Hierarchical Unlearning Framework for Multi-Class Classification

Abraham Chan
University of British Columbia
abrahamc@ece.ubc.ca

Arpan Gujarati
University of British Columbia
arpanbg@cs.ubc.ca

Karthik Pattabiraman
University of British Columbia
karthikp@ece.ubc.ca

Sathish Gopalakrishnan
University of British Columbia
sathish@ece.ubc.ca

Abstract

As machine learning (ML) systems require increasing quantities of data for training, regulations around ML privacy and fairness have been created. Machine unlearning (MU) aims to comply with such regulations by fulfilling data deletion requests on trained ML models. In multi-class classification, existing MU techniques often shift the weight of the forget data, to other classes through fine-tuning. However, such techniques do not scale well when a large number of classes are present. We propose HUF, a hierarchical unlearning framework to effectively process MU requests, by adopting a hierarchical classification architecture. We evaluate the efficiency of HUF by measuring the number of epochs needed to reach a similar MU efficacy to a retrained model, against both random data and class-wise forgetting. We find that HUF is able to unlearn with fewer epochs compared to a single unlearned model, while sustaining the test accuracy of the original model.

1 Introduction

Machine learning (ML) applications, such as those used in autonomous driving [1], object detection [30], and facial recognition [24], require large amounts of training data. The significance of training data has given rise to efforts to assure their privacy and fairness. Legislations, such as the General Data Protection Regulation (GDPR) [27] in the European Union, aim to protect individual privacy and data ownership by granting the *the right to be forgotten*, which is the fundamental right where individuals can request an online platform to remove or *forget* all of their associated data. Concurrently, service providers may wish to remove entire classes of data over concerns of ML fairness [4] to comply with legal and ethical obligations [28]. For example, CelebA [19], a large-scale facial recognition dataset, is overly biased with examples of people with blonde hair for the female class, which can have negative impact on trained facial recognition models [4]. As a result, service providers deploying ML applications need to provide a way to effectively remove training data and their influence from trained ML models.

Machine unlearning (MU) is the task of causing a trained ML model to forget individual data points or a class of data in the training data, without adversely affecting the accuracy on the remaining data points [29]. MU is usually achieved without retraining the model from scratch, which is resource demanding. For multi-class classification, the focus of our work, MU is usually implemented by shifting the ML model's weights to other classes in lieu of the original class of the data point. While the objective of MU is to maximize the loss of the training sample from its target label, the efficacy of MU also hinges on how well the predictions for the forget data point are distributed among remaining classes [10]. Otherwise, the forget data point can still be detected as a member of the training dataset, defeating the goal of MU. This random distribution is harder to achieve with datasets comprised of a large number of classes.

Through our experiments, we make two observations:

1. MU for multi-class classification is more efficient and efficable when fewer classes are present [15].
2. MU is more effective when weights are shifted to more dissimilar classes. We show this in Section 4.2.

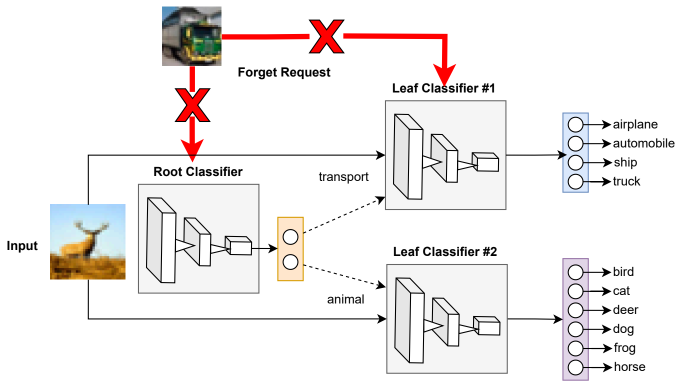


Figure 1: An example of a two-level HUF configured for the CIFAR-10 dataset.

We ask the following question: *how do we build scalable ML models that can effectively handle MU requests?* We propose Hierarchical Unlearning Framework (HUF), a novel unlearning framework inspired by hierarchical classification as shown in Fig. 1, to support MU in multi-class classification more effectively. To the best of our knowledge, our work is the first to address the problem of effectively accomodating MU requests, for both random data and class-wise forgetting, on multi-class classification models with large number of classes. As shown in Fig. 1, HUF introduces a tree-like hierarchy where a root classifier first classifies inputs into broader classes, while secondary leaf classifiers classifies inputs into finer classes. Similar classes are clustered together to form the broader classes. MU only needs to be conducted at the root classifier and the select leaf classifier. While we focus on building a two-level hierarchical architecture, HUF can be extended to additional levels. HUF improves the efficiency of MU by reducing the number of classes at each classifier to be unlearned, which leads to a faster unlearning time by requiring fewer unlearning epochs, where each unlearning epoch consists of unlearning the forget dataset and fine-tuning on the retained dataset.

HUF consists of two separate processes: (1) building the hierarchical unlearning architecture, and (2) handling an MU request. The first process is a one-time operation, where the ML model is evaluated against a test dataset, and its similarity matrix [9, 17] is analyzed. Similar classes are grouped together, which will be handled by the leaf classifiers of the neural network. The leaf classifiers are joined with a root classifier, representing the superclasses. Each individual classifier consists of a convolution neural network, which is fine-tuned to the reduced classes. The second process is a recurring operation for each MU request. HUF directs the data points to be unlearned to the relevant leaf classifier. This allows MU to be performed at the root classifier and the relevant leaf classifier, which is faster and efficacious than unlearning a single model with many classes. Since ML models are more likely to memorize training data as the number of features increase, MU can drastically degrade the test accuracy when more classes are present [26].

In summary, the contributions of this work are as follows:

- We propose HUF, a hierarchical unlearning architecture to effectively process MU requests. Applying agglomerative clustering [21] on the similarity matrix of a dataset, we automatically transform a single trained model into a hierarchical architecture, consisting of root and leaf classifiers. This architecture allows MU to be performed separately at the root and leaf classifiers.

- An evaluation of HUF's unlearning efficiency against two popular image classification datasets, CIFAR-10 and CIFAR-100 [16], against both random data and class-wise forgetting. We find that HUF is able to unlearn with fewer epochs while closely matching the MU efficacy of a model retrained from scratch without the training samples to be forgotten.

2 Related Work

Machine Unlearning (MU) approaches can be divided into two categories [29]: exact unlearning and approximate unlearning. Exact unlearning aims to completely remove the training data sample from the ML model. The objective is to produce a ML model that has never encountered the training data sample. SISA [2] is an example of exact unlearning, which isolates the training data in shards, and a separate model is trained per shard. The models are then combined as an ensemble. However, exact unlearning approaches such as SISA are computationally expensive and generally incur heavy memory overhead to store extra weights and backup models. Approximate unlearning aims to forget the influence of training data samples, subject to some guarantee, without retraining the model from scratch. Approximate unlearning approaches include random labelling [11, 18], where the forget data sample is intentionally labelled as another class. Approximate learning is much more efficient compared to exact unlearning. HUF supports the more effective application of approximate learning.

Hierarchical Classification (HC) leverages the hierarchical relations between classes to improve the classification accuracy [3, 8, 25]. HC has been successfully applied in *incremental learning*, where trained ML models learn additional information. Tree-CNN [22], is a tree-based HC architecture that has been found to robustly accomodate new data classes being added, by treating incremental learning requests as tree operations. However, unlike Tree-CNN, HUF does not assume that labels follow an apriori hierarchical ordering. Instead, HUF automatically infers hierarchical relations between label classes by analyzing similarity matrices.

3 Methodology

3.1 Machine Unlearning (MU)

We provide the existing formal definition of (ϵ, δ)-MU based on ϵ-differential privacy [5]. This definition quantifies the difference in distributions between an unlearned model and a retrained model (training from scratch without the forget data points). MU is considered successful if the difference is small.

Suppose $\mathcal{D}$ is a training dataset, while $\mathcal{R}$ is the output space. Suppose $\mathcal{A}$ is an ML algorithm, while $\mathcal{U}$ is a unlearning algorithm. Let $\mathcal{D}_f \in \mathcal{D}$ be the forget dataset. Let $\epsilon \geq 0$ represent the extent of privacy loss associated with a single data point, whereas $0 \leq \delta \leq 1$ indicates the likelihood that two datasets can be differentiated. Smaller values of ϵ and δ indicate the unlearned and retrained models are similar. Then, MU is formally defined [23] such that both Eq. (1) and Eq. (2) must be satisified:

$$\Pr[\mathcal{A}(\mathcal{D} \setminus \mathcal{D}_f) \in \mathcal{R}] \leq e^{\epsilon}\Pr[\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}_f, \mathcal{D}) \in \mathcal{R}] + \delta \tag{1}$$

$$\Pr[\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}_f, \mathcal{D}) \in \mathcal{R}] \leq e^{\epsilon}\Pr[\mathcal{A}(\mathcal{D} \setminus \mathcal{D}_f) \in \mathcal{R}] + \delta \tag{2}$$

3.2 Hierarchical Machine Unlearning

We modify the previously given MU definition over a single ML model to a new definition that applies to hierarchical classification.

Suppose there are M distinct classes in the training dataset $\mathcal{D}$. Let S be the superclasses of M such that $\phi : M \to S$ describes the mapping between a class M to its superclass S. Let $\mathcal{L}$ be defined as a conditional subset of M such that $\mathcal{L} = \{m \in M | \phi(m) = s\}$ for some superclass $s \in S$.

$\mathcal{A}_S$ is an ML algorithm trained on the superclasses only, which represents the root classifier. $\mathcal{A}_{\mathcal{L}}$ is an ML algorithm trained on a subset of classes, where every class belongs to the same superclass s, representing a leaf classifier. Then, we extend the MU definition as follows where both Eq. (3) and Eq. (4) must be satisified:

$$\Pr[\mathcal{A}_S(\mathcal{D}\backslash\mathcal{D}_f)\cdot\mathcal{A}_\mathcal{L}(\mathcal{D}_\mathcal{L}\backslash\mathcal{D}_f) \in \mathcal{R}] \leq e^\epsilon \Pr[\mathcal{U}(\mathcal{A}_S(\mathcal{D}), \mathcal{D}_f, \mathcal{D})\cdot\mathcal{U}(\mathcal{A}_\mathcal{L}(\mathcal{D}_\mathcal{L}), \mathcal{D}_f, \mathcal{D}_\mathcal{L}) \in \mathcal{R}] + \delta \quad (3)$$

$$\Pr[\mathcal{U}(\mathcal{A}_S(\mathcal{D}), \mathcal{D}_f, \mathcal{D})\cdot\mathcal{U}(\mathcal{A}_\mathcal{L}(\mathcal{D}_\mathcal{L}), \mathcal{D}_f, \mathcal{D}_\mathcal{L}) \in \mathcal{R}] \leq e^\epsilon \Pr[\mathcal{A}_S(\mathcal{D}\backslash\mathcal{D}_f)\cdot\mathcal{A}_\mathcal{L}(\mathcal{D}_\mathcal{L}\backslash\mathcal{D}_f) \in \mathcal{R}] + \delta \quad (4)$$

Our hypothesis is that while Eq. (3) and Eq. (4) are equivalent to Eq. (1) and Eq. (2), Eq. (3) and Eq. (4) is more efficient to achieve.

3.3 Construction of HUF

HUF requires that classes are organized into superclasses in order to train the root and leaf classifiers. We present the algorithm (Algorithm 1), used to achieve this mapping without manual relabelling.

First, we obtain the confusion matrix C of the trained model, by comparing the model's predicted labels of the test dataset against their ground truth labels. We then apply the following transformation to obtain its similarity matrix, Z. We define ψ as a function that takes a square matrix as input, and yields its lower triangular matrix and sets its diagonals to 0. Further, *norm* is a function that row-normalizes the matrix. Then, $Z = \psi(1 - \text{norm}(\psi(C + C^T)))$.

In Fig. 2, we show an example of a similarity matrix generated for the CIFAR-10 dataset. Smaller values represent classes that are more similar. Using this similarity matrix, we apply agglomerative hierarchical clustering [21] to obtain the superclass to class mapping (ϕ) - each cluster represents a superclass. We choose agglomerative clustering since it is a bottom-up clustering approach that efficiently scales with the number of clusters (*i.e.* superclasses) and samples (*i.e.* classes) [20]. We set the number of clusters to $\sqrt{m}$ where m is the number of classes in the dataset.

Finally, based on the superclasses discovered through clustering, we retrain the root and leaf classifier models.

Algorithm 1 Construction of HUF

```
procedure ConstructionHUF($\mathcal{A}$, $\mathcal{D}$, $M$)
    Evaluate $\mathcal{A}$ against test dataset and retrieve confusion matrix ($C$)
    Apply transformation: $Z = \psi(1 - \text{norm}(\psi(C + C^T)))$
    Agglomerative Clustering on $Z$; Obtain $\phi : M \to S$
    $\mathcal{A}_S \leftarrow$ Train the Root Classifier
    for Superclass $s$ in $S$ do
        Training Subset $\mathcal{D}_S$ = { }
        for Class $m$ in $M$ do
            if $\phi(m) = s$ then
                $\mathcal{D}_S \leftarrow$ Append($\mathcal{D}$ where label belongs to class $m$)
            end if
        end for
        $\mathcal{A}_\mathcal{L} \leftarrow$ Train Leaf Classifier on $\mathcal{D}_S$
    end for
end procedure
```

3.4 Handling Forget Requests

We utilize random labelling [11, 18] to perform MU, where each sample in $\mathcal{D}_f$ is relabelled to a randomly chosen label class. Random labelling has the most simple implementation paired with high MU efficacy [7]. Each unlearning epoch consists of training the model on the randomly relabelled $\mathcal{D}_f$, followed by fine-tuning again on $\mathcal{D}_r$ in order to prevent catastrophic forgetting [14].

HUF performs preprocessing to determine the relevant classifiers for each forget sample in $\mathcal{D}_f$, by mapping the sample's label class to its superclass according to the mapping, ϕ. While MU is always performed at the root classifier, only data samples belonging to the relevant leaf classifier are unlearned. Since the classifiers are independent, MU at each classifier can be performed in parallel.

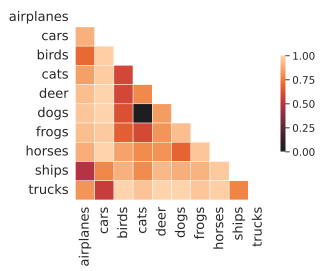


Figure 2: Similarity matrix (Z) generated for the CIFAR-10 dataset.

4 Evaluation

4.1 Experimental Setup

Datasets We use the CIFAR-10 and CIFAR-100 datasets [16], which are both popular multi-class object recognition datasets. While CIFAR-10 and CIFAR-100 have 10 and 100 classes respectively, both datasets comprise of 50000 training images, balanced among classes. Additionally, both datasets have classes with similar features (*i.e.* cats and dogs) that enable them to be hierarchically clustered.

Network Models We use the ResNet-18 [12] architecture for both the root and leaf classifiers as it offers high classification accuracy on images while being relatively efficient at both training and inference due to its residual layers.

Metrics Suppose $\mathcal{D}$ is the training dataset. Let $\mathcal{D}_f$ be the forget dataset, containing the input data that is to be unlearned. Let $\mathcal{U}$ represent an unlearned model. Let $\text{Acc}_d()$ represent the accuracy of an unlearned model, evaluated against some dataset d. Let $\mathcal{D}_r$ be the retain dataset, where $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$.

We measure the unlearning accuracy (**UA**), retain accuracy (**RA**), test accuracy (**TA**) and membership inference attack efficacy (**MIA**) of HUF, which are all established MU evaluation metrics [6, 13]. The metrics are defined as follows:

- **UA** measures the complement of the model's accuracy on the forget dataset. $\text{UA} = 1 - \text{Acc}_{\mathcal{D}_f}(\mathcal{U})$.
- **RA** measures the model's accuracy on the remainder of the training dataset. $\text{RA} = \text{Acc}_{\mathcal{D}_r}(\mathcal{U})$.
- **TA** measures the test accuracy of $\mathcal{U}$ against the test dataset.
- **MIA**, based on differential privacy, measures the MU efficacy by evaluating the proportion of $\mathcal{D}_f$ not predicted as members of the retain dataset, $\mathcal{D}_r$.

While UA and MIA directly measures the efficacy of unlearning, RA and TA measures the functionality of deploying the unlearned model for inference. All metrics range from 0 to 1.

Training Configuration We construct and train HUF for CIFAR-10 and CIFAR-100 as described in Section 3.3. CIFAR-10 and CIFAR-100 are organized into 2 and 10 superclasses respectively at the root classifier.

All of the networks are trained using stochastic gradient descent for 150 epochs. The learning rate is set to 0.1, and is decayed by 1/10, every 30 epochs.

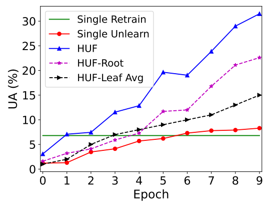
Unlearning Configuration We evaluate HUF under two MU configurations: random and class-wise unlearning. In random unlearning, $\mathcal{D}_f$ is constructed with a random 10% selection of the training data. Random unlearning represents the case where the ML model must handle a set of uncoordinated MU requests from users. In class-wise unlearning, $\mathcal{D}_f$ is populated with all training samples belonging to a randomly chosen class. Class-wise unlearning represents cases where an entire class is removed due to bias or changed requirements.

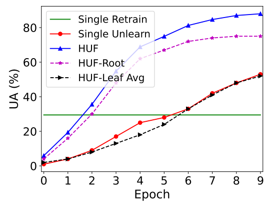
Under each MU configuration, we compare the MU efficacy of HUF to a single non-hierarchical classifier model, which also uses ResNet-18. We use the *single model retrain* and the *single model unlearning* as our baselines. Under single model retrain, the model is trained from scratch, with only

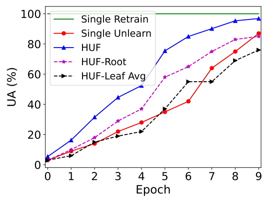
$\mathcal{D}_r$, for 150 epochs. Under single model unlearn, the model is trained with the $\mathcal{D}_f$ that is randomly relabelled, and followed by fine-tuning on $\mathcal{D}_r$.

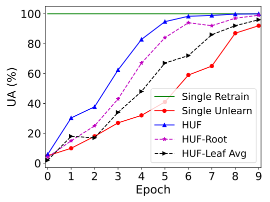
We set the unlearning rate to 0.01, without decay.

4.2 Experimental Results



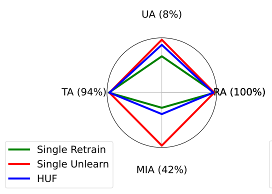


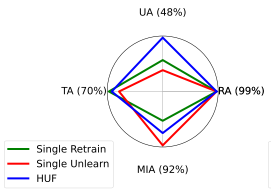


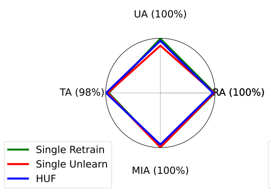


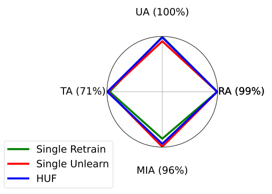
(a) CIFAR-10, Random (b) CIFAR-100, Random (c) CIFAR-10, Class-wise (d) CIFAR-100, Class-wise

Figure 3: UA of HUF after a given unlearning epoch. Single model retrain, single model unlearn, HUF root and leaves for comparison. Values closest to single model retrain are best.









(a) CIFAR-10, Random (b) CIFAR-100, Random (c) CIFAR-10, Class-wise (d) CIFAR-100, Class-wise

Figure 4: MU efficacy of HUF measured with UA, RA, MIA, TA. Single model retrain, single model unlearn, HUF root and leaves for comparison. Values closest to single model retrain are best.

Fig. 3 shows the UA of each MU technique at each epoch of unlearning. An ideal MU algorithm unlearns just enough to acquire a UA that approximates a single model retrained from scratch without $\mathcal{D}_f$ - we call this the *optimal UA* and overlay it as a flat line across epochs. If an MU algorithm's UA is greater than the optimal UA, it can over-forget, and result in a lower TA.

To measure the time taken for MU, which represents its efficiency, we count the number of epochs over the training data. For HUF, which has separate classifiers for the root and leaves, we show the UA separately in the graph. We also measure the overall UA of HUF, after performing N epochs on the root and leaf classifiers.

Models unlearn faster with fewer classes. For random unlearning, HUF takes 2 epochs to reach the optimal UA, compared to 5 epochs for single model unlearning on both CIFAR-10 (Fig. 3a) and CIFAR-100 (Fig. 3b). Similarly, for class-wise unlearning, HUF takes fewer epochs than single model unlearn to reach the optimal UA: 1 fewer epoch for CIFAR-10 (Fig. 3c) and 5 fewer epochs for CIFAR-100 (Fig. 3d).

We analyze HUF in finer detail by showing the UA of its root classifier and the average UA across its leaf classifiers, where each classifier is trained on fewer classes than the original single model. Additionally, the root classifier is always trained on fewer classes than its leaves. We observe the root classifier always unlearns faster than the leaf classifier, which in turn is usually faster than single model unlearning.

HUF preserves functionality after unlearning. We stop each MU algorithm when it approaches the optimal UA, and measure their RA, MIA, TA. An ideal MU algorithm performs the closest to single model retrain for each metric. As shown in Figs. 4a and 4b for random unlearning, HUF exhibits similarly high RA compared with single model unlearn. However, the MIA and TA of HUF track closer to single model retrain than single model unlearn, despite requiring fewer epochs to unlearn. Most evidently for CIFAR-100, shown in Fig. 4b, single model unlearn has a much greater MIA than HUF, while having a lower TA. When there are more classes available for training samples to be relabelled, MIA's success rate is lower against the forget samples. Coincidentally, the TA is also reduced since there is a greater likelihood for overfitting to occur on the forget samples during unlearning. For class-wise unlearning, there is minimal difference in functionality between HUF and single model unlearning, as seen in Figs. 4c and 4d.

Models unlearn more effectively when weights are shifted to unrelated classes. We observe the root classifier, which is trained on superclasses only, consistently unlearns faster than the single model retrain, which is trained on all classes. When applying random labelling to $\mathcal{D}_f$ of the root classifier, the samples are relabelled to a different superclass. Because each superclass is generated by clustering the classes, a different superclass incurs greater dissimilarity than regular classes. This helps to maximize the loss function computed on the forget samples from their original target labels.

5 Conclusions and Future Work

Machine unlearning (MU) is the task of forgetting the influence of data points from trained ML models, which has gained traction due to data privacy and bias concerns. Existing MU methods do not scale well in datasets where the number of classes are large. We propose HUF, a hierarchical unlearning framework, which aims to improve the efficiency of handling MU requests for large multi-class datasets.

In the future, we will explore the impact of incorporating different MU algorithms, additional hierarchical layers to HUF, as well as, weight sharing among classifiers for further improvements to MU efficiency. Another area of exploration is to reduce the size of the retain dataset. HUF currently expects access to the original training dataset in full, by requiring the retain dataset during MU. Instead of training the root classifier on the entire dataset, a subset of training data could be used instead. If the training data used for the root and leave classifiers are non-overlapping, this reduces the size of the retain dataset at each individual classifier. Thus, HUF provides a framework to support further MU optimizations.

Acknowledgments and Disclosure of Funding

We thank the FITML'24 reviewers for their valuable comments. This work was funded in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and a Four Year Fellowship from UBC.

References

[1] S. S. Banerjee, S. Jha, J. Cyriac, Z. T. Kalbarczyk, and R. K. Iyer. Hands Off the Wheel in Autonomous Vehicles?: A Systems Perspective on over a Million Miles of Field Data. In *Proceedings of IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2018.

[2] Lucas Bourtoule et al. Machine Unlearning. In *IEEE Symposium on Security and Privacy (SP)*, 2021.

[3] Boli Chen, Xin Huang, Lin Xiao, Zixin Cai, and Liping Jing. Hyperbolic Interaction Model For Hierarchical Multi-Label Classification. In *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)*, 2020.

[4] Ruizhe Chen et al. Fast Model Debias with Machine Unlearning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[5] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, 2006.

[6] Chongyu Fan, Jiancheng Liu, Alfred Hero, and Sijia Liu. Challenging Forgets: Unveiling the Worst-Case Forget Sets in Machine Unlearning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[7] Chongyu Fan, Jiancheng Liu, Yihua Zhang, Dennis Wei, Eric Wong, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

[8] Dehong Gao. Deep Hierarchical Classification for Category Prediction in E-commerce System. In *Proceedings of the 3rd Workshop on e-Commerce and NLP*, 2020.

[9] Shantanu Godbole. Exploiting Confusion Matrices for Automatic Generation of Topic Hierarchies and Scaling Up Multi-Way Classifiers. *Indian Institute of Technology–Bombay*, 2002.

[10] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal Sunshine of the Spotless Net: Selective Forgetting in Deep Networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] Tomohiro Hayase, Suguru Yasutomi, and Takashi Katoh. Selective Forgetting of Deep Networks at a Finer Level than Samples, 2020. arXiv:2012.11849.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[13] Jinghan Jia et al. Model Sparsity Can Simplify Machine Unlearning. In *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[14] James Kirkpatrick et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 2017.

[15] Sangamesh Kodge, Gobinda Saha, and Kaushik Roy. Deep Unlearning: Fast and Efficient Gradient-free Approach to Class Forgetting. 2024.

[16] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[17] Young-Woo Lee and Heung-Seok Chae. Identification of untrained class data using neuron clusters. *Neural Computing and Applications*, 2023.

[18] Junde Li and Swaroop Ghosh. Random Relabeling for Efficient Machine Unlearning, 2023. arXiv:2305.12320.

[19] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[20] Nicholas Monath et al. Scalable hierarchical agglomerative clustering. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD)*, 2021.

[21] Fionn Murtagh and Pierre Legendre. Ward's Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward's Criterion? *Journal of Classification*, 2014.

[22] Deboleena Roy, Priyadarshini Panda, and Kaushik Roy. Tree-CNN: A Hierarchical Deep Convolutional Neural Network for Incremental Learning. 2019.

[23] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember What You Want to Forget: Algorithms for Machine Unlearning, 2021. arXiv:2103.03279.

[24] Prateek Singhal et al. A Survey: Approaches to Facial Detection and Recognition with Machine Learning Techniques. In *Proceedings of Doctoral Symposium on Computational Intelligence (DoSCI)*, 2021.

[25] Matthias Springstein et al. Visual Narratives: Large-scale Hierarchical Classification of Art-historical Images. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024.

[26] Ayush K Tarun, Vikram S Chundawat, Murari Mandal, and Mohan S. Kankanhalli. Fast Yet Effective Machine Unlearning. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[27] European Union. General Data Protection Regulation. `https://gdpr-info.eu/`, 2018.

[28] Alice Xiang and Inioluwa Raji. On the Legal Compatibility of Fairness Definitions, 2019.

[29] Jie Xu, Zihan Wu, Cong Wang, and Xiaohua Jia. Machine Unlearning: Solutions and Challenges. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2023.

[30] Xiangxin Zhu, Carl Vondrick, Deva Ramanan, and Charless Fowlkes. Do We Need More Training Data or Better Models for Object Detection? In *Proceedings of the British Machine Vision Conference (BMVC)*, 2012.