Machine Unlearning

Variational Bayesian Unlearning Using Shards

CS772A: Probabilistic Machine Learning

Himanshu Mittal, Paturi Bhavya, Nikhil Mishra, Yash Sinha, Somya Gupta

Problem Statement

- Regulations like GDPR specify a “right to be forgotten” which requires machine learning model providers to enable mechanisms that allow for deletion of data and its contributions to the learning process. This can be achieved using **Machine Unlearning** models.

- Bayes’ rule can be used to cast approximate unlearning of a small subset of the training data as an inference problem. However we face a challenge of not having access to the exact posterior of the model parameters.

- Existing unlearning methods, especially probabilistic ones, can be slow. We explore ensembles to achieve faster unlearning while maintaining accuracy.

Literature Review

To address the problem of Machine unlearning we have gone through the following research works.

1. Deep Ensembles from a Bayesian Perspective discusses deep ensemble methods to address the epistemic uncertainty in **approximating Bayesian Models**.
2. **Variational Bayesian Unlearning** empirically demonstrates unlearning methods on Bayesian models such as **Sparse Gaussian Process** and logistic regression using synthetic and real-world datasets through **VI Framework**.
3. Challenges and Pitfalls of Bayesian Unlearning evaluates the effectiveness of **Laplace Bayesian Unlearning** and **Variational Bayesian Unlearning** methods for undoing learned information and discusses the drawbacks of both approaches.
4. **Machine Unlearning** introduces **SISA training**, a framework that expedites the unlearning process by strategically limiting the influence of a data point in the training procedure.

Variational Bayesian Unlearning

Minimizing Adjusted Evidence Upper Bound (EUBO)

Objective function has limitations- underestimates the variance of the true posterior and struggles to optimize for regions where variational dist q(θ|D) is low.

The paper proposes an adjusted likelihood approach to tackle this unlearning issue.

$$p_{\text{adj}}(\mathcal{D}_e|\boldsymbol{\theta};\lambda) \triangleq \begin{cases} p(\mathcal{D}_e|\boldsymbol{\theta}) & \text{if } q(\boldsymbol{\theta}|\mathcal{D}) > \lambda \max_{\boldsymbol{\theta}'} q(\boldsymbol{\theta}'|\mathcal{D}) \\ 1 & \text{otherwise} \end{cases}$$

$$\widetilde{\mathcal{U}}_{\text{adj}}(\lambda) \triangleq \int \tilde{q}_u(\boldsymbol{\theta}|\mathcal{D}_r;\lambda)\ \log p_{\text{adj}}(\mathcal{D}_e|\boldsymbol{\theta};\lambda)\ \mathrm{d}\boldsymbol{\theta} + \mathrm{KL}[\tilde{q}_u(\boldsymbol{\theta}|\mathcal{D}_r;\lambda)\ \|\ q(\boldsymbol{\theta}|\mathcal{D})]$$

λ denotes the threshold for unlearning (λ = 1 means no unlearning and 0 indicates full unlearning)

However in practice when we visualize the effect on the unlearned EUBO distribution as λ tends to zero we see that $\tilde{q}_u(\boldsymbol{\theta}|\mathcal{D}_r;\lambda)$ moves away from q(θ|D_r).

Reverse KL Method (rKL)

The paper minimizes reverse KL divergence w.r.t. the approximate posterior belief recovered by directly unlearning from erased data D_e

$$\mathrm{KL}[\tilde{p}(\boldsymbol{\theta}|\mathcal{D}_r)\ \|\ \tilde{q}_v(\boldsymbol{\theta}|\mathcal{D}_r)] = C_0 - C_1\ \mathbb{E}_{q(\boldsymbol{\theta}|\mathcal{D})}\left[(\log \tilde{q}_v(\boldsymbol{\theta}|\mathcal{D}_r))/p(\mathcal{D}_e|\boldsymbol{\theta})\right]$$

C_0 and C_1 are constants independent of $\tilde{q}_v(\boldsymbol{\theta}|\mathcal{D}_r)$

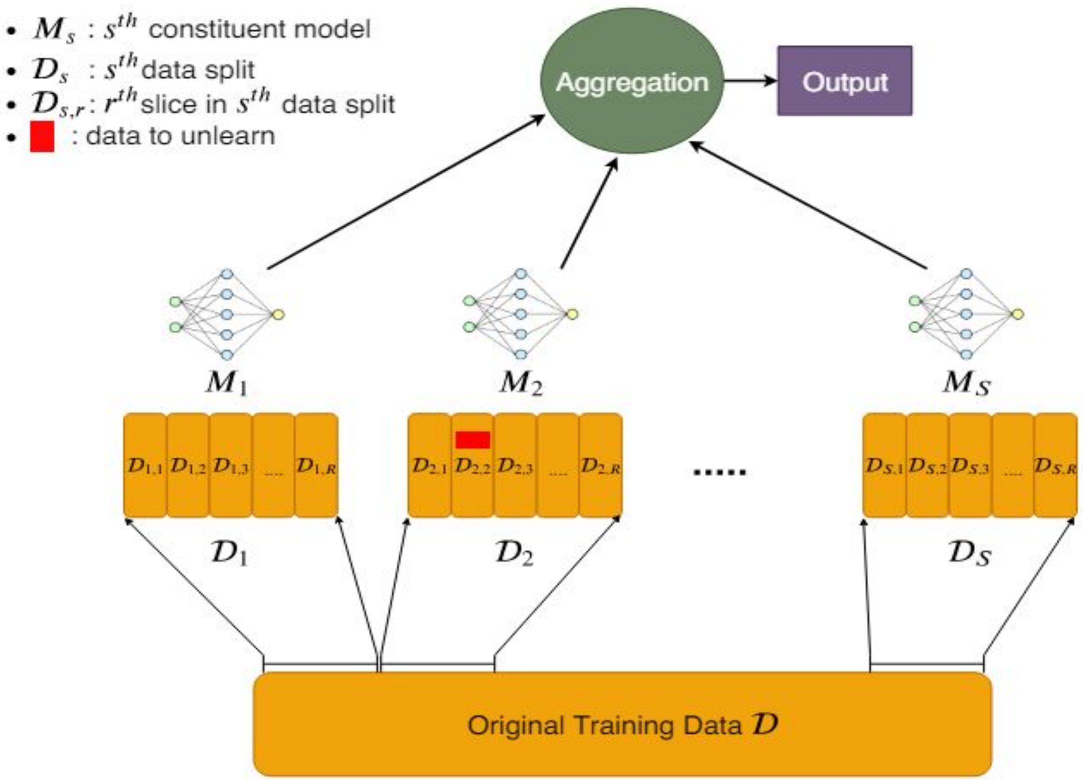
In contrast to the optimized $\tilde{q}_u(\boldsymbol{\theta}|\mathcal{D}_r;\lambda)$ from minimizing EUBO, the optimized $\tilde{q}_v(\boldsymbol{\theta}|\mathcal{D}_r)$ from minimizing the reverse KL divergence overestimates variance of $\tilde{p}(\boldsymbol{\theta}|\mathcal{D}_r)$

SISA Training Approach

- SISA training approach was proposed in the paper “*Machine Unlearning, 2020”[4].*

- Sharded Isolated Sliced and Aggregated (SISA) training approach divides the data into multiple shards, trains individual weak models on each shard, stores them, and aggregates them to form the final model.

- When unlearning a specific data point, the corresponding shard is retrained and then aggregated into the main model, ensuring exact unlearning.



- M_s : s^{th} constituent model
- $\mathcal{D}_s$: s^{th} data split
- $\mathcal{D}_{s,r}$: r^{th} slice in s^{th} data split
- 🟥 : data to unlearn
Aggregation
Output
M_1
M_2
M_S
$\mathcal{D}_{1,1}$ $\mathcal{D}_{1,2}$ $\mathcal{D}_{1,3}$ .... $\mathcal{D}_{1,R}$
$\mathcal{D}_{2,1}$ $\mathcal{D}_{2,2}$ $\mathcal{D}_{2,3}$ .... $\mathcal{D}_{2,R}$
.....
$\mathcal{D}_{S,1}$ $\mathcal{D}_{S,2}$ $\mathcal{D}_{S,3}$ .... $\mathcal{D}_{S,R}$
$\mathcal{D}_1$
$\mathcal{D}_2$
$\mathcal{D}_S$
Original Training Data $\mathcal{D}$

Our Approach

- Integrating these two approaches could potentially combine the benefits of Bayesian methods from VBU with the efficiency of SISA’s exact model, leading to significant improvements.
- We partition the dataset D_e into smaller subsets known as shards, and subsequently apply unlearning techniques such as EUBO and rKL to each shard.
- This process yields posterior distributions for each unlearned shard.
- Through Distributed Learning, we combine these distributions and compute the KL divergence between this aggregated distribution and the distribution obtained by training the model from scratch with the remaining data.
- We analyze the outcomes of these techniques and explore the impact of varying the number of shards on predictions.

Trained Model

We used a binary classifier trained on the synthetic moon dataset.

- In this dataset, the probability of an input x belonging to the 'blue' class is determined by 1/(1 + exp(f_x)) where f_x is a latent function represented by a sparse Gaussian Process (GP).

- The GP is characterized by 20 inducing variables, and its approximate posterior beliefs are modeled as multivariate Gaussians with full covariance matrices prior of which can be defined by the widely-used squared exponential covariance function:

$$k_{\mathbf{x}\mathbf{x}'} \triangleq \sigma_f^2 \exp(-0.5\|\Lambda(\mathbf{x}-\mathbf{x}')\|_2^2)$$

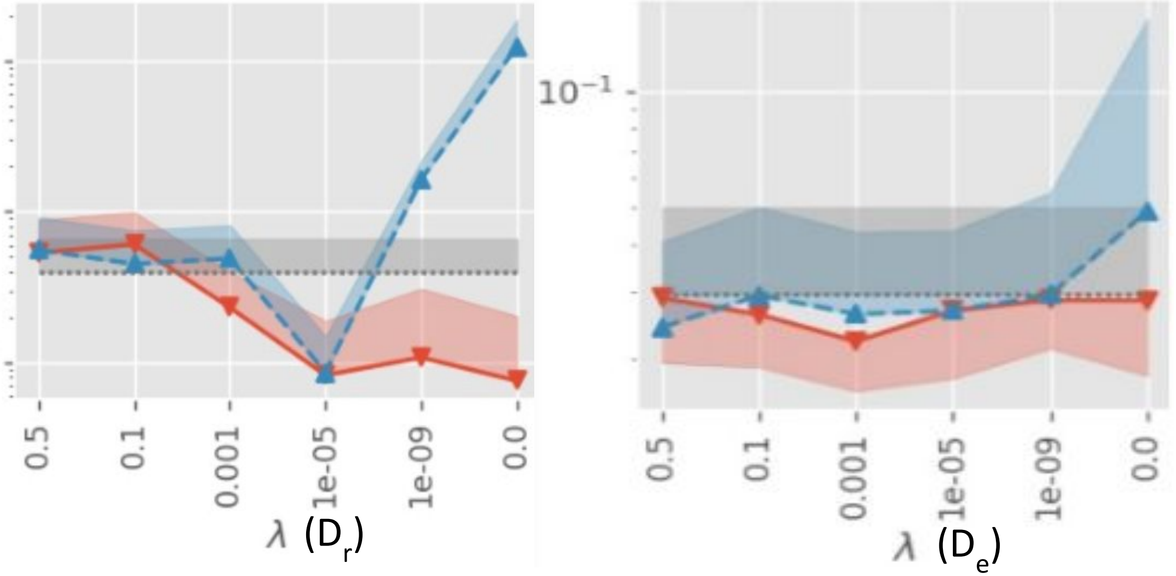
where $\boldsymbol{\Lambda} = \mathrm{diag}[\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2]$

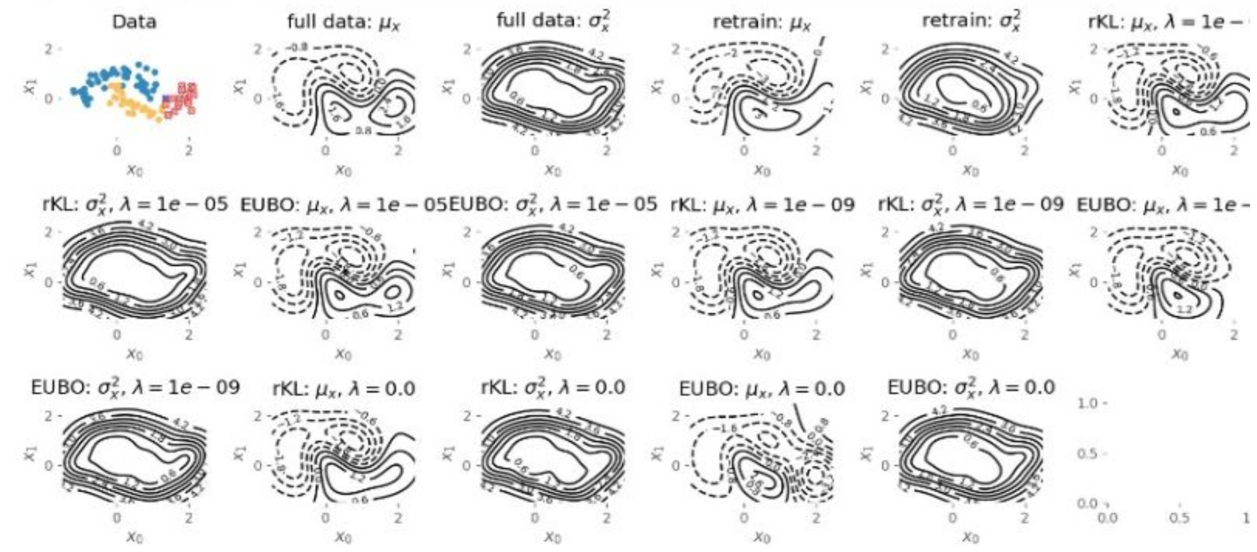
$\lambda_1 = 1.56$, $\lambda_2 = 1.35$, and $\sigma^2_f = 4.74$

- The posterior belief of the latent function value f_x at a new input x is a Gaussian:

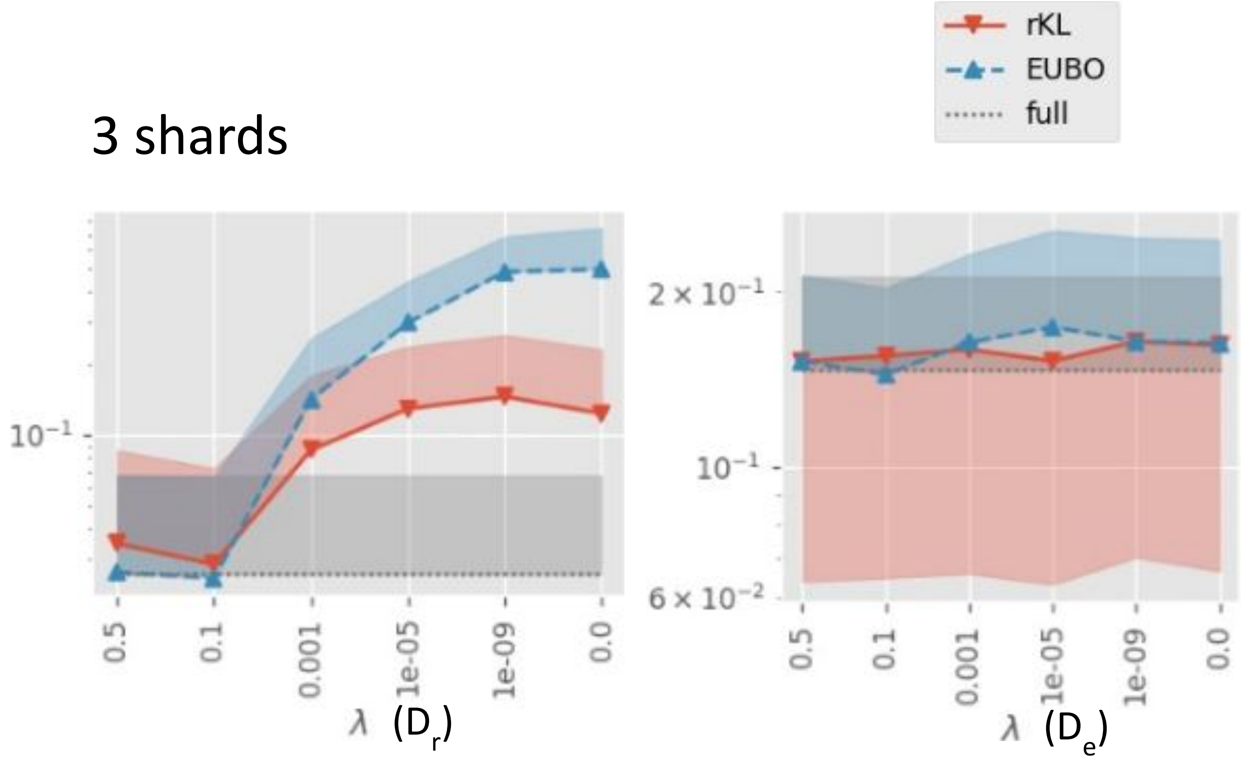
$$p(f_{\mathbf{x}}|\mathbf{f}_{\mathcal{X}_u}) = \mathcal{N}(\mathbf{k}_{\mathbf{x}\mathcal{X}_u}\mathbf{K}^{-1}_{\mathcal{X}_u\mathcal{X}_u}\mathbf{f}_{\mathcal{X}_u}, k_{\mathbf{x}\mathbf{x}} - \mathbf{k}_{\mathbf{x}\mathcal{X}_u}\mathbf{K}^{-1}_{\mathcal{X}_u\mathcal{X}_u}\mathbf{k}_{\mathcal{X}_u\mathbf{x}})$$

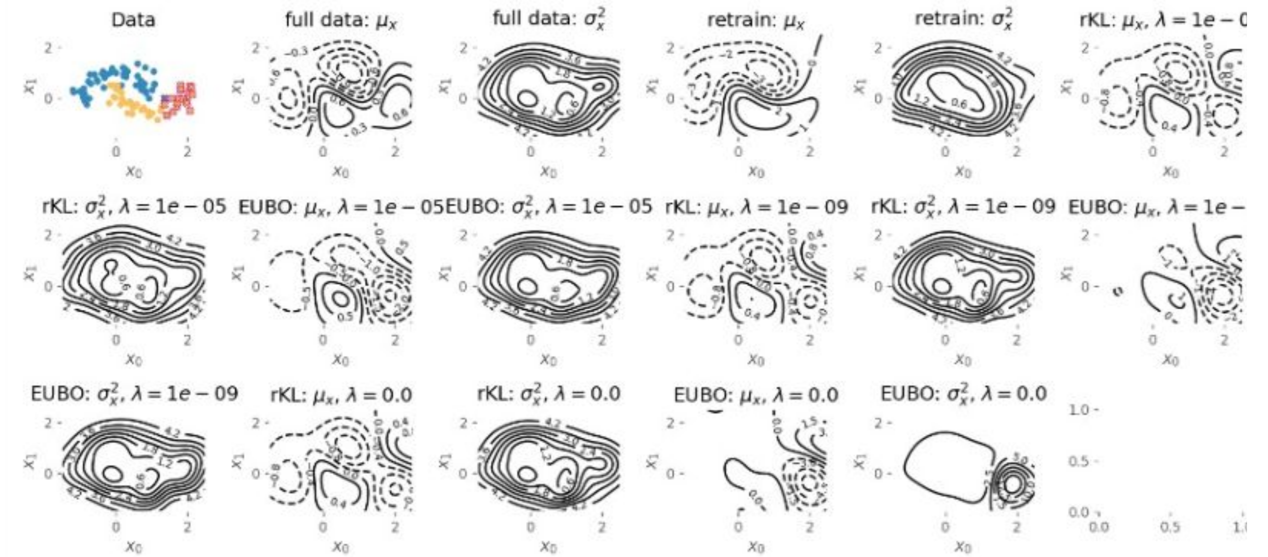
1 shard (Original Model)





3 shards

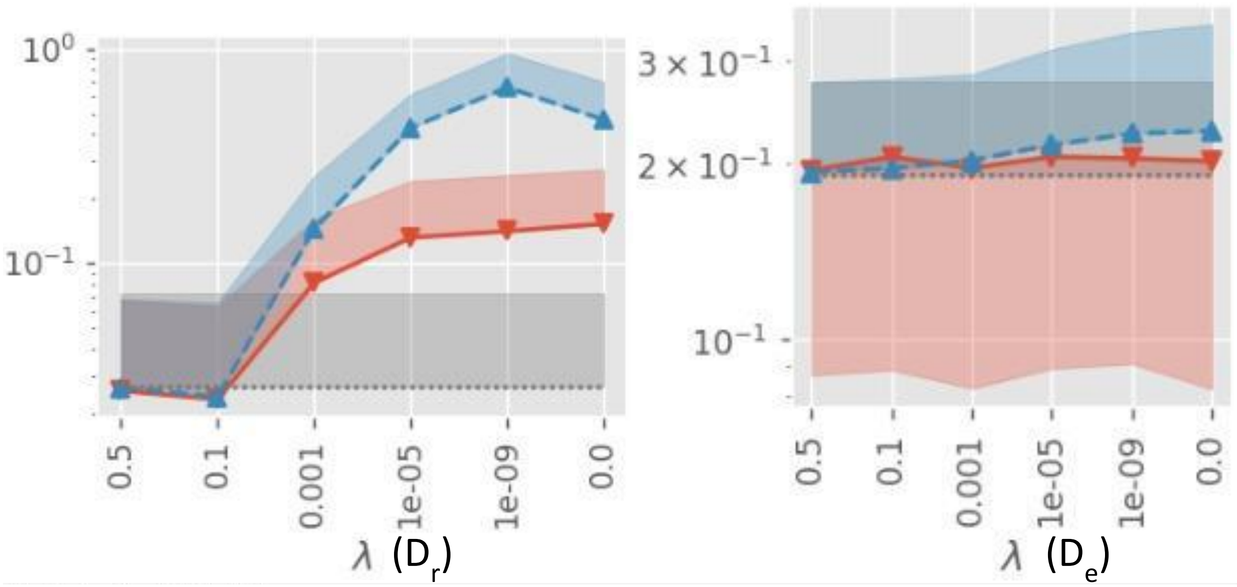


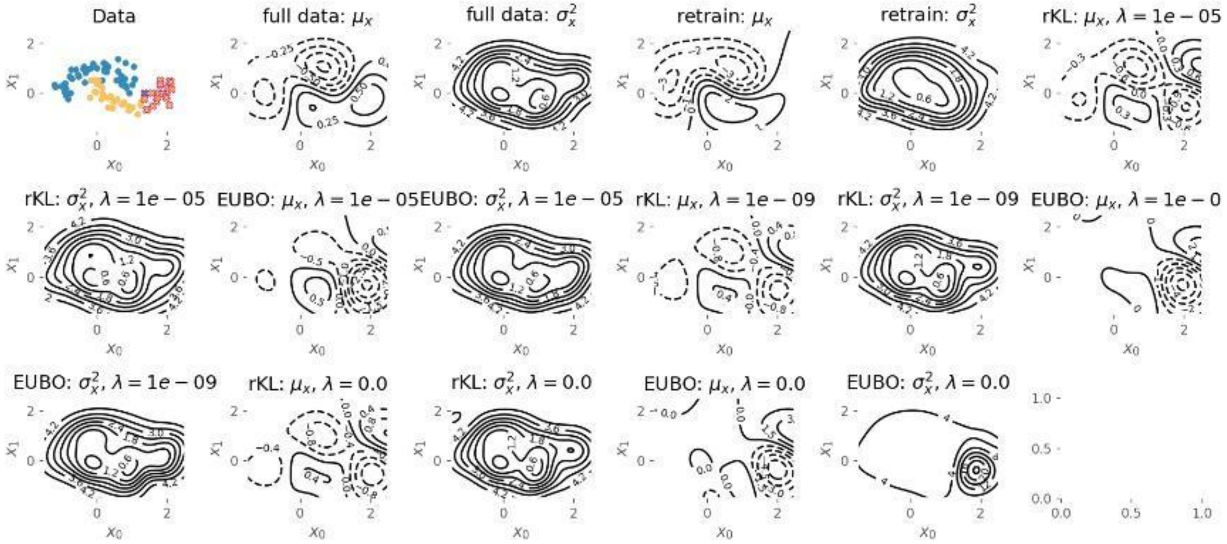


Results

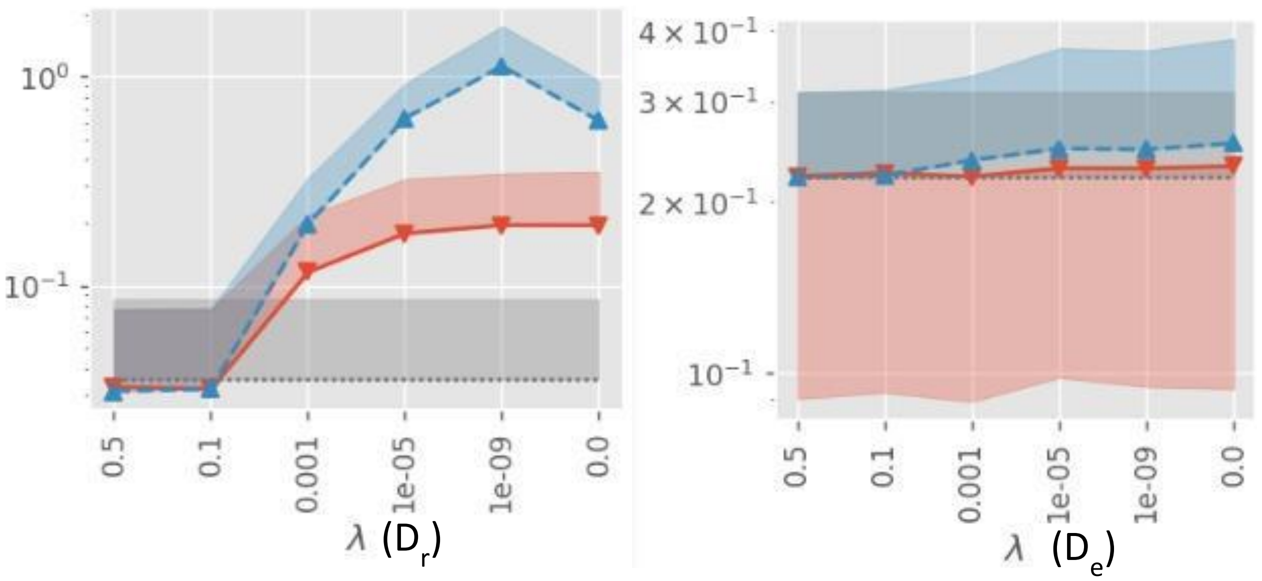
▼ rKL
▲ EUBO
full

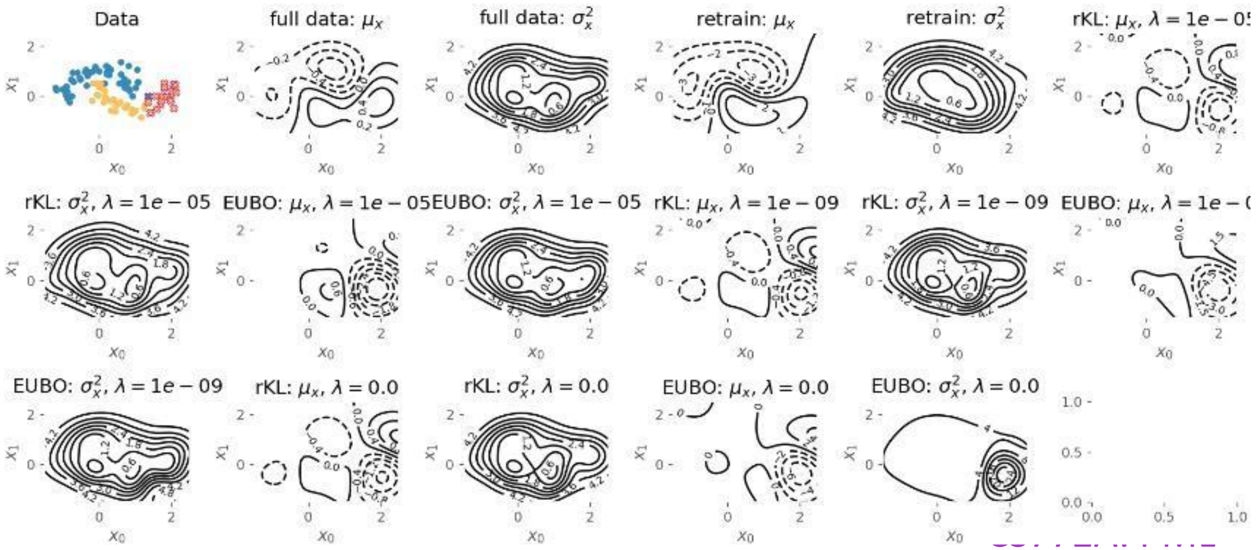
5 shards





10 shards





Observations

- The graphs above depict the variation of KL divergence between the unlearned model and the retrained model, with λ in different shards.
- Increasing the number of shards results in a significant escalation in the aggregated model's KL divergence with the retrained model.
- The analysis of the graphs reveals that as the number of shards increases, there is a corresponding rise in variance.
- Furthermore, examination of the contour plots illustrates that with an increase in shards, the contour of the yellow class becomes fragmented and distorted.
- Across all plots, the model's uncertainty or variance is significantly higher for erased data. This serves as compelling evidence of the occurrence of unlearning.

Conclusion

In this work, we implemented variational Bayesian unlearning (VBU) using the Shards framework. Our results show a trade-off between accuracy and number of shards. As the number of shards increases, the model's overall performance weakens, evident from the rising variance. This is because of information loss during aggregation from individual shards to the final model. Additionally, we observed that the effectiveness of both the rKL and EUBO methods hinges on the lambda parameter, which controls the degree of unlearning. Finding the optimal lambda value remains crucial for balancing unlearning and performance.

Thank you