not included in the collection of training data for InvWNN, and 3) each experiment commenced with a unique random initialization.

Following SalUn (Fan et al. 2023), we measured 1) Unlearning accuracy (UA), i.e., 100%− the accuracy on $\mathcal{D}_f$, 2) remaining accuracy (RA), i.e., the accuracy on $\mathcal{D}_r$, 3) test accuracy (TA), i.e., the accuracy on test dataset, and 4) membership inference attack (MIA) (Shokri et al. 2017) to evaluate performance. We also prepared a ResNet18 trained solely on $\mathcal{D}_r$ from scratch. Many previous studies in machine unlearning have not specified termination conditions, crucial when unlearning with only $\mathcal{D}_f$. Without proper constraints, both TA and RA can degrade to 0%. For fair comparison, we adopted a termination condition: reaching an UA of unlearned model to the UA of the retrained model.

Table 1 shows that all methods exhibited comparable RA, FA, and TA values. Among them, our approach achieved outstanding results, particularly in terms of RA and TA, while maintaining similar UA values. However, we concluded that this experiment alone is insufficient for validation, so we further evaluated the methods based on their effectiveness in side effect removal.

Side Effect Removal

In this section, we delineate our experimental setup designed to assess the efficacy of different unlearning methods with a focus on the elimination of side effects caused by problematic training data ($\mathcal{D}_f$). We defined three key performance attributes for an ideal unlearning method: 1) significant degradation in accuracy for $\mathcal{D}_f$, indicating effective unlearning; 2) sustained performance on the remaining data ($\mathcal{D}_r$), demonstrating preservation of useful knowledge; and

Method	Termination Condition: UA $>$ 8.11%			
	RA (%)	UA (%)	TA (%)	MIA (%)
Pretrained	99.96±0.02	0.02±0.01	94.38±0.10	9.75±1.61
Retrain	99.92±0.02	8.11±0.19	91.45±0.32	29.07±0.42
Random Label	91.03±0.55	8.99±0.53	83.63±0.44	33.19±2.46
Boundary	91.24±0.21	8.42±0.16	83.72±0.10	28.51±3.97
SalUn	91.33±0.45	8.58±0.39	84.45±0.35	32.62±3.17
NegGrad	91.26±0.31	8.59±0.11	85.01±0.95	**28.56±9.60**
Task Vector	92.03±0.72	8.54±0.54	83.29±0.82	30.86±6.65
Proposed	**92.24±0.71**	8.31±0.24	**85.04±1.10**	31.94±4.56

Table 1: Results of five trials for unlearning randomly selected 50% of CIFAR10. Note that better performance corresponds to a smaller gap with the retrained model.

3) complete removal of side effects induced by $\mathcal{D}_f$. While previous approaches such as random label have predominantly addressed the first two attributes, our experiments also incorporated a novel protocol to evaluate the third attribute, which includes scenarios such as:

Label noise: Assessing the ability to eliminate errors introduced by noisy labels.

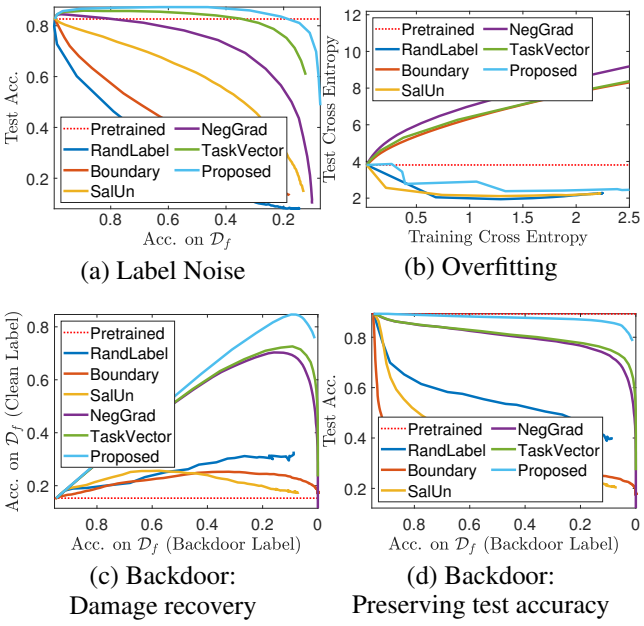
Overfitting: Evaluating the reduction in model overfitting.

Backdoor attacks: Testing the ability to rectify misfunctions due to backdoor attacks.

For these experiments, we utilized CIFAR10 as our dataset and ResNet18 as our model architecture. We trained four different ResNet18 models on CIFAR10, each modified with label noise, reduced sample sizes, and backdoor attacks to establish our pretrained weights (Θ_{PT}). Various unlearning methods were then applied starting from these pretrained weights. For evaluation, we focused on two primary metrics: the elimination of side effects as specified in our new protocol, and the accuracy on the **clean CIFAR10 testset**. The results are visually represented, illustrating the unlearning process progressing from left to right.

Removing Confusion due to Label Noise Similarly to some prior works (Kurmanji et al. 2023; Goel et al. 2022), we induced confusion by introducing label noise to the CIFAR10 training dataset. Specifically, we randomly altered the labels of 20% of the training data with incorrect values, which reduced the performance of the trained ResNet18, manifesting as an unintended side effect. We expected that ideal unlearning would lead to both an increase in test accuracy and a decrease in misclassifications on $\mathcal{D}_f$.

The results are shown in Fig. 3(a). The results reveal that methods such as random label, boundary unlearning, and SalUn were unable to correct the damage. Rather than eliminating the acquired knowledge about $\mathcal{D}_f$, these methods further compounded the issue by embedding new incorrect information. This intervention resulted in additional misclassifications on $\mathcal{D}_f$, failing to restore the degraded test accuracy. NegGrad and Task vector showed a slight improvement in test accuracy initially, but a gradual decrease during the unlearning process. Our work showed notable improvements in test accuracy and well preserved higher accuracy, indicating its superiority in selectively removing and fixing the inaccurate knowledge about $\mathcal{D}_f$. This suggests that our approach is more adept at selectively eliminating and correcting inaccurate knowledge associated with $\mathcal{D}_f$, validating the effectiveness of our unlearning strategy. Also, it is shown that rewinding approaches can partially rectify the side effects, while approaches based on false knowledge fail.



(a) Label Noise (b) Overfitting

(c) Backdoor: Damage recovery (d) Backdoor: Preserving test accuracy

Figure 3: Unlearning results for (a) label noise, (b) overfitting, and (c-d) backdoor attacks. The x-coordinate represents the unlearning process, from less (left) to more (right). Better unlearning is indicated by higher test accuracy (label noise), lower cross-entropy (overfitting), and higher accuracy with clean labels or test accuracy (backdoor attacks) at the same x-coordinate.

Alleviating Overfitting Through Unlearning To evaluate a remedy for overfitting, we overtrained a ResNet18 on only 1% CIFAR10 training data for an excessive number of epochs. This resulted in clear overfitting, shown by a rise in test loss beyond a certain point. We applied various methods to this 1% training data, expecting effective unlearning to alleviate overfitting and reduce the test loss.

The results are depicted in Fig. 3(b). Notably, methods such as random label and SalUn, showed effectiveness in partially mitigating overfitting because they work as random noise injection similarly to perturbative gradient descent. In contrast, other prior works (i.e., Boundary unlearning, NegGrad, and Task vector) progressively deteriorated the model's performance, underscoring their limitations in this context. Our work shows quite reasonable results in overfitting case. These results indicate that our approach successfully identifies a viable direction for unlearning, aligning with the weights from earlier epochs. Also, it is shown that NegGrad and Task vector are weak in rewinding for highly dynamic search space, indicating the necessity of InvWNN.

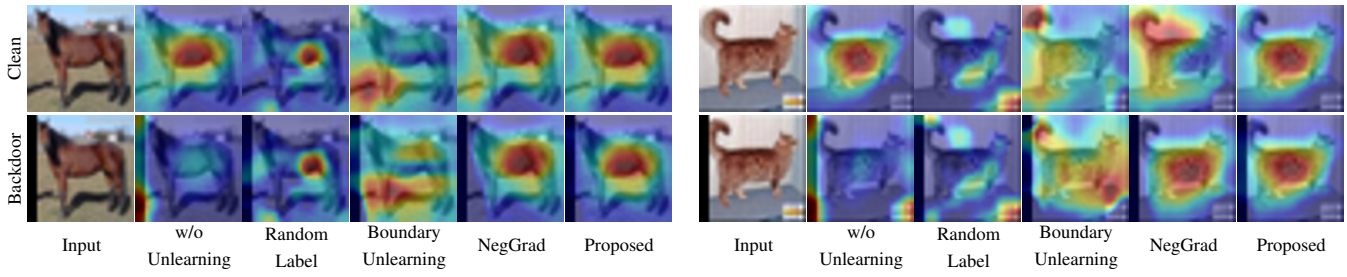


Figure 4: CAM for backdoor data of each unlearning method. The black line at left of backdoor images is the hidden signature.

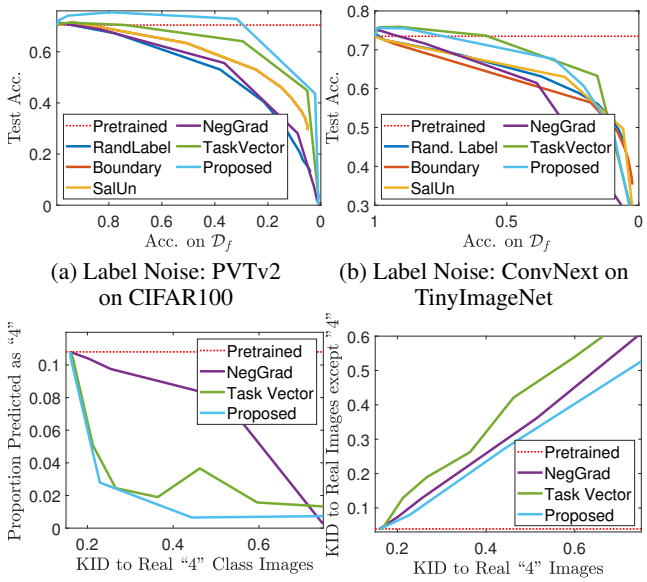
Rectifying Backdoor Attack Damages We employed the BadNet (Gu et al. 2019) method, a popular backdoor attack, which involves embedding signatures to CIFAR10 images and altering their labels to a predetermined adversarial label. This manipulation results in a trained model misclassifying any signed image as the adversarial label, regardless of its original label. For our experiments, we designated 20% of the training data as the target for the BadNet attack ($\mathcal{D}_f$) and trained a ResNet18 model using this compromised dataset. It was expected that the ideal unlearning on the backdoor data can rectify the intentional misclassifications.

The results are presented in Fig. 3 (c-d). The misclassification for images with the hidden signature, a result of the backdoor attack, is evident. About 93% of $\mathcal{D}_f$ with the hidden signature was incorrectly classified as the adversarial label, with only about 7% accuracy for the original label, as illustrated in Fig. 3(c). Methods such as NegGrad and Task Vector proved effective in improving the accuracy for $\mathcal{D}_f$ with the correct labels while maintaining overall test accuracy. In contrast, other approaches exacerbated the issue by introducing new errors, leading to further misclassifications. Our method demonstrated superior performance, outpacing previous works in restoring accurate classification of the affected images. Also, Fig. 4 shows the class activation map (CAM) (Zhou et al. 2016), captured after achieving 10% accuracy for $\mathcal{D}_f$ with the adversarial label. This CAM further validates the effectiveness, as our approach successfully removes the specific focus on the signature without damage.

General Applicability

The previous sections demonstrated the efficacy using CIFAR10 and ResNet18 with various side effects, providing a controlled environment to assess our method. To establish the generalizability of our InvWNN across diverse scenarios, this section details additional experiments incorporating more complex datasets, advanced DNN architectures, and a further task. Please note that these new settings were not part of the training data for InvWNN, ensuring fair evaluation.

Robustness to Novel Datasets and Architectures We have broadened the application of our methodology to two challenging scenarios involving label noise: 1) CIFAR100, using the Pyramid Vision Transformer v2 (PVTv2) (Wang et al. 2022), and 2) TinyImageNet, implemented with ConvNext (Liu et al. 2022). These datasets, which are more intricate than CIFAR10, provide a robust benchmark for evaluating the applicability to various data complexities. The use of advanced architectures (i.e.,PVTv2 and ConvNext) further allows us to assess the generalization capabilities of our approach across more sophisticated DNNs. We used ImageNet-pretrained weights for both cases. For each setting, we finetuned using entire training set with 20% label noised data and applied unlearning methods. As demonstrated in Fig. 5, the enhancement in test accuracy was observed only in our work and Task vector, indicating the necessity of long-term tendency for unlearning.



(a) Label Noise: PVTv2 on CIFAR100

(b) Label Noise: ConvNext on TinyImageNet

(c) Diffusion: Ratio of images classified as "4"

(d) Diffusion: KID to all real images

Figure 5: Unlearning across novel datasets, architectures, and tasks. For diffusion, a lower ratio (left) and lower KID (right) at the same x-coordinate indicate better unlearning.

Applicability to Further Task We applied InvWNN to the task of image synthesis, training a diffusion model (Ho, Jain, and Abbeel 2020) on the MNIST dataset with ℓ_2 loss to remove knowledge of the digit "4." Ideally, this would result in generating all digits except "4." Random label and boundary unlearning were inapplicable as they require classification labels. For evaluation, we compared our method to NegGrad and Task vector using two metrics: the ratio of generated images classified as "4" and the Kernel Inception

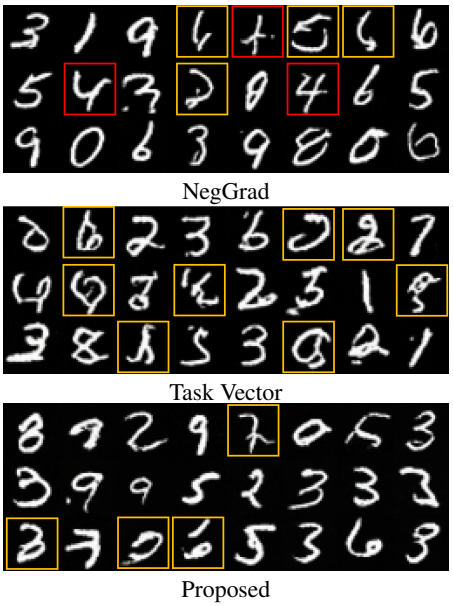


Figure 6: Generated images with similar KID ($\simeq$0.14), using the same random seed. Red boxes indicate images resembling "4," and yellow boxes indicate low-quality images.

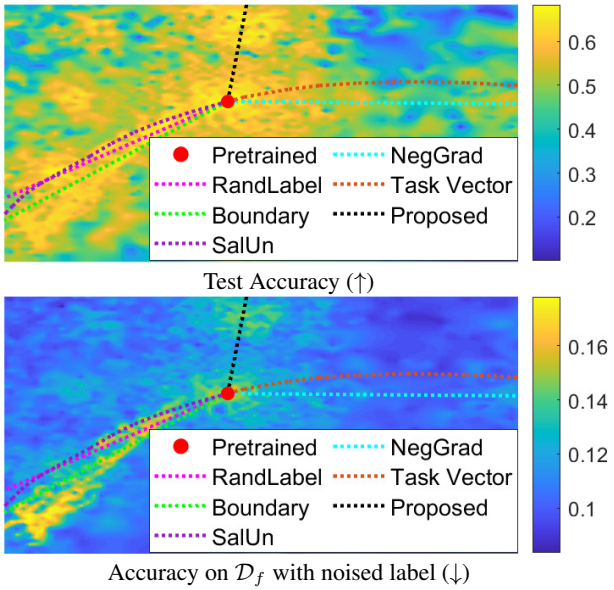


Figure 7: Landscape visualization of unlearning on label-noised data. The upper shows the test accuracy landscape, reflecting the ability to preserve accuracy. The lower depicts accuracy on $\mathcal{D}_f$, representing forgetting performance.

Distance (KID) (Bińkowski et al. 2018). To assess the first metric, we used a small classifier trained on MNIST (with about 99% test accuracy) to classify images generated by DDIM sampling (Song, Meng, and Ermon 2020). KID, a computationally efficient alternative to FID (Heusel et al. 2017), measures image generation quality, with lower KID values indicating better quality. The results, shown in Fig. 5(c-d), reveal that our method effectively reduced the number of synthetic images classified as "4" while maintaining high quality indicated by a low KID. Fig. 6 shows the generated images post-unlearnin, achieving a similar KID. Unlike NegGrad, which often generated "4"-like images, Task vector and our method did not generate any. However, Task vector generates many abnormal images, indicating high damage to the original performance. These findings reveal that our work can eliminate targeted knowledge, even in novel tasks and with unseen loss functions.

Visualization

To visualize the unlearning trajectories of various methods, we used a compact CNN with several convolution layers, global pooling, and a classification layer. This model was trained on $\mathcal{D}_r \cup \mathcal{D}_f$ of the label noise scenario to derive Θ_{PT}. We then created a range of weight variations around Θ_{PT} by adding random noise, and then evaluated test accuracy and accuracy on $\mathcal{D}_f$. For these weight sets, PCA was applied to reduce dimensions for visualization as landscape. We then plotted the trajectories of unlearning methods starting from Θ_{PT} on this landscape, as depicted in Fig. 7. Ideal unlearning updates move towards satisfying both the yellow region in the upper figure and the blue region in the lower figure. The trajectories depicted show that methods like Random Label and Boundary Unlearning tend towards areas of high test accuracy but fail to reduce accuracy on $\mathcal{D}_f$, suggesting their limited effectiveness. SalUn, which selectively updates sensitive weights, follows a trajectory close to that of Random Label but converges to a mid-point. These suggest that while they preserve test performance, they are less effective in forgetting $\mathcal{D}_f$. NegGrad and Task vector, conversely, rapidly decrease both the performance on $\mathcal{D}_f$ with label noise but also test accuracy, indicating significant disruption to the original task. In contrast, our method successfully avoids the regions with high accuracy on $\mathcal{D}_f$, while preserving high test accuracy. This visualization demonstrates our method's ability to find the correct direction for unlearning.

Conclusion

In this paper, we address several limitations identified in previous unlearning studies, including: 1) the dependence on remaining data ($\mathcal{D}_r$), 2) the risk of introducing incorrect knowledge, 3) limited applicability across different tasks, and 4) inefficacy in complex search spaces. To overcome the limitations, we introduce the InvWNN, a sub-module designed to facilitate unlearning through the past weight prediction. InvWNN leverages an understanding of the general tendencies in DNN training to accurately predict the past weights that is less trained on specific data. Crucially, InvWNN operates effectively using only the forget data ($\mathcal{D}_f$), eliminating the need for access to $\mathcal{D}_r$. Our experiments demonstrates that InvWNN successfully mitigates various side effects caused by problematic data, and surpasses existing methods in performance. Moreover, our work exhibits robust generalization capabilities, proving effective across a range of novel datasets, architectures, and tasks.

Acknowledgements

This work was partly supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government foundation (25ZB1200, Research of Human-centered Autonomous Intelligence System Original Technology, 60%), and Korea Institute of Marine Science & Technology Promotion (KIMST) funded by the Korea Coast Guard (RS-2023-00238652, Integrated Satellite-based Applications Development for Korea Coast Guard, 40%)

References

Andrychowicz, M.; Denil, M.; Colmenarejo, S. G.; Hoffman, M. W.; Pfau, D.; Schaul, T.; Shillingford, B.; and de Freitas, N. 2016. Learning to learn by gradient descent by gradient descent. In *NeurIPS 2016*.

Bińkowski, M.; Sutherland, D. J.; Arbel, M.; and Gretton, A. 2018. Demystifying MMD GANs. In *ICLR 2018*.

Cao, Y.; and Yang, J. 2015. Towards making systems forget with machine unlearning. In *S&P*, 463–480. IEEE.

Chen, M.; Gao, W.; Liu, G.; Peng, K.; and Wang, C. 2023. Boundary Unlearning: Rapid Forgetting of Deep Networks via Shifting the Decision Boundary. In *CVPR 2023*, 7766–7775.

EU. 2023. The EU AI Act. https://www.euaiact.com.

Fan, C.; Liu, J.; Zhang, Y.; Wong, E.; Wei, D.; and Liu, S. 2023. SalUn: Empowering Machine Unlearning via Gradient-based Weight Saliency in Both Image Classification and Generation. In *ICLR 2023*.

Foster, J.; Schoepf, S.; and Brintrup, A. 2024. Fast machine unlearning without retraining through selective synaptic dampening. In *AAAI 2024*.

Goel, S.; Prabhu, A.; Sanyal, A.; Lim, S.-N.; Torr, P.; and Kumaraguru, P. 2022. Towards adversarial evaluations for inexact machine unlearning. *arXiv preprint arXiv:2201.06640*.

Golatkar, A.; Achille, A.; and Soatto, S. 2020a. Eternal sunshine of the spotless net: Selective forgetting in deep networks. 9304–9312.

Golatkar, A.; Achille, A.; and Soatto, S. 2020b. Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations. 383–398. Springer.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Gu, T.; Liu, K.; Dolan-Gavitt, B.; and Garg, S. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7: 47230–47244.

Guo, C.; Goldstein, T.; Hannun, A.; and Van Der Maaten, L. 2020. Certified Data Removal from Machine Learning Models. In *ICML 2020*.

Hayase, T.; Yasutomi, S.; and Katoh, T. 2020. Selective Forgetting of Deep Networks at a Finer Level than Samples. *arXiv preprint arXiv:2012.11849*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS 2017*, 30.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *NeurIPS 2020*.

Huang, T.; Chen, T.; Liu, S.; Chang, S.; Amini, L.; and wang, Z. 2022. Optimizer Amalgamation. In *ICLR*.

Ilharco, G.; Ribeiro, M. T.; Wortsman, M.; Schmidt, L.; Hajishirzi, H.; and Farhadi, A. 2022. Editing models with task arithmetic. In *ICLR 2022*.

Jang, J.; Yun, W.-h.; Kim, W. H.; Yoon, Y.; Kim, J.; Lee, J.; and Han, B. 2023. Learning to boost training by periodic nowcasting near future weights. In *ICML 2023*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Kurmanji, M.; Triantafillou, P.; Hayes, J.; and Triantafillou, E. 2023. Towards unbounded machine unlearning. *NeurIPS 2023*, 36.

Liu, J.; Ram, P.; Yao, Y.; Liu, G.; Liu, Y.; SHARMA, P.; Liu, S.; et al. 2023. Model sparsity can simplify machine unlearning. *NeurIPS 2023*.

Liu, Z.; Mao, H.; Wu, C.-Y.; Feichtenhofer, C.; Darrell, T.; and Xie, S. 2022. A convnet for the 2020s. In *CVPR 2022*, 11976–11986.

Lv, K.; Jiang, S.; and Li, J. 2017. Learning gradient descent: Better generalization and longer horizons. In *ICML 2017*.

Peste, A.; Alistarh, D.; and Lampert, C. H. 2021. Ssse: Efficiently erasing samples from trained machine learning models. *arXiv preprint arXiv:2107.03860*.

Shokri, R.; Stronati, M.; Song, C.; and Shmatikov, V. 2017. Membership inference attacks against machine learning models. In *S&P 2017*.

Sinha, A.; Mukherjee, A.; Sarkar, M.; and Krishnamurthy, B. 2017. Introspection:Accelerating Neural Network Training By Learning Weight Evolution. In *ICLR 2017*.

Song, J.; Meng, C.; and Ermon, S. 2020. Denoising Diffusion Implicit Models. In *ICLR 2020*.

Tarun, A. K.; Chundawat, V. S.; Mandal, M.; and Kankanhalli, M. 2023. Fast yet effective machine unlearning. *IEEE Transactions on Neural Networks and Learning Systems*.

Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; and Shao, L. 2022. Pvtv2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3): 1–10.

Wichrowska, O.; Maheswaranathan, N.; Hoffman, M. W.; Colmenarejo, S. G.; Denil, M.; Freitas, N.; and Sohl-Dickstein, J. 2017. Learned optimizers that scale and generalize. In *ICML 2017*.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. In *CVPR*, 2921–2929.