Machine Unlearning in 2024

Ken Ziyu Liu

Stanford Computer Science

May 2024

As our ML models today become larger and their (pre-)training sets grow to inscrutable sizes, people are increasingly interested in the concept of **machine unlearning** to edit away undesired things like private data, stale knowledge, copyrighted materials, toxic/unsafe content, dangerous capabilities, and misinformation, without retraining models from scratch.

Machine unlearning can be broadly described as removing the influences of training data from a trained model. At its core, unlearning on a *target model* seeks to produce an *unlearned model* that is equivalent to—or at least "behaves like"—a *retrained model* that is trained on the same data of target model, minus the information to be unlearned.

There's a lot hidden in the above description. How do we describe the information to be unlearned? Do we always have ground-truth retrained models? If not, how do we actually evaluate the unlearning? Can we even verify and audit the unlearning? Is *pretending* to unlearn, as humans often do, sufficient? Is unlearning even the right solution? If so, for what problems?

The precise definitions of unlearning, the techniques, the guarantees, and the metrics/evaluations would depend on:

1. The ML task (e.g., binary classification or language modeling);
2. The data to unlearn (e.g., a set of images, news articles, or the knowledge of making napalm);
3. The unlearning algorithm (e.g., heuristic fine-tuning vs deleting model components);
4. The goal of unlearning (e.g., for user privacy or harmfulness removal).

In this educational and position post, I hope to give a gentle, general ML audience introduction to machine unlearning and touch on things like copyright protection, New York Times v. OpenAI, right-to-be-forgotten, NeurIPS machine unlearning challenge, retrieval-based AI systems, AI safety, along with some of my thoughts on the field. While unlearning is broad topic applicable to most ML models, we will focus a lot on foundation models.

1. A bit of history & motivations for unlearning

People have thought about the unlearning problem for a while now. The initial research explorations were primarily driven by Article 17 of GDPR (European Union's privacy regulation), often referred to as "right-to-be-forgotten" (**RTBF**) since 2014. RTBF basically says a user has the right to request deletion of their data from a service provider (e.g. deleting your Gmail account).

RTBF was well-intentioned. It was also very actionable when said service providers store user data in a structured way, like how Google removed a bunch of links from its index in repsonse to RTBF requests.

However, RTBF wasn't really proposed with machine learning in mind. In 2014, policymakers wouldn't have predicted that deep learning will be a giant hodgepodge of data & compute, and that separating and interpreting this hodgepodge turned out to be hard. The hardness of erasing data from ML models has subsequently motivated research on what is later referred to as "data deletion" and "machine unlearning".

A decade later in 2024, user privacy is no longer the only motivation for unlearning. We've gone from training small convolutional nets on face images to training giant language models on pay-walled, copyrighted, toxic, dangerous, and otherwise harmful content, all of which we may want to "erase" from the ML models—sometimes with access to only a handful of examples. The nature of the models has changed too. Instead of using many small specialized models each good at one task, people started using a single giant model that knows just about any task.

Currently, I think the motivations for unlearning fall into two categories:

1. **Access revocation** (think unlearning private and copyrighted data). In an ideal world, data should be thought of as "borrowed" (possibly unpermittedly) and thus can be "returned", and unlearning should enable such revocation.

   Unlearning is challenging from this perspective. One key difficulty is that our limited understanding of deep learning itself makes data trained into a model akin to "consumables" (which can't just be "returned" after consumption). Data may also be non-fungible (e.g. your chat history) and may even be thought of as labor with its own financial and control interests. Another challenge is that access revocation may require a *proof* of unlearning; as we will explore in the coming sections, this isn't always possible.

   These difficulties suggest that it's perhaps also worth revising laws like RTBF and thinking about alternatives such as *data markets*, where data owners are properly compensated so they won't want to request unlearning in the first place. To illustrate the intuition, suppose Bob ate Alice's cheesecake (data), Alice would much rather Bob pay her or return something equivalent (compensation) than Bob puking to his pre-eating state (unlearning).[1]

[1]The cheesecake analogy is mostly meant as an illustration that there are convenient alternatives to unlearning as it pertains to access revocation. Indeed, there are cases where the analogy doesn't quite fully capture the nuances of the unlearning problem. For example, what

In practice, one way to implement access revocation is via some form of *periodic re-training* of the base model. Many model providers already do this to keep their models competitive and up-to-date. For example, OpenAI can collect a bunch of unlearning requests, and batch-satisfy them during the re-training every year (or, guided by RTBF's "undue delay" period by which the request must be satisfied). More broadly, this suggests *socio-technical solutions* for unlearning: policymakers can mandate such periodic re-training and set economically viable deadlines to offload the costs to the model owners.

2. **Model correction & editing** (think toxicity, bias, stale/dangerous knowledge removal). That is, the model was trained on something undesirable and we'd like to fix it. This is closely related to the model editing literature. The concept of "corrective machine unlearning", where unlearning serves to correct the impact of bad data, was recently proposed to capture this motivation. From this perspective, unlearning may also be viewed as a post-training risk mitigation mechanism for AI safety concerns (discussed further in Section 4).

   Unlike access revocation, we could be more lenient towards with model correction since the edit is more of a desire than a necessity mandated by law, much like model accuracy on image classification or toxicity of generated text. (Of course, these can cause real harm too.) Here, we won't necessarily need formal guarantees (despite wanting them) for the unlearning to be practically useful; we have plenty of examples where people would happily deploy models that are deemed "sufficiently safe". The recent WMDP benchmark, which quizzes a model on hazardous knowledge, is a good example of empirically evaluating unlearning efficacy.

2. Forms of unlearning

Unlearning is trivially satisfied if we can just retrain the model without the undesired data. However, we want something better because (1) retraining can be expensive and (2) it can be a lot of work *just to find out* what to remove from training data—think finding all Harry Potter references in a trillion tokens. Unlearning techniques essentially seek to mitigate or avoid this retraining cost while producing identical or similar results.

The unlearning literature can roughly be categorized into the following:

1. Exact unlearning
2. "Unlearning" via differential privacy
3. Empirical unlearning, where data to be unlearned are precisely known (training examples)

if Bob digested the cheesecake (data) and there is nothing to puke (data can't possibly be removed from a system without rebuilding it)? Also, Alice doesn't want the puked content (processed data) since it generally wouldn't be particularly useful, but what if it is under certain circumstances?

4. Empirical unlearning, where data to be unlearned are underspecified (think "knowledge")
5. Just *ask* for unlearning?

The non-exact unlearning methods are sometimes known as "**approximate unlearning**"[2] in that the unlearned model approximates the behavior of the retrained model. Form 5 is quite new and interesting, and more specific to instruction-following models.

The term "**empirical unlearning**" here broadly refers to procedures that start from the original trained model and apply some post-hoc processing that does not provide guarantees to the (pre-)training data (hence "empirical"). Much of the prior literature here can be thought of as model fine-tuning with some form of a gradient-based procedure, though more complex post-hoc methods are conceivable (e.g., second-order methods, distillation, weight selection). We'll mainly focus on gradient-based methods in this blog.

In the following, we will go through what each of these types roughly looks like, along with what I think are the promises, caveats, and questions to ask looking forward.

Figure 1. Illustration of approximate unlearning. Source: NeurIPS Machine Unlearning Challenge.

2.1. Exact unlearning

Exact unlearning roughly asks that the unlearned model and the retrained model to be *distributionally identical*; that is, they can be exactly the same under fixed randomness.

Techniques for exact unlearning are characterized by the early work of Cao & Yang and SISA. In SISA, a very simple scheme, the training set is split into N non-overlapping subsets, and a separate model is trained for each subset. Unlearning involves retraining the model corresponding to and without the data points to be unlearned. This reduces cost from vanilla retraining by $1/N$ (cheaper if we keep model checkpoints). Inference then involves model ensembling.[3]

Figure 2. Illustration of SISA: just train models on data shards (image source).

[2]The term "approximate unlearning" may also be used to exclusively refer to DP-like unlearning definitions (Section 2.2) appeared in prior literature. Here, I am using the term more in its literal sense as approximating the behavior of the unlearned model to (what we think is) the behavior of the retrained model.

[3]Technically, SISA may not give *exact* unlearning in the sense of identical model distributions between the retrained model and the unlearned model, since after a sequence of unlearning requests, the data shards may end up in a state that we wouldn't otherwise get into in the first place (e.g., some shards have way more data than others after unlearning). For practical purposes, nevertheless, this is subtle enough that the nice properties about exact unlearning, as discussed later in the section, would still hold.

More generally, the essence of exact unlearning of this form is that we want *modular components* in the learning algorithm to correspond to different (potentially disjoint) sets of the training examples.

There are several benefits of exact unlearning:

1. **The algorithm *is* the proof**. If we implement something like SISA, we know by design that the unlearned data never contributed to other components. As it turns out, formally proving the model has unlearned something is quite challenging otherwise.
2. **It turns the unlearning problem into an accuracy/efficiency problem.** This makes exact unlearning more approachable due to the messiness of unlearning evaluation and lack of benchmarks.
3. **Interpretability by design**. By providing a structure to learning, we also have better understanding of how certain data points contribute to performance.

The main drawback seems obvious: modern scaling law of large models argues against excessive data & model sharding as done in SISA. *Or does it?* I think it would be very interesting to revisit sharding in the context of large models, in light of the recent model merging literature that suggests the feasibility of weight-space merging between large models. As we'll learn in the coming sections, the messiness of approximate unlearning and its evaluation, especially in the context of large models, makes exact unlearning very appealing.

2.2. "Unlearning" via differential privacy

This line of work roughly says: if the model behaves more or less the same with or without any particular data point, then there's nothing we need to unlearn from that data point. More broadly, we are asking for *distributional closeness* between the unlearned and the retrained models.

For readers unfamilar with differential privacy (DP) in machine learning, DP defines a *quantifiable* indistinguishability guarantee between two models M, M' trained on datasets X, X' that differ in any single training example. The canonical procedure, DP-SGD, works by clipping the L2-norm of the per-example gradients and injecting some per-coordinate Gaussian noise to the gradients. The idea is that the noise would mask or obscure the contribution of any single gradient (example), such that the final model isn't sensitive to any exmaple. It is usually denoted by (ε, δ)-DP; the stronger the noise, the smaller the scalars (ε, δ), the more private.

The intuition is that if an adversary cannot (reliably) tell apart the models, then it is as if this data point has never been learned—thus no need to unlearn. DP can be used to achieve this form of unlearning, but due to the one-sidedness of unlearning (where we only care about data removal, not addition), DP is a strictly stronger definition. This notion of unlearning is sometimes known as "**(α, β)-unlearning**" where (α, β) serve similar roles as (ε, δ) to measure distributional closeness.

Example techniques along this direction include: (1) storing checkpoints of (DP) convex models and unlearning is retraining from those checkpoints; and (2) on top of the previous technique, add SISA for *adaptive* unlearning requests (i.e. those that come in after observing the published model).

DP-based unlearning is good in that it gives some form of a statistical guarantee. **However, there are some important considerations that limit its applicability to large models**:

1. Many such unlearning results apply only to convex models or losses.
2. What levels of unlearning (values of (ε, δ)-DP or (α, β)-unlearning) are sufficient? *Who decides?*
3. For large models, current ML systems don't fit well with the *per-example* workloads of DP-like procedures. The memory overhead will also be prohibitive.
4. Moreoever, like DP, the guarantees can fall off quickly with more unlearning requests (at best the rate of $O(\sqrt{k})$ with k requests following DP composition theorems).
5. DP-like definitions implicitly assume we care about all data points *equally*. But some examples are more likely to receive unlearning request, and some examples would not have contributed to the learning at all.
6. DP-like procedures may also just hurt model accuracy a lot, sometimes in an unfair way.

For large models in particular, it's also worth distinguishing the cases of **unlearning pre-training data** vs **unlearning fine-tuning data**. The latter is a lot more tractable; for example, we could indeed fine-tune large models with differential privacy but not so much with pre-training.

2.2.1. ***Forging*** and its implications on DP-like unlearning definitions

An unlearning procedure may sometimes require an external *audit*, meaning that we'd like to prove that the unlearning procedure has actually happened.

The main idea of "forging" is that there exists two distinct datasets that, when trained on, would produce *the same gradients and (thus) the same models*. This is true intuitively:

1. Think linear regression of points on a perfect line; removing any 1 point doesn't change the fitted line;
2. Think mini-batch GD, where replacing one example gradient with the sum of several "fake" gradients would give the same batch gradient.

Forging implies that DP-based approximate unlearning may not be auditable—that is, the unlearning service provider cannot formally prove that the forget set is really forgotten. In fact, if we only look at the model weights, even exact unlearning may not be auditable.

While one can brush this off as a theoretical result, it does mean that policymakers should think carefully about how a future version of "right-to-be-forgotten" (if any) should look like and whether similar policies are legally and technically enforceable.

Indeed, what qualifies as an "audit" could very well be definition and application dependent. If the auditor only cares that the unlearned model performs poorly on a specified set of inputs (say on a set of face images), then even empirical unlearning is "auditable" (see next section).

2.3. Empirical unlearning with known example space ("example unlearning")

This line of work is essentially "training to unlearn" or "unlearning via fine-tuning": just take a few more heuristically chosen gradient steps to shape the original model's behavior into *what we think* the retrained model would do (while also optionally resetting some parameters in the model). It may also be referred to as "example unlearning", since the training, retain, and forget sets are often clearly defined.

The **NeurIPS 2023 Machine Unlearning Challenge** collected many methods along this direction. The challenge roughly runs as follows:

- You are given a face image dataset with designated retain/forget example splits for the training set, a target model trained on everything, and a secret model trained only on the retain set.
- You are asked to design an unlearning algorithm that produces 512 unlearned models from the target model that "match" the secretly kept 512 retrained models. Models are produced over different seeds for distributional closeness measurements.
- The "match" or evaluation metric uses a DP-like output-space similarity: for each forget example, compute an "empirical ε" over the 512 unlearned models based on true/false positive rates of an adversary (also provided by the organizer), and aggregate across examples.
- All models are a small ConvNet.

To give an intuition about how well empirical unlearning is doing without fully explaining the metric: the ground-truth retrained model gets about ~0.19, the winning submission gets to ~0.12, and the baseline (simple gradient ascent on forget set) is ~0.06.[4]

So what do the winning ideas look like? Participants experimented techniques along the lines of the following:

1. Gradient ascent on the forget set;

[4]One criticism for the unlearning metric used in the NeurIPS unlearning challenge is that the 512 unlearned models are all produced from a *single* target model provided by the organizers, which can itself be sensitive to randomness despite the 512 seeds to smooth out the randomness in retrained and unlearned models.

2. Gradient descent on the retain set (and hope that catastrophic forgetting takes care of unlearning);
3. Gradient descent on the forget set, but with uniformly random labels (to "confuse" the model);
4. Minimize KL divergence on outputs between unlearned model and original model on the retain set (to regularize unlearned model performance on unrelated data);
5. Add some noise to the weights;
6. Re-initialize weights that had similar gradients on the retain set and forget sets, and finetune these weights on the retain set;
7. Prune weights by L1-norm and fine-tune on the retain set;
8. Reset first/last k layers and fine-tune on the retain set; and
9. Heuristic/arbitrary combinations of the above.

Prominent ingredients in many winning solutions include the addition of noise, weight re-initialization, KL regularization, and fine-tuning on the retain set.

Indeed, despite the heuristic nature of these approaches, these are what most empirical unlearning algorithms, especially those on large (language) models, are doing these days.

People explore empirical approaches because theoretical tools are usually impractical; for example, enforcing DP simply hurts accuracy and efficiency too much, even for the GPU rich. On the flip side, empirical methods are often fast and easy to implement, and their effects are often qualitatively visible.

Another key motivation for empirical unlearning is that *counterfactuals* are unclear, especially on LLMs. In deep learning, we often don't know how the retrained model would behave on unseen data. What should the LLM think who Biden is, if not a politician? Should image classifiers give uniformly random predictions for unlearned images? Do they generalize? Or are they confidently wrong? Any of these is possible and it can be up to the practitioner to decide. It also means that *behaviors that are equally plausible can lead to wildly different measurements* (e.g., KL divergence between output distributions of unlearned & retrained model), complicating theoretical guarantees.

2.4. Empirical unlearning with unknown example space ("concept/knowledge unlearning")

What if the train, retain, or forget sets are poorly specified or just not specified at all? Foundation models that train on internet-scale data may get requests to unlearn a "concept", a "fact", or a piece of "knowledge", all of which we cannot easily associate a set of examples. The terms "**model editing**", "**concept editing**", "**model surgery**", and "**knowledge unlearning**" are closely related to this notion of unlearning.[5]

[5]More broadly, "unlearning" falls under the umbrella of "model editing" in the sense that a deletion is also an edit. Similarly, one could argue that the concept of "continual learning" falls under the umbrella too, where an association (say an input/label pair, or a piece of

The underspecification of the unlearning requests means that we now have to deal with the notions of "**unlearning scope**" (or "editing scope") and "**entailment**". That is, unlearning requests may provide canonical examples to indicate what to unlearn, but the same information can manifest in the (pre-)training set in many different forms with many different downstream implications such that simply achieving unlearning on these examples—even *exactly*—would not suffice.

For example:

- The association "Biden is the US president" is dispersed throughout various forms of text from news articles, books, casual text messages, or this very blog post. Can we ever unlearn all occurrences? Moreover, does unlearning Joe Biden also entail unlearning the color of Biden's cat?
- Artists may request to unlearn art style by providing art samples, but they won't be able to collect everything they have on the internet and their adaptations.
- New York Times may request to unlearn news articles, but they cannot enumerate quotes and secondary transformations of these articles.

Such vagueness also suggests that **unlearning pre-training data from large models are perhaps necessarily empirical:** it is unlikely to derive formal guarantees if we can't clearly specify what to (and what not to) unlearn in the trillions of tokens and establish clear information boundaries between different entities. An interesting implication of achieving unlearning empirically is that the unlearning *itself* can be unlearned.

What does existing work do, then, with underspecified unlearning requests? Most techniques are more or less the same as before, except now we also need to find the examples to fine-tune on. For example, attempting to unlearn Harry Potter involves asking GPT-4 to come up with plausible alternative text completions (e.g. that Mr. Potter studies baking instead of magic); and attempting to unlearn harmful behavior involves collecting examples of hatespeech.

Another set of techniques involves training the desired behavior (or its opposite) into task/control vectors and harnessing the capability of large models to undergo weight-space merging or activation steering. The fundamental approach of the above is more or less the same, nevertheless—obtaining these edit vectors involves (heuristically) designing what gradients to take and what data on which to take them. One could also frame the unlearning problem as an *alignment* problem and applies the forget examples with a DPO-like objective.

factual association) is updated by deleting of an old association and creating a new, clearly specified association. One could imagine using continual learning to help achieve unlearning and vice versa.

2.5. Just *ask* for unlearning?

It turns out that powerful, instruction-following LLMs like GPT-4 are smart enough to *pretend to unlearn*. This means crafting prompts to induce a (sufficiently) safe behavior for the target unlearning application.

This is an interesting approach because no gradients are involved whatsoever (big plus from a systems perspective), and intuitively the end results could very well be as good as existing empirical unlearning techniques. Among different ways we could prompt, past work explored the following two directions.

Literally asking to pretend unlearning. We can ask in the system prompt to, say, pretend to not know who Harry Potter is. By design, this works best for common entities, facts, knowledge, or behaviors (e.g. the ability to utter like Trump) that are well-captured in the pre-training set, since the LLM needs to *know it well to pretend not knowing it well.* On the other hand, suppose now we'd like to unlearn the address of an obscure person; the pre-training set is so large that we suspect it's part of training data. We now face a variant of the Streisand effect: is it even worth asking the model to pretend unlearning by accurately describing it in-context, and subsequently risk leaking it in subsequent model responses?

Few-shot prompting or "in-context unlearning". Suppose we now have a clearly defined set of forget examples with corresponding labels. We can *flip* their labels and put them in the prompt, along with more retain examples with correct labels, with the intuition that the model would treat these falsely labelled forget examples as truths and act accordingly—much like one could jailbreak a model this way.[6] Indeed, this works best when the forget examples and the counterfactual labels are clearly defined and (somewhat) finite. It *may* work for factual associations (e.g. Paris is the captial of France) by enumerating a *lot* of examples, but unlikely to work for unlearning toxic behaviors (where space of possible outputs is much larger).

In a sense, these approaches are complementary as they work for different kinds of unlearning requests.

More broadly, one could imagine a *boxed* LLM system for unlearning through prompting, where:

1. Only the input and output interfaces are exposed (like ChatGPT);
2. Different instances of a powerful LLM are responsible for accurately mimicking different parts of a desired unlearning behavior (for example, one LLM instance specializes in general trivia-style QA while another handles sequence completions);
3. An orchestrator/router LLM decides which unlearning worker instance to call depending on the input; and

[6]There is also evidence that in-context demonstrations mostly serve to elicit a particular behavior and that the labels don't even matter that much. It's unclear yet how we could reconcile this finding with "in-context unlearning".

4. A composer/summarizer LLM that drafts the final output conforming to the desired unlearning behavior; it may also apply some output filtering.

Some readers may grumble about the heuristic nature of such prompting-based techniques; that there is no proof of unlearning whatsoever. We should keep in mind that fine-tuning based empirical unlearning, as most recent approaches do, is perhaps not fundamentally different. I think it ultimately comes down to the following questions:

1. Which of fine-tuning or prompting can **better steer model behavior**?
2. Which of them are **less susceptible to attacks** (exposing less surfaces and/or requiring more effort for an adversary to revert the unlearning)?

My intuition of our current models says that both questions point to fine-tuning based unlearning, but this is very much up for debate and can change as we get more powerful models and better defense mechanisms. For example, the recent notion of an instruction hierarchy may help make such as an LLM system less susceptible to malicious prompts.

It might be useful to note that humans don't really "unlearn" a piece of knowledge either.[7] In fact, by claiming to have unlearned something, we often have: (1) not only learned it well to be able to make the very claim that we have unlearned it, and (2) consciously decided that it's no longer useful / beneficial to apply this knowledge to our current world state. Who is to say that unlearning for LLMs should be any different?

3. Evaluating unlearning

Unlearning is messy for many reasons. But one of the biggest broken things about unlearning is evaluation. In general, we care about three aspects:

- **Efficiency**: How fast is the algorithm compared to re-training?
- **Model utility**: Do we harm performance on the retain data or orthogonal tasks?
- **Forgetting quality**: How much and how well are the "forget data" actually unlearned? Relatedly, how fast can we recover (re-learn) them?

Evaluating efficiency and model utility are (slightly) easier; we already measure them during training. The key challenge is in understanding the forgetting quality.[8]

If the forget examples are specified, this *feels* easy too. For example, unlearning a particular image class may intuitively mean getting a near-chance accuracy

[7]Humans do *forget* things though, which is different. The ML analogy might be "catastrophic forgetting"; humans similarly forget things under information overload.

[8]In particular, recall that for *exact unlearning*, understanding forgetting quality isn't strictly necessary because the algorithm would remove the forget data from the picture by construction (through retraining). Thus it may be acceptable even if the unlearned model does well on the forget set (as it could be a result of generalization from the retain set). We will focus the discussions of unlearning evaluation on approximate unlearning.

on the images in that class. An evaluation protocol may also measure accuracy (high on retain & test set, low on forget set) or the likelihood of the forget text sequences (lower the better). One could also perform membership inference attacks on the forget examples and decide that the unlearning is successful if the attack success drops below a certain threshold.

However, even when the forget examples are clearly defined, evaluating unlearning can be hard, and the above intuitive choices of metrics aren't necessarily principled or extensible to settings like knowledge unlearning in LLMs. Expecting the model to perform poorly on an unlearned example ignores *generalization*, as the forget examples could very well be an interpolation/duplicate of certain retain examples (and thus retain a high accuracy after unlearning).

More broadly, a key challenge of evaluating unlearning, due to the black-box nature of deep learning, is that the *counterfactual* of not ever seeing the forget data can technically be undefined, even when forget examples are clearly defined. We don't always have oracle models that have never seen the forget examples; e.g., LLMs that have never seen Wikipedia articles are unlikely. Many low-level metrics, such as those based on similarity to retraining, implicitly select such a counterfactual (say through the choice of the optimization algorithm), but other counterfactuals exist too.

Evaluating unlearning on LLMs had been more of an art than science. For example, to unlearn "Harry Potter" as an entity, people would visualize how the token probabilities would decay for Harry Potter related text—and some other folks would come along and show that the model can indeed still answer Harry Potter trivia questions. The key issue has been the *desperate* lack of datasets and benchmarks for unlearning evaluation.

Since 2024, nevertheless, the benchmarking crisis is getting better. There are two recent projects worth highlighting:

- TOFU: A benchmark focusing on unlearning individuals (specifically book authors). It involves asking GPT-4 to create fake author profiles, fine-tuning an LLM on them, and using the fine-tune as the unlearning target model and the original LLM as the oracle "retrained" model. It provides QA pairs on the generated fake authors to evaluate a model's knowledge of these authors before/after applying unlearning.
- WMDP: A benchmark focusing on unlearning dangerous knowledge, specifically on biosecurity, cybersecurity, and chemical security. It provides 4000+ multiple-choice questions to test a model's hazardous knowledge before/after applying unlearning. As part of the report the authors also propose an activation steering based empirical unlearning method.

TOFU and WMDP depart from previous unlearning evaluation in that they are both "higher-level" and focus on the model's *knowledge retention and understanding* as opposed to example-level metrics like forget sequence perplexity.

This is particularly relevant for LLMs as they are generally capabale of giving the same answer in many different ways that low-level metrics can't capture.

Looking forward, I think **application-oriented unlearning benchmarks and evaluations** like TOFU and WMDP, as opposed to low-level, example-based evaluation like that of the NeurIPS unlearning challenge, are more practical for evaluating foundation models, owing to the multi-tasking nature of these models and the disparate definitions of "unlearning success" for each of these tasks. This applies to both when forget examples are clearly defined and when they are not. Indeed, one might imagine separate benchmarks on unlearning personally identifiable information (PII), copyrighted content, speech toxicity, or even model backdoors. For example, for unlearning PII, we might care about exact token regurgitation, whereas for toxicity, the unlearning metric would be the score reported by a ToxiGen classifier.

4. Practice, pitfalls, and prospects of unlearning

Unlearning is a hard problem, especially in the context of foundation models. As we actively research to make unlearning work in practice, it helps to philosophize a bit on what unlearning really means and whether it is the right solution for our current problems.

4.1. The spectrum of unlearning hardness

Intuitively, unlearning infrequent textual occurrences in LLMs like car accidents in Palo Alto should be easier than unlearning frequent occurrences like "Biden is the US president", which is in turn easier than unlearning fundamental facts like "the sun rises every day".

This spectrum of *unlearning hardness* emerges because as a piece of knowledge becomes more fundamental, it will have more associations with other pieces of knowledge (e.g. as premises or corollaries) and an exponentially larger unlearning scope. In fact, a piece of knowledge can be so embedded in the model's implicit knowledge graph that it cannot be unlearned without *introducing contraditions* and harming the model's utility.[9]

This intuition implies that certain unlearning requests are much harder or simply unsatisfiable (any attempts are bound to have flaws). Indeed, humans have experiences that form the basis of their subsequent actions and world models; it is subjective, blurry, and philosophical as to what capacity can humans unlearn their formative past memories.

More broadly, the unlearning hardness problem applies to all kinds of models, and for reasons beyond embeddedness in a knowledge/entailment graph. Let's consider two more seemingly contradictory intuitions for unlearning hardness:

[9]Note that this "embeddedness" of a piece of data is related but distinct from whether the data is *in or out of distribution*, which should also affects how an unlearning algorithm should behave (e.g. unlearning a perfect inlier should be no-op for an ideal unlearning algorithm).

1. An example seen later in the training should be *easy to unlearn*, since the model would have moved only slightly in weight space (e.g. due to decayed learning rate) and one could either just revert gradients or revert to a previous checkpoint (if stored). In contrast, examples seen early gets "built on" by later examples (in the curriculum learning sense), making them harder to unlearn.
2. An example seen later should be *harder to unlearn*, since examples seen earlier are gradually (or catastrophically) forgotten over the course of training; this may be especially true for LLMs.

Failure to reconcile these intuition would suggest that the interplay across *memorization/forgetting*, *example importance* (in the sense of data selection and coresets), *learning hardness* (in the sense of prediction flips), and unlearning hardness is unclear.

Some interesting questions:

- Is there a qualitative/fundamental difference between unlearning "easy" data (e.g. a local news event) and "hard" data (e.g. cats have four legs)?
- If there is a spectrum of unlearning hardness, does there exist a threshold to tell apart what is "easy" and "hard", and thus what is unlearnable or shouldn't be unlearned? Does there exist, or can we train, such an oracle classifier? Can humans even tell?
- How does unlearning hardness relate to **influence functions** and **data attribution**? If a certain piece of knowledge (as it manifests in a model's output) can be attributed to a larger fraction of the training data, does it make it harder to unlearn?
- Can we benchmark how easy is it to unlearn something? What metrics are suitable to measure unlearning hardness? Low-level metrics such as accuracy and membership inference attack success rates are directly applicable if the forget examples are known, but recall from Section 3 that they may not be perfect.

4.2. Copyright protection

On the surface, unlearning seems to be a promising solution for copyright protection: if a model violates the copyright of some content, we could attempt to unlearn said content.[10] It is conceivable that to resolve copyright violations via unlearning, provable and exact unlearning is necessary (and possibly sufficient); on the other hand, approximate unlearning, without guarantees and with the possibility of being hacked, is certainly insufficient and likely unnecessary.

In practice, however, there is a lot more nuance due to the questionable effectiveness of current unlearning methods and the unclear legal landscape at the

[10]Of course, we must first verify that such content has been trained on by the model in the first place. We can be almost certain that contents like Wikipedia articles are trained on, but we are generally less sure about a random blogpost somewhere on the internet. This is basically the membership inference problem.

intersection of AI and copyright. Since I am no legal expert (and clearly none of this section constitutes legal advice), we will mostly focus on asking questions. The central question seems to be: **is unlearning the right solution for copyright protection?**

Recall that the fair use doctrine[11] permits limited use of copyrighted material contigent on four factors: (1) purpose and character of the use ("transformativeness"), (2) the nature of the copyrighted work, (3) amount and substantiality of the use, and (4) the effect on material's value. If the use of copyrighted content in a model qualifies as fair use, then unlearning such content from the model is unnecessary.

Suppose a model is trained on some copyrighted content and is risking copyright violation, as in New York Times v. OpenAI. Should OpenAI invest in (empirical) unlearning algorithms on ChatGPT? Or should they focus on the transformativeness axis of fair use and invest in deploying empirical *guardrails*, such as prompting, content moderation, and custom alignment to prevent the model from regurgitating training data? The latter seems to be what's being implemented in practice.

More broadly, there could also be economic solutions to copyright violation as alternatives to unlearning. For example, model owners may provide an exact unlearning service (e.g. via periodic retraining) while also offering to indemnify model users for copyright infringement in the mean time, as seen in the case of OpenAI's "Copyright Shield". People are also starting to explore how one may price copyrighted data using Shapley values. In general, it is unclear right now how much of a role (if any) unlearning will play for resolving copyright related issues. Exact unlearning (extending to retrieval-based systems, see next section) does hold promises since deletion is clean and provable, but it seems that *legally binding* auditing procedures/mechanisms need to be first in place.

4.3. Retrieval-based AI systems

An obvious alternative to unlearning is to not learn at all. One way this could manifest for an LLM is that we take all content from the pre-training set that may receive unlearning requests (e.g., New York Times articles) and put them to an external data/vector store. Any questions relating to them will then be RAG'ed during inference, and any unlearning requests can be trivially satisfied by removing the data from the database. Min et al. demonstrates that this approach can be competitive to (though not quite matching) the trained baseline in terms of final perplexity.

Retrieval-based solutions are promising because of the increasing capabilities of the base models to reason in-context. However, there are few considerations

[11] Fair use is a doctrine applicable specifically in the United States. The reader should refer to related doctrines in corresponding jurisdictions, such as fair dealings in Commonwealth countries.

before taking retrieval systems as the no-brainer solution to unlearning:

1. **Removing protected content from pre-training corpus can be a hard de-duplication problem.** Much like removing data contamination is hard, how can we be sure that paraphrases, quotations/citations, or other adaptations of the protected content are removed?
2. **What if the data to be unlearned can't be retrieved?** Today we fine-tune many things into a model that aren't documents or knowledge items; for example, it is unclear (yet) if things like as human preferences and desired behaviors (e.g. ability to write concisely) can be "retrieved" from a database.
3. **Dumping stuff in-context can open new attack surfaces.** Many RAG methods for LLMs work by putting related content in-context and ask the model to reason on them. Having the protected data in-context means they are now more susceptible to data extraction (simple prompting attacks may work just fine).
4. **Utility gap between retrieval and training.** While there is evidence that retrieval-based solutions can be competitive, there is no general consensus that retrieval alone can replace fine-tune workloads; indeed, they can be complementary. More broadly, what if the space of unlearnable data is too large such that if all of it goes to an external store, the base model wouldn't be as useful?

4.4. AI safety

As models become more capable and are granted agency, one concrete application domain for unlearning that is gaining traction is **AI safety**.

Roughly speaking, safety concerns stem from a model's *knowledge* (e.g., recipe of napalm), *behaviors* (e.g., exhibiting bias), and *capabilities* (e.g., hacking websites). Examining current AI systems and extrapolating forward, one may imagine the following examples to apply unlearning and improve AI safety:

- removing **hazardous knowledge**, as seen in the WMDP benchmark;
- removing **model poisons and backdoors**, where models respond to adversarially planted input triggers;
- removing **manipulative behaviors**, such as the ability to perform unethical persuasions or deception;
- removing **bias and toxicity**; or even
- removing **power-seeking tendencies**.

For safety-oriented applications, it is worth noting that unlearning should be treated as a post-training *risk mitigation and defense mechanism*, alongside existing tools like alignment fine-tuning and content filters. And as with any tool, we should view unlearning through its trade-offs in comparison to other tools in the toolbox (e.g., unlearning is more adaptive but more expensive than content filters), as opposed to brushing it off because of the potential lack of guarantees and efficacy.

Acknowledgements: The author would like to thank Aryaman Arora, Jiaao Chen, Irena Gao, John Hewitt, Shengyuan Hu, Peter Kairouz, Sanmi Koyejo, Xiang Lisa Li, Percy Liang, Eric Mitchell, Rylan Schaeffer, Yijia Shao, Chenglei Si, Pratiksha Thaker, Eleni Triantafillou, Xindi Wu for helpful discussions and feedback before and during the drafting of this post. Any errors and hot/bad takes are those of the author.

Minor Revision (May 13, 2024): Special thanks to Eleni Triantafillou for detailed feedback, especially on correcting technical descriptions for the NeurIPS unlearning challenge and on improving the clarity of the sections on empirical unlearning and evaluation. Also added some footnotes to help clarify some technical terms/descriptions.

Citation

If you find this post helpful, it can be cited as:

Liu, Ken Ziyu. (May 2024). Machine Unlearning in 2024. Ken Ziyu Liu - Stanford Computer Science. https://ai.stanford.edu/~kzliu/blog/unlearning.

Or

```
@misc{liu2024unlearning,
  title   = {Machine Unlearning in 2024},
  author  = {Liu, Ken Ziyu},
  journal = {Stanford Computer Science},
  year    = {2024},
  month   = {May},
  url     = {https://ai.stanford.edu/~kzliu/blog/unlearning},
}
```
