Report date 04/02/2025

Paper report:

**Overview:**

The paper explains the concept of seeing machine unlearning as IVM on ML models .

The purposed of MUL is introduced, the need for low-latency MUL models is introduced too.

To have unlearnable models we must be able to represent the whole training procedure to relational operations on sets, for which we know how to do IVM on.

The first model being analyzed is a Naive Bayes classifier used to unlearn an email spam filter.

The first approach is modeling the algorithm as an algebraic view of the query used to get all the class probabilities (or counts, to then get the probability) to be used in the classifier. The two operators in the query are: selection (easy to unlearn) and equi-join.

The second unlearning implementation is done through Differential Dataflow, the Dataflow implementation is simpler.

The second model being unlearnt is a Vector-Session-kNN, in this case only the Differential Dataflow implementation is provided and its performances are compared with the non-unlearnable version.

The results show a 1ms increase in the unlearnt version performance (about 10% increase in runtime), showing the little impact and cost of having such an unlearnable model via Differential Dataflow.

Overall the paper concludes saying that implementing unlearnable ML models via Differential Dataflow is a promising technique. The authors are still investigating other algorithms (namely nearest neighbour and tree-based) to be efficiently modeled as a differential computations.

**Experimental metrics and scenarios:**

One experiment consists in unlearning a Naive Bayes classifier for spam emails. The task is performed both via relational algebra operations and via Differential Dataflow, no quantitative results are provided for this experiment.

The second experiments works on Sequential Recommendation with Vector Session kNN and only uses Differential Dataflow. The author compares the runtime for the standard recommendation vs the recommendation latency of the unlearnable model, before recommending from the unlearnable model an unlearning operation is performed by removing 10 historical sessions from the training data. The results show that the latency is increased by only 10% in the unlearnable model.

**Limitations:**

In order for such an approach to be viable in terms of runtime (shorter runtime than complete retraining), the models have to be sparse-computational dependencies. This means that each model training datapoint should affect only a limited part of the model, in this way deleting a datapoint would reflect changes in a small subset of the model values. If every change would reflect in a whole model recomputation then the benefit of IVM would be lost.

The dependencies of a model can be shown via a bipartite dependency graph: G(Vi, Vm, E), where Vi are the partitions of the input data, Vm are the partitions (or the parts) of the model, and E are the edges between inputs and model parts that represent which input data affects which model part.

Gradient based algorithms, such as NNs or LLMs, have very dense computational dependencies since a model is learnt by building on top of the gradient of the loss function, the argument of the loss function contains all the previous iterations of the model (since it contains the previous iteration which is build on top of its previous and so on). For this reason such intermediate model is dependent on all input with a gradient non-zero.

**Repo:**

**Authors:**

**BIFOLD**

Sebastian

**Year:**

2021