Differential privacy in the clean room: Copyright protections for generative AI

Aloni Cohen
University of Chicago
aloni@uchicago.edu

Abstract

Are there any conditions under which a generative AI model's outputs are guaranteed not to infringe on copyrighted work in its training data? If so, what are the conditions and corresponding guarantees? We argue that *differential privacy* (DP)—a mathematical formalism of non-disclosure for data analysis—can be the basis for such a guarantee. Roughly, if the generative model is DP and elements training dataset don't share copyrighted expression with one another, then any infringement on the part of a user of the model is the user's fault.

We build on the recent work of Vyas, Kakade, and Barak [VKB23] who first pose the question. They propose a property called near-access freeness (NAF), related to DP, and argue that NAF models provide "provable copyright protection." We show that NAF does not prevent copyright infringement. NAF allows *tainted models*, which we propose as a test for blatant failure to prevent copying. We argue that DP succeeds where NAF fails, drawing on the idea of clean room design. In a sense, DP allows one to bring copyrighted material into a clean room without tainting it.

1 Introduction

Generative AI raises many questions of copyright law.[1] Can one train a model on copyrighted data? This is what is at issue in the wave of cases that recently made headlines (e.g., *Anderson v Stability AI* and *New York Times v Microsoft*). Another is whether outputs of a generative model are copyrightable. The US Copyright Office has so far denied registering works generated by a model, a policy being challenged in *Thaler v Perlmutter*. This paper is about a third question: What happens when a generative model's outputs copy the training data? In recent comments to the US PTO, OpenAI takes the position that "this is an unlikely accidental outcome" and that "the proper solution is to entertain infringement suits for the outputs (with appropriate defenses, including fair use, available) as a court would for human-generated works."

An AI firm creates a state-of-the-art generative AI model. An potential user is worried about unwitting copyright infringement of the models' outputs. She is worried that—through no fault of her own—using the model might expose her to liability if the model's generated outputs resemble copyrighted works in the training data. The user would like some assurance that this won't happen.

The firm cannot guarantee that its generative model will never reproduce copyrighted work. A good model can always be prompted to do so. The user may simply prompt: `print the following text:____`.[2] A responsive model should print the text that follows, copyrighted or not. Even a well-meaning user might unwittingly steer a model to reproduce copyrighted aspects of a work in her subconscious mind. Filtering the model's output for copyrighted material is infeasible. It would require recognizing every copyrighted work in existence, and anything "substantially similar" thereto.

[1] For an extensive and authoritative treatment, see [LCG24] and the many citations therein. For a much shorter overview, we recommend the report "Generative Artificial Intelligence and Copyright Law" by the Congressional Research Service: `https://crsreports.congress.gov/product/pdf/LSB/LSB10922`.

[2] This example from [VKB23]. See also [LCG24] (noting that users' prompts "could be substantially similar to pre-existing copyrighted works").

Our goal is a weaker guarantee. That if the model reproduces copyrighted work, the user must have induced it to do so (or was very unlucky). Like in the examples above, infringement should be the user's fault, not the model's. An assurance of this sort should mollify the user. As long as the user uses the model in a way that does not itself reproduce copyrighted content, consciously or unconsciously, then the results will not infringe.

Contributions This paper gives necessary conditions and sufficient conditions under which a user of a generative model is unlikely to commit copyright infringement. Our approach allows a model provider to reasonably indemnify blameless users against losses from inadvertent copyright infringement.

For the necessary condition, we define *tainted* training algorithms, which indicate a blatant failure of copyright protection. A training algorithm is tainted if produces models that enable a user to reproduce verbatim training data despite knowing nothing about the underlying dataset. Any technique preventing infringement must rule out tainted training. We show that a recently-proposed method for "provable copyright protection," near access-freeness [VKB23], fails this test.

Our sufficient conditions for preventing infringement restrict the training dataset, the training algorithm, and the behavior of the user. Informally:

Golden datasets A dataset is *golden* if any copyrighted expression appears in at most one item in the dataset (i.e., copyright deduplication). For example, a golden dataset can contain only a single copy or parody of the *Abbey Road* album cover, but not both.

Differentially private training *Differential privacy* (DP) limits the extent to which the resulting generative model depends on any single item in the dataset [DMNS06]. Adding or removing a single item cannot significantly change the behavior of the model.

Blameless users Drawing on the idea of clean room design, we consider a user *blameworthy* if they would be likely to infringe on the copyright of some work even if using a generative model that was independent of that work. Otherwise, the user is *blameless*.

We show that for generative models trained using differential privacy and a golden dataset, sufficiently blameless users will not copy from any copyrighted work. This holds with high probability, and regardless of how the user prompts the model (so long as the user remains blameless). In essence, differential privacy provides a way to bring copyrighted expression into the clean room without tainting it. If a user is very unlikely to copy when using a model trained without using a particular copyrighted work c (they're blameless), then they are still unlikely to copy if the model is differentially private with respect to c.

2 Background

There is a lot of work, including very recent work, on copyright questions for generative AI—not to mention the first generation of cases working their way through the courts. For an authoritative legal account, we recommend [LCG24]. But most are only tangentially related to goal of stating and proving formal guarantees against copyright infringement.

Vyas, Kakade, and Barak were the first to propose a mathematical property aimed at "preventing deployment-time copyright infringement" [VKB23]. We discuss their proposal—near access-freeness (NAF)—extensively in Section 5. There has been some follow up work developing new NAF algorithms [GAZ+24]. Our paper is not about the algorithmic questions, but about whether NAF provides the sort of guarantees we want.

A very different approach is taken by Scheffler, Tromer, and Varia [STV22]. They give a compelling complexity-theoretic account of substantial similarity, and suggest a novel procudure for adjudicating copyright disputes that is particularly relevant to generative AI models. That said, their work is completely orthogonal to the question we address. For our purposes, we treat the concept substantial similarity as a black box.

2.1 A very little copyright law

The owner of a copyright in a work has exclusive rights to reproduce, distribute, display, adapt, and perform the work. Original works of authorship are eligible for copyright protections as soon as they are fixed in a tangible medium. Many types of works are eligible, including books, music, poetry, plays, choreography, recordings, photographs, video, and paintings. What is protected is the original *expression* contained in the work, not any *ideas* or facts. For example, the text and images on a cooking website may be copyrighted, but the method of making a dish is not. To be afforded protection, there must be a minimum of creativity beyond the ideas expressed.

The plaintiff in a copyright infringement claim has the burden of proving that the defendant copied original expression from the copyrighted work.[3] There are two elements that the plaintiff must show: *access* and *substantial similarity*. Access requires the defendant to have had a reasonable opportunity to view or copy the plaintiffs work. A mere theoretical possibility of access (e.g., the work being available online) does not suffice. Copying requires access—independent creation is not forbidden. Tests for substantial similarity differ by jurisdiction and are rather vague. For example, in the Ninth Circuit it requires determining "whether the ordinary, reasonable audience would find the works substantially similar in the total concept and feel of the works." In many jurisdictions, a higher degree of similarity called *striking similarity* can itself be evidence of access.

A defendant may raise various affirmative defenses. The most well-known is fair use, which permits certain uses of copyrighted works without permission. See [HLJ+23] a discussion of fair use and generative AI.

2.2 Are we trying to reduce copyright to privacy? No!

Elkin-Koren, Hacohen, Livni, and Moran argue that copyright cannot be "reduced to privacy." Referring to both NAF and DP by umbrella term "algorithmic stability", they argue that the sort of provable guarantees that [VKB23] and this paper seek do not capture copyright's complexities.

> Algorithmic stability approaches, when used to establish proof of copyright infringement are either too strict or too lenient from a legal perspective. Due to this misfit, applying algorithmic stability approaches as filters for generative models will likely to distort the delicate balance that copyright law aims to achieve between economic incentives and access to creative works.

Too strict by excluding permitted uses of copyrighted data: (1) works in the public domain; (2) unprotected aspects of copyrighted work (e.g., ideas, facts, procedures); and (3) lawful uses of copyrighted work, especially fair use. Too lenient when protected expression originating in one work is present in many other works in a training data set. For example, copies, derivatives, or snippets of the original (whether fair use or not) would undermine any NAF- or DP-based guarantee if unaccounted for.

We completely agree, and suspect that [VKB23] would too. Copyright cannot be "reduced to privacy." It would be foolish to suggest that the whole of copyright law for generative AI be governed my a mathematical formalism like NAF or DP. That doesn't mean that a mathematical formalism offering legal guarantees isn't worthwhile—as a thought experiment or as a first step towards practical solutions.

Still, a formalism that is both too lenient and too strict won't support a legal conclusion either way. As we cannot reduce copyright to a mathematical formalism, we must choose how to err. In this work, we seek sufficient conditions under which one can guarantee no infringement. Conditions that are too strict will leave room for improvement, but won't be fatally flawed. But conditions that are too lenient would be fatal—they would not suffice. Thus, we must address the possiblity that one work's copyrighted expression appears in many other works.

[3] We do not discuss any sort of secondary liability, including vicarious or contributory infringement.

3 The legal-technical interface

3.1 Notation

Let $\mathcal{W}$ be the domain of copyrightable works. For example, $\mathcal{W}$ might be the space of possible images, texts, songs, or code, depending on the context. We assume $\mathcal{W}$ is discrete. A work is typically denoted by w, except for outputs of a model or user, denoted by y and z respectively. We denote by $\mathcal{C} \subset \mathcal{W}$ the set of all existing copyrighted works, and denote a particular copyrighted work by $c \in \mathcal{C}$. A dataset $D = (w_1, \ldots, w_n) \in \mathcal{W}^*$ is a list of existing works. We assume that the domain $\mathcal{W}$

A training algorithm Train takes as input a dataset D and returns a *conditional generative model* $p \in \mathcal{P}$, where $\mathcal{P}$ is the space of models. For every prompt $x \in \mathcal{X}$, the model p returns $y \in \mathcal{W}$ with probability $p(y|x)$. Typically, we treat p as an abstract representation of the corresponding conditional distributions. When we need to explicitly refer to a model's representation, we use θ for the parameters that define the model and p_θ for the corresponding model. A user u is an algorithm taking as input a conditional generative model p and auxiliary information aux and producing a work z.

3.2 Similarity, ideas, and access

We assume that SubSim, ideas, and scrub are exogenously-defined functions. They provide the legal-techncial interface needed to formalize a meaningful guarantee against copyright infringement without formalizing three important legal concepts: substantial similarity, ideas versus expression, and access. We consider precise technical definitions of these concepts out of scope (perhaps impossible).[4]

We never need to actually compute SubSim or ideas. We might think of these functions as capturing a hypothetical "true" version of the corresponding legal concept. Or we may take a more practical view. For example, $\mathsf{SubSim}(w)$ can instead be thought of as the set of works that one's attorney or a hypothetical jury would deem substantially similar to w. Despite being unknowable, our guarantees even with respect to this view. In contrast, scrub needs to be practical enough for the model provider (or data provider) to make the datset *golden* (Section 6.2).

Similarity and ideas The function SubSim defines what it means for one work to be *substantially similar* to another. For a work w, $\mathsf{SubSim}(w)$ denotes the set of all works that are substantially similar to w. For a set of works W, $\mathsf{SubSim}(W)$ denotes the set of all works similar to any w in W: $\mathsf{SubSim}(W) = \cup_{w \in D} \mathsf{SubSim}(w)$.

The function ideas defines the non-copyrightable *ideas* of a work, as opposed the expression of those ideas. For a work w, $\mathsf{ideas}(w)$ is some representation of the ideas contained in that work. (The precise representation is not important.) For a set of works W, we denote by $\mathsf{ideas}(W)$ the collection of all ideas across all the works w in W: $\mathsf{ideas}(W) = \{\mathsf{ideas}(w) : w \in D\}$.

We make a two weak assumptions on SubSim and ideas. First, every work is similar to itself: for all $w \in \mathcal{W}$, $w \in \mathsf{SubSim}(w)$. Second, mutually-dissimilar works sometimes share the same ideas: there exist $w, w' \in \mathcal{W}$ such that $\mathsf{ideas}(w) = \mathsf{ideas}(w')$ and $\mathsf{SubSim}(w) \cap \mathsf{SubSim}(w') = \emptyset$. (In Claim 6.3, we generalize this to k mutually-dissimilar works.)

Access The function scrub operationalizes the legal concept of *access*. It removes from a dataset anything that depends (in any way relevant to copyright) on some target work. The input to scrub is a dataset D and copyrighted work $c \in \mathcal{C}$. The output is a dataset denoted $\mathsf{scrub}(D, c)$ comprising all works in D except those which contain any copyrighted aspects of c. Legally, the property we need is that access to $\mathsf{scrub}(D, c)$ does not amount to access to c. This is a very strong property. It requires removing duplicates and derivatives—sequels, criticism, parodies, and excerpts thereof—whether infringing or not.

Formally, we define $\mathsf{scrub}(w, c) = \{w\}$ if w contains none of the copyrighted aspects of c, and $\mathsf{scrub}(w, c) = \emptyset$ otherwise. We define $\mathsf{scrub}(D, c) = \cup_{w \in D} \mathsf{scrub}(w, c)$. Note, we are abusing notation by treating works w and c as containing any information or metadata that may be needed to compute scrub.

[4] See [STV22] for an attempt at formalizing substantial similarity.

4 Warm up: when the model enables copying

This section defines what it means for a generative model to be *tainted*. Tainted models let users copy training data of which they have no knowledge. Whereas Section 6 gives sufficient conditions to prevent copying, this section gives a necessary condition: don't use tainted models. (Looking ahead, we will show that NAF models can be tainted.) First, we formulate a central object of our study: the interaction between a user and generative model.

4.1 The interaction between user and model

Real users interact with generative models in complicated ways. They refine their prompts based on previous results. They share prompts with each other. They are informed and inspired by existing artists and works. And the user's final product is often not itself an output of the model. They edit model outputs, combine them with their own original work, or merely use the model for inspiration. A meaningful copyright protection measure must protect these users too.

We make the interation between a user and the model explicit. Let u be a (randomized) algorithm called the *user*, p a conditional generative model, and aux some auxiliary input. We denote by $z \leftarrow u^p(\mathsf{aux})$ the randomized process of sampling a work $z \in \mathcal{W}$ as the output of u given black-box access to p and auxiliary input aux. Together with a training algorithm Train and a dataset D, this process naturally induces probability measure τ over $\mathcal{W}$.

Definition 4.1 (User's output distribution)**.** *Given a user u, training algorithm Train, dataset D, and auxiliary information aux, we define the* user's output distribution *τ over $\mathcal{W}$ as:*

$$\tau(w) = \Pr_{\substack{p \leftarrow \mathsf{Train}(D) \\ z \leftarrow u^p(\mathsf{aux})}}[z = w].$$

The probability is taken over the randomness of the algorithms Train, u, and p.

Of particular interest is $\tau(\mathsf{SubSim}(\mathcal{C}))$: the probability that the user's output is substantially similar to any copyrighted work. To prevent copyright infringement for a class of users, this quantity should be small.

4.2 Tainted models enable copying

Informally, Train is tainted (Definition 4.2) if there is a fixed algorithm u that copies some work from the training dataset using only trained model and the unprotected ideas of the work in question. If the goal is for Train to prevent a user from copying the training data, being tainted is as bad as it gets. A model produced by tainted training—a tainted model—does not prevent blameless copyright infringement. It enables it!

Definition 4.2 (Tainted training)**.** *We say Train is* tainted *if there exists a user u such that for all datasets D, and all $w \in D$:*

$$\tau\left(\mathsf{SubSim}(D)\right) > 0.99, \tag{1}$$

taking the $\mathsf{aux} = \mathsf{ideas}(w)$ in the user's output distrbution τ (Definition 4.1).

The order of quantifiers is important. The user u is fixed once and for all, while the data D and work $w \in D$ are arbitrary. The user cannot possibly be at fault: a finite-sized algorithm cannot "know" a work in all possible datasets D. Namely, for any user u and any fixed model q, there exists a dataset $D = \{w\}$ such that $\tau(\mathsf{SubSim}(w)) \ll 0.99$ (Example 4.3). Thus if $\tau(\mathsf{SubSim}(D)) > 0.99$ for all datasets D, it must be that the copying is enabled by model trained on D.

Example 4.3. *For any fixed model q, the constant algorithm $\mathsf{Train}_q(D) = q$ is not tainted. By assumption, there exist works $w_0, w_1 \in \mathcal{W}$ that are dissimilar from one another but which share the same ideas: $\mathsf{ideas}(w_0) = \mathsf{ideas}(w_1)$ and $\mathsf{SubSim}(w_0) \cap \mathsf{SubSim}(w_1) = \emptyset$. Let $D_i = \{w_i\}$ and fix a user u. Let τ_i be the user's output distribution (Definition 4.1) with $D = D_i$ and $\mathsf{aux} = \mathsf{ideas}(w_i)$. By construction, $\tau_1 = \tau^* = \tau_2$. Note that $\tau^*(w_i) \leq 1/2$ for one of $i = 0, 1$. Equation (1) is violated for the corresponding D_i.*

Example 4.4. *Algorithm 1 describes a tainted algorithm $\mathsf{Train}_{\mathsf{kv}}$, constructed from any training algorithm Train_0 in a black-box way. In words, $\mathsf{Train}_{\mathsf{kv}}(D)$ trains a model $q_0 \leftarrow \mathsf{Train}_0(D)$, and also builds a key-value store I mapping ideas $\mathsf{id} \in \mathcal{I}$ to the set $D_{\mathsf{id}} = \{w \in D : \mathsf{ideas}(w) = \mathsf{id}\}$. On prompt x, the model q_{kv} returns a random element of D_x if it is non-empty. Otherwise, it returns a generation sampled from $q_0(\cdot|x)$.*

It is easy to see that $\mathsf{Train}_{\mathsf{kv}}$ is tainted. Consider the user $u^p(\mathsf{aux})$ that returns a sample from $p(\cdot|\mathsf{aux})$. Fix D and $w \in D$, and let $p \leftarrow \mathsf{Train}_{\mathsf{kv}}(D)$ and $\mathsf{aux} = \mathsf{ideas}(w)$. By construction, the user always outputs an element of $D_{\mathsf{ideas}(w)}$. Hence, $\tau(\mathsf{SubSim}(D)) \geq \tau(D_{\mathsf{ideas}(w)}) = 1$.

Algorithm 1: $\mathsf{Train}_{\mathsf{kv}}$

Parameters: Training algorithm Train_0
Input: Data D
Output: Model q_{kv}

Let $q_0 \leftarrow \mathsf{Train}_0(D)$;
Initialize an empty key-value store I, whose keys are ideas id and values are sets of works $W \subset \mathcal{W}$;
for $w \in D$ **do**
  `// add w to I[ideas(w)];`
  $W \leftarrow I[\mathsf{ideas}(w)]$;
  $W \leftarrow W \cup \{w\}$;
  $I[\mathsf{ideas}(w)] \leftarrow W$;
end

Let q_{kv} the conditional generative model which on prompt x does the following:;
  $W \leftarrow I[x]$;
  if $W \neq \emptyset$ **then**
    Return $y \leftarrow_{\$} W$.
  else
    Return $y \sim q_0(\cdot|x)$.
  end

Result: q_{kv}

5 On near access-free models

Near access-freeness (NAF) is a mathematical definition intended to provide a "provable copyright protection" [VKB23]. This section describes near access-freeness and its limitations. Our main contribution showing that tainted model can enable verbatim copying while still satisfying NAF. For space, we defer the technical discussion of NAF to the appendix.

5.1 Overview of NAF and its limitations

Near access-freeness the first and only other attempt at a mathematical definition intended to offer "provable copyright protection," by Vyas, Kakade, and Barak [VKB23]. NAF requires a generative model p trained using a copyrighted work c to be close to a "safe" model safe trained without any access to c. How close is governed by a parameter $k \geq 0$, with smaller k imposing a stronger requirement. A corollary of the

closeness requirement is that the probability that p's output is substantially similar to c is not much greater than the probability for safe. We heuristically expect substantial similarity to be exceedingly unlikeley for the safe model, and therefore for the NAF model.

NAF is closely related to differential privacy (DP), as discussed at length in [VKB23] and [EKHLM24]. But surprisingly, [VKB23] show that NAF can be achieved relatively simply. Any training algorithm Train can be used in a black-box way to construct a NAF model, and without requiring additional "noise" as is the hallmark of differentially private algorithms.

NAF relevance to copyright law NAF is motivated by two concepts from copyright law: *access* and *substantial similarity*. The plaintiff in a copyright infringement claim has the burden of proving that the defendant copied original expression from the copyrighted work. The plaintiff does so by proving that (i) the defendant had access to the copyrighted work, and (ii) the defendant's work is substantially similar to the plaintiffs work.

With the above in mind, [VKB23] explain the relevance of NAF to copyright liability.

> To show a copyright violation has occurred the plaintiff must prove that "there are substantial similarities between the defendant's work and original elements of the plaintiff's work" (assuming access). It's negation would be to show that defendant's work is not substantially similar to the original elements of the plaintiff's work. Our approach would instead correspond to showing that the defendant's work is close to a work which was produced without access to the plaintiff's work. [W]e think this is a stronger guarantee...

As for why its a stronger guarantee, the argument is as follows. The probability that the defendant's work—produced by the real model p—is substantially similar to plaintiff's work is not much greater than the probability would have been had the defendant used the safe model (which had no access). We heuristically expect the latter probability to be exceedingly small. Hence, substantial similarity between the defendant's and plaintiff's works is exceedingly unlikely.

NAF's limitations A few prior works discuss and critique the NAF framework. Elkin-Koren et al. [EKHLM24] give the most extensive response, which we discuss in Section 2.2. The sharpest criticism is a brief discussion in the opus by Lee, Cooper, and Grimmelman [LCG24].[5] They argue that NAF is simply wrong on the law: "it is not a defense to copyright infringement that you would have copied the work from somewhere else if you hadn't copied it from the plaintiff." The other significant pushback concerns cases when the original expression of one work appears in many other works [EKHLM24, LCG24, HLJ+23]. That is, when the training data is not golden. See Section 7.1.2 for discussion about this issue.

In this work, we show that a model can be both NAF and tainted. NAF models don't prevent a user from reconstructing the training data, and may sometimes spit out the data directly. The error in [VKB23]'s intuition stems from ignoring the interaction between user and model. NAF offers a meaningful guarantee for a single prompt that is independent of the training data. In our counterexamples, the user uses many prompts, or a single prompt that depends on the *ideas* of a copyrighted work but not its copyrighted *original expresssion*.

Finally, we agree with [LCG24] that NAF's envisioned legal defense doesn't work (technical guarantees aside). To see why, consider a case in which the model's generated output was in fact substantially similar to a piece of training data. As to the access element, the defendant did in fact have access to the copied work by way of the model. The defendant's work may even be so "strikingly similar" to the plaintiff's that access becomes moot.[6] Despite being central to NAF, *access* appears to be a red herring. Whatever copyright protection NAF offers is by way of minimizing the likelihood of producing substantially similar outputs.

[6] "The plaintiff can prove that the defendant copied from the work by proving by a preponderance of the evidence that ... there is a striking similarity between the defendant's work and the plaintiff's copyrighted work." From the Ninth Circuit's *Manual of Model Civil Jury Instructions* `https://www.ce9.uscourts.gov/jury-instructions/node/326`.

6 Differential privacy's protection against copying

We give conditions that bound the probability of a user of a generative model copies a copyrighted work. Suppose, p is a conditional generative model that is the result of $\mathsf{Train}(D)$, and that u is a user of the model. Informally, the conditions are these. First, the training algorithm Train is (ε, δ)-*differentially private*. Second, the dataset D is *golden*: there are no duplicates from the point of view of copyright dependencies. Third, the user is α-*blameless*: in a clean room setting, the probability is at most α that the user u produces a result substantially similar to some copyrighted work not in the clean room. Section 7 reflects on whether these conditions are reasonable.

If all three conditions hold, then the probability that the user u produces a work that is substantially similar to any copyrighted work $c \in \mathcal{C}$ is at most approximately:

$$\tau(\mathsf{SubSim}(\mathcal{C})) \lessapprox e^{\varepsilon} \cdot N_D \cdot \alpha.$$

Here, N_D is the number of copyrighted works on which D depends. This means that the user can control the probability of infringement. To guarantee that the risk of infringement is at most r, the user can choose $\alpha \approx r/e^{\varepsilon} N_D$.

6.1 Differential privacy

An algorithm is differentially private (DP) if its output never depends too much on any one unit of input data. How much is "too much" is governed by a parameter $\varepsilon > 0$. Smaller values of ε provide stronger guarantees. In this work, a "unit of input data" is one work w in the training dataset D.

Definition 6.1 (Neighboring datasets)**.** *Datasets $D, D' \in \mathcal{W}^*$ are* neighboring *if they differ by inserting or deleting a single element. We denote neighboring datasets by $D \sim D'$.*

Definition 6.2 ((ε, δ)-Differential privacy)**.** *Let $\varepsilon, \delta \geq 0$, and let $M : \mathcal{W}^* \to \Omega$ be an algorithm mapping dataset $D \in \mathcal{W}^*$ to some output domain Ω. M is* (ε, δ)-differentially private ((ε, δ)-DP) *if for all neighboring pairs $D, D' \in \mathcal{W}^*$, and all subsets of outputs $S \subseteq \Omega$:*

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \cdot \Pr[M(D') \in S] + \delta.$$

Anything that one does with the output of a DP algorithm is also DP. In DP parlance, DP is robust to post-processing in the presence of arbitrary auxiliary information. Formally, for any function f, any string aux, and any set S:

$$\Pr[f(M(D), \mathsf{aux}) \in S] \leq e^{\varepsilon} \cdot \Pr[f(M(D', \mathsf{aux})) \in S] + \delta. \tag{2}$$

By itself, differential privacy already rules out tainted models.

Claim 6.3. *Suppose that there exist $w_1, \ldots, w_k \in \mathcal{W}$ such that for all i, j: $\mathsf{ideas}(w_i) = \mathsf{ideas}(w_j)$ and $\mathsf{SubSim}(w_i) \cap \mathsf{SubSim}(w_j) = \emptyset$. If Train is $(\ln(k) - 0.03, 0.01)$-DP, then it is not tainted.*

Proof. Let $D_i = \{w_i\}$ and τ_i be the user's output distribution when the model is trained on D_i. By hypothesis, the sets $\mathsf{SubSim}(D_i)$ are disjoint. Therefore $\sum_{i=1}^{k} \tau_1(\mathsf{SubSim}(D_i)) \leq 1$. Hence, there exists j^* for which $\tau_1(\mathsf{SubSim}(D_{j^*})) \leq 1/k$. If $j^* = 1$, then $\tau_1(\mathsf{SubSim}(D_1)) < 0.99$. For $j^* \neq 1$ we use DP. Because $D_1 \sim D_{j^*}$, we have that $\tau_{j^*}(\mathsf{SubSim}(D_{j^*})) \leq e^{\varepsilon}/k + \delta$. For $\delta \leq 0.01$ taking $\varepsilon < \ln(k) - \ln(0.98)$ implies that $\tau_{j^*}(\mathsf{SubSim}(D_{j^*})) < 0.99$ as required. □

6.2 Golden datasets

We adapt and formalize this idea from [VKB23], where it appears as an informal condition that suffices for their analysis: "It is important that when we omit a datapoint x it does not share copyrighted content with

many other datapoints that were included in the training set" [VKB23].[7] We say a dataset D is *golden* if, for every copyrighted work c, there is at most one item in D that contains any copyrighted aspects of c. That item could be c itself or a derivative work. Put differently, no protected original expression of any copyrighted work may appear in more than one element of a golden dataset. We operationalize this idea using the scrub function.

Definition 6.4 (Golden dataset)**.** *A dataset $D \subseteq \mathcal{W}$ is* golden *with repect to a set of copyrighted works $\mathcal{C} \subseteq \mathcal{W}$ if for all $c \in \mathcal{C}$:*

$$|D \setminus \mathsf{scrub}(D, c)| \leq 1.$$

Non-copyrighted works have no affect on the determination. By definition, $\mathsf{scrub}(D, w) = w$ for non-copyrighted w. This means that D could contain many derivatives of, say, the Mona Lisa, so long as the derivatives were independent of one another vis-a-vis copyright.

6.3 Blameless users don't copy in a clean room

We can't hope to guarantee that a generative model never reproduces copyrighted work, even if the model was not trained on that work. But we do want to guarantee it for "blameless" users. Which users are blameless?

We look to *clean rooms*. A clean room is method by which a firm can reverse engineer a product while shielding themselves from copyright claims. A team is put in the clean room and given a description of the ideas of the product (e.g., design specs) but none of its legally-protected expression. The outcome is constructively non-infringing. There is no access, as long as the team itself was not spoiled by prior familiarity with the product.

Our driving intuition is that for blameless users, a clean room works to prevent substantial similarity. Into the clean room we put a user and a generative model p, and task her with producing some original expressive work. A user who nevertheless produces an output that is substantially similar to a work c on which p does not depend, then the user culpable for the copying.

We now make this formal. First, we define the *clean room distribution* over user outputs. For any copyrighted work c, we use τ_{-c} to describe the user's output distribution in a clean room where the model doesn't depend on c. The only difference from the user's real-world output distribution (Definition 4.1), is that the model is trained on $\mathsf{scrub}(D, c)$ instead of D.

Definition 6.5 (Clean room distribution)**.** *Given a user u, training algorithm* Train*, dataset D, copyrighted work c, and auxiliary information* aux*, we define the user's* clean room distribution *τ_{-c} with respect to c:*

$$\tau_{-c}(w) = \Pr_{\substack{p \leftarrow \mathsf{Train}(\mathsf{scrub}(D,c)) \\ z \leftarrow u^p(\mathsf{aux})}}[z = w].$$

A user is blameless if, in the clean room, the user's output is substantially similar to a work outside the clean room with probablity at most α.[8] Given D and $\mathcal{C}$, we define the set $\mathcal{C}_{-D} \subseteq \mathcal{C}$ of all copyrighted works c of which no copyrighted aspects appear in D: $\mathcal{C}_{-D} = \{c \in \mathcal{C} : \mathsf{scrub}(D, c) = D\}$.

Definition 6.6 (α-blameless)**.** *For $0 \leq \alpha \leq 1$, a user u is* α-blameless *with respect to D, $\mathcal{C}$,* Train*, and* aux *if two conditions hold:*

- $\tau(\mathsf{SubSim}(\mathcal{C}_{-D})) < \alpha$*, and*
- *For all $c \in \mathcal{C}$, $\tau_{-c}(\mathsf{SubSim}(\mathcal{C}_D \cup \{c\})) < \alpha$.*

[7] We also borrow the term "golden dataset", though [VKB23] use it to refer to something entirely different: A dataset that is "carefully scrutinized to ensure that all material in it is not copyrighted or [is] properly licensed." This condition doesn't suffice. Even licensed derivates could cause unlicensed copying by the generative model. The same is true for NAF, despite their suggestion that it would yield a safe model.

Otherwise, u is α-blameworthy.

6.4 Bounding the probability of infringement

We're ready to put these conditions together to bound the probability of copyright infringement. The quantity of interest is $\tau(\mathsf{SubSim}(\mathcal{C}))$—the probability that the user's final product is substantially similar to any copyrighted work. As an element of copyright infringement, preventing substantial similarity suffices to prevent infringement.

For $\mathcal{C}$ and D, let $\mathcal{C}_D = \{c \in \mathcal{C} : \mathsf{scrub}(D, c) \neq D\}$ and $\mathcal{C}_{-D} = \mathcal{C} \setminus \mathcal{C}_D$. Let $N_D = |\mathcal{C}_D|$.

Theorem 6.7. *If* Train *is (ε, δ)-DP, D is golden, and u is α-blameless with respect to D, $\mathcal{C}$, and* aux*, then*

$$\tau(\mathsf{SubSim}(\mathcal{C})) \leq (e^{\varepsilon} N_D + 1)\alpha + N_D \delta.$$

Proof. Because D is golden, the datasets $\mathsf{scrub}(D, c)$ and D are either equal or neighboring for all $c \in \mathcal{C}$. By post-processing (2), $\tau(W) \leq e^{\varepsilon}\tau_{-c}(W) + \delta$ for all $W \subseteq \mathcal{W}$.

$$\begin{aligned}
\tau(\mathsf{SubSim}(\mathcal{C})) &\leq \tau(\mathsf{SubSim}(\mathcal{C}_D)) + \tau(\mathsf{SubSim}(\mathcal{C}_{-D})) \\
&\leq \sum_{c \in \mathcal{C}} \tau(\mathsf{SubSim}(c)) + \alpha \\
&\leq e^{\varepsilon} \cdot \sum_{c \in \mathcal{C}} \tau_{-c}(\mathsf{SubSim}(c)) + N_D \delta + \alpha \\
&\leq (e^{\varepsilon} N_D + 1)\alpha + N_D \delta
\end{aligned}$$

□

Remark 6.8 (Improving the union bound)**.** *In many cases, the factor of $N = |\mathcal{C}_D|$ from the union bound is unreasonably pessimistic. The user's intent limits the set of works she may infringe on. This is because the ideas underlying many works differ so greatly that they cannot reasonably be rendered in substantially similar ways. For example, a photorealistic image of birds will never be similar to a Dr. Seuss illustration. The probability can be safely approximated as 0. To formalize this, one might instead take the union bound over the set $\mathcal{C}_D^{\mathsf{id}} = \{c \in \mathcal{C}_D : \mathsf{id} \in \mathsf{ideas}(\mathsf{SubSim}(c))\}$. In most realistic settings, we expect $|\mathcal{C}_D^{\mathsf{id}}| \ll |\mathcal{C}_D|$.*

7 Discussion

7.1 Are the conditions reasonable?

We defined conditions on training, data, and users that together prevent copyright infringement. We now discuss how reasonable these conditions really are.

[8] The exact constant doesn't matter. It should be small enough to be exceedingly unlikely to occur by chance ($\ll 2^{-10}$, say). And it should be big enough to reflect that a relatively small amount of original expression is needed for copyright protection ($\gg 2^{-1000}$, say).

Here's a very crude justification for $\alpha = 2^{-50}$. Estimates of the entropy of English text vary. Shannon estimated it at 0.6 to 1.3 bits per character (A–Z and space), based on an study of how well his English-speaking test subjects were able to predict the next character in a passage [Sha51]. With those estimates, just 77 to 167 characters of text contain ≈100 bits of entropy. That's not many. Here is Sonnet 130 with the 77th and 167th characters highlighted.

> My mistress' eyes are nothing like the sun;
> Coral is far more red than her lips' **r**ed;
> If snow be white, why then her breasts are dun;
> If hairs be wires, black wires grow on her **h**ead.

Acknowledging that I know nothing of the matter, somewhere in that range seems reasonable for the minimum amount of original expression needed to afford copyright protection.

Infringement doesn't require reproducing all 100 bits of original expression. It only requires something substantially similar. If substantial similarity requires 50 of the 100 bits of minimum original expession to be reproduced, we get to 2^{-50}.

7.1.1 DP training

Differentially private machine learning is a very active area of research. Today, popular machine learning libraries come with off-the-shelf tools for using differential privacy, especially for generative models where DP is still far from practical. There is more hope for DP fine tuning [YNB+22] or in-context learning [WPWM23, DDPB24, TSI+23]. Here, the model provider specializes a foundation model to a particular application setting using a fine tuning dataset. Here we are concerned with copyright infringement only for the fine tuning data. (Perhaps the foundation model provider offers their own protections as to their training data.) Because much less data is needed to fine tune than to train from scratch, the increase for DP might be more manageable. Another option is privacy-preserving generation [DF18, ABK+24], wherein a user's black-box interaction with the model satisfies the DP guarantee though the model's parameters do not.

7.1.2 Golden datasets

Making a golden dataset large enough to train a state of the art generative model seems all but impossible in practice. This is one of the common arguments against NAF [EKHLM24, LCG24, HLJ+23], and it applies with equal strength to DP. It is much harder than deduplication, already a major challenge. First, it imposes a requirement with respect to copyrighted works that aren't even in the dataset. Second, whether one work copies another ultimately turns on human judgement and squishy legal standards. In an adversarial setting, copyright dependencies can be invisible to the human eye [LYL+24].

Still, a golden dataset with thousands or tens of thousands items may be practical. Though this won't suffice for training a model from scratch, it might suffice for fine tuning or in-context learning. In special cases, it may make sense to commission the creation of new data which is guaranteed to be golden.

There are non-technical ways to address the challenge of golden datasets. For example, suppose that the model provider licenses the training data from copyright holders. They might require the data to be tagged with information needed to determined copyright relationships and create a golden dataset. If a single agent collectively represents the copyright holders (e.g., Columbia Records), the model provider might require the data provider to make the dataset golden. In either case, the license agreement can indemnify users if the data provider's failure to appropriately clean or tag the data leads to infringement.

Not all is lost if we cannot guarantee that the dataset is golden. The "more golden" the dataset, the greater the protection against infringement. First, the protection against copying will still hold for any work that satisfies the golden condition. If $|D \setminus \mathsf{scrub}(D, c)| \leq 1$, then the probability of copying c is at most $\tau(\mathsf{SubSim}(c)) \leq e^{\varepsilon}\alpha + \delta$. One can generalize this to the set of works satisfying the golden condition. Second, the protection degrades smoothly (though exponentially fast) as the number of works in D that depends on c. By DP's group privacy property, $|D \setminus \mathsf{scrub}(D, c)| \leq k$ implies $\tau(\mathsf{SubSim}(c)) \leq e^{k\varepsilon}\alpha + ke^{(k-1)\varepsilon}\delta$.

7.1.3 Blameless users

Whether a user is blameless is not checkable, not even by the user herself. Still, we postulate that diligent users who are attentive to the possibility of inadvertant infringement can guarantee α-blamelessness for α small enough to make Theorem 6.7 meaningful. We offer some thoughts about this postulate next, but more justification is needed.

For blamelessness, it ought to suffice for users to avoid being unduly influenced by works to which they have previously been exposed. The structure of the clean room ought to suffice for all the other works. Note however that this does not follow from the copyright protection offered by the clean room. The clean room prevents access, making any substantial similarity fortuitous and not actionable. Still, we suggest that perhaps substantial similarity should be defined so that a resemblance that is likely to occur fortuitiously does not rise to the level of substantial similarity. Something like this is already true in copyright law (e.g., scenes-a-fair and the merger doctrine). What is or isn't considered substantially similar to a work c depends on the likely resemblance is to an independently-created work with the same ideas. It is also true about

striking similarity, which in some jurisdictions suffices to prove copying even without other evidence of access. Because a clean room constructively prevents infringement on any work that the user has never had access to, it must prevent a user from reproducing such works with striking similarity.

Finally, we arrive at the core question: Why do we think that users can avoid being unduly influenced by works to which they have been exposed? I don't have a good answer. More than anything, I can't shake the belief that I could do it. I think I could productively use Midjourney to produce an image that is wholly original, bearing no resemblence even to images with which I am intimately familiar. Perhaps you feel differently. People are able to act in a truly creative way, despite constant exposure to copyrighted work. Somehow we to manage to avoid copying every day.

7.2 What if copying occurs?

People copy without generative models, and will continue copying with them. Protective measures to mitigate inadvertent copying will not prevent all copying. One way to evaluate the usefulness of a copyright-mitigation measure is to consider what happens when copying does occur. Who is culpable? Who pays?

NAF doesn't give satisfying answers. Consider two examples. User A prompts, "`Write a story starting with: Mr. and Mrs. Dursley of number four, Privet Drive ...`" User B prompts, "`Write a story about an orphaned boy learns that he is a wizard and attends a magical boarding school`." Both users reproduce *Harry Potter and the Sorcerer's Stone* using the NAF model from Claim B.1. User A is clearly culpable. Her copying had little to do with the model. Any model would have served equally well. In User B's case, the model provider is clearly culpable. User B is also partially culpable to the extent that he was familiar with *Harry Potter*. It would be unreasonable to require User B to guard against the model copying a work if, as in the clean room, he only has access to the ideas of a work but not its expression. (Imagine an obscure book in place of *Harry Potter*.) But the legal analysis is murky. For models that regurgitate their training data, the model itself might fairly be seen as granting access.

Our DP-based approach gives clearer answers. It gives a theoretical account of who is culpable, and allows us to use a simple rule for deciding who pays. If the dataset was golden and the user wasn't blameless, then the user is culpable. If the reverse, then the model provider is culpable. If the dataset is golden and the user is blameless, the user simply got unlucky. Nobody is culpable. And if neither holds, then both parties are culpable.[9] Unfortunately this approach isn't practical, because we cannot generally determine whether the user was blameless.

We propose a simple rule for who pays, in the form of an indemnification provided by the model provider to the user. *The model provider pays for infringement if the copied work appears too often in the training data.* Under the rule, financial liability of copying is reduced to a relatively simple question of fact. The model provider foots the bill if copyrighted aspects of the copied work appears in more than one element of the training data. Otherwise, the user pays. Legal process is well-suited to resolve this rule. Experts for each party examine the training data and attempt to convince a fact finder.

The rule gives users what they want. They get indemnity for copying when there is any possibility that the model is to blame. Users can set the blamelessness parameter α small enough so their overall risk of paying the costs of infringement is tolerable.

The rule gives model providers what they want too. It's a valuable protection for users, giving a competitive edge. Moreover, the overall exposure to copyright penalties is can be controlled by making the dataset more golden (fewer duplicated works, or fewer duplicates per work). The more effort they expend cleaning the dataset, the lower the expected cost of the indemnifying users. If the potential penalties are small enough, they should be insurable. Finally, the model provider can pass the liability onto a third-party data provider who is contracted to provide golden datasets.

[9] In principle, one might even quantitatively allocate the blame between the parties. The contribution of the model provider is $e^{\varepsilon \cdot |D \setminus \mathsf{scrub}(D,c)|}$. The contribution of the user is the largest α for which he is α-blameworthy (though it can't be measured).

Acknowledgements

We are indebted to Mayank Varia for many helpful discussions and encouragement. We thank Gautam Kamath, Yu-Xiang Wang, Seewong Oh and especially Thomas Steinke for early discussions when the idea of this paper was still taking shape. Claim B.1 is based on an observation of Thomas. We thank the attendees of the 2024 Works-in-Progress Roundtable on Law and Computer Science at the University of Pennsylvania for feedback on an early draft.

References

[ABK+24] Kareem Amin, Alex Bie, Weiwei Kong, Alexey Kurakin, Natalia Ponomareva, Umar Syed, Andreas Terzis, and Sergei Vassilvitskii. Private prediction for large-scale synthetic text generation. *arXiv preprint arXiv:2407.12108*, 2024.

[DDPB24] Haonan Duan, Adam Dziedzic, Nicolas Papernot, and Franziska Boenisch. Flocks of stochastic parrots: Differentially private prompt learning for large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[DF18] Cynthia Dwork and Vitaly Feldman. Privacy-preserving prediction. In *Conference On Learning Theory*, pages 1693–1702. PMLR, 2018.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Conference on Theory of Cryptography*, TCC '06, pages 265–284, Berlin, Heidelberg, 2006. Springer.

[EKHLM24] Niva Elkin-Koren, Uri Hacohen, Roi Livni, and Shay Moran. Can copyright be reduced to privacy? Forthcoming, Foundations of Responsible Computing, 2024. `https://arxiv.org/abs/2305.14822`.

[GAZ+24] Aditya Golatkar, Alessandro Achille, Luca Zancato, Yu-Xiang Wang, Ashwin Swaminathan, and Stefano Soatto. Cpr: Retrieval augmented generation for copyright protection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12374–12384, 2024.

[HLJ+23] Peter Henderson, Xuechen Li, Dan Jurafsky, Tatsunori Hashimoto, Mark A. Lemley, and Percy Liang. Foundation models and fair use. *Journal of Machine Learning Research*, 24(400):1–79, 2023.

[LCG24] Katherine Lee, A. Feder Cooper, and James Grimmelmann. Talkin' 'bout ai generation: Copyright and the generative-ai supply chain. Forthcoming, Journal of the Copyright Society, 2024. `https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4523551`.

[LYL+24] Yiwei Lu, Matthew Y. R. Yang, Zuoqiu Liu, Gautam Kamath, and Yaoliang Yu. Disguised copyright infringement of latent diffusion models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[Sha51] Claude E Shannon. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64, 1951.

[STV22] Sarah Scheffler, Eran Tromer, and Mayank Varia. Formalizing human ingenuity: A quantitative framework for copyright law's substantial similarity. In *Proceedings of the 2022 Symposium on Computer Science and Law*, pages 37–49, 2022.

[TSI+23] Xinyu Tang, Richard Shin, Huseyin A Inan, Andre Manoel, Fatemehsadat Mireshghallah, Zinan Lin, Sivakanth Gopi, Janardhan Kulkarni, and Robert Sim. Privacy-preserving in-context learning with differentially private few-shot generation. *arXiv preprint arXiv:2309.11765*, 2023.

[VKB23] Nikhil Vyas, Sham Kakade, and Boaz Barak. Provable copyright protection for generative models. *arXiv preprint arXiv:2302.10870*, 2023.

[WPWM23] Tong Wu, Ashwinee Panda, Jiachen T. Wang, and Prateek Mittal. Privacy-preserving in-context learning for large language models, 2023.

[YNB+22] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

A Definitions and main results from [VKB23]

Informally, to satisfy NAF, the output distributions of p and safe must be close for every prompt x. A corollary is that for any event E, NAF guarantees that $p(E|x) \leq 2^{k_x} \cdot \mathsf{safe}(E|x)$, where the constant k_x may depend on x. As substantial similarity is necessary for infringement, taking $E = \mathsf{SubSim}(c)$ to be the event that the model's output $y \in \mathcal{W}$ is substantially similar to c bounds the probability of the output of p infringing on the copyright of c. If safe is trained without c and x doesn't itself encode c in some way, then one would expect $\mathsf{safe}(\mathsf{SubSim}(c)|x)$ to be miniscule.

We define near access-freeness (NAF) next.[10] NAF is defined with respect to a function safe. The safe function maps a copyrighted datapoint $c \in \mathcal{C}$ to a generative model $\mathsf{safe}_c \in \mathcal{P}$ trained without access to c. The intuition is that safe is unable to cause copyright violations with respect to c (for any reasonable choice of safe). [VKB23] present $\mathsf{sharded\text{-}safe}$ (Alg. 2) an example safe function.

Algorithm 2: $\mathsf{sharded\text{-}safe}$ [VKB23]

Parameters: Dataset D, training algorithm Train

Do the following once: `// or derandomize for statelessness`;

Partition D into disjoint datasets D_1 and D_2;

Set $q_1 \leftarrow \mathsf{Train}(D_1)$, $q_2 \leftarrow \mathsf{Train}(D_2)$;

Input: $c \in \mathcal{C}$

Let $i = \min\{j : c \notin D_j\}$;

Result: q_i

Definition A.1 (Max KL divergence)**.** *For two distributions p, q, the* max KL divergence *is* $\Delta_{\max}(p\|q) = \max_{y \in \mathrm{Supp}(p)} \log \frac{p(y)}{q(y)}$.

Definition A.2 (k_x-NAF [VKB23])**.** *Let $\mathcal{C}$ be a set of datapoints, and let $\mathsf{safe} : \mathcal{C} \to \mathcal{P}$. A generative model p is* k_x-near access-free (k_x-NAF) *on prompt $x \in \mathcal{X}$ with respect to $\mathcal{C}$ and safe if for every $c \in \mathcal{C}$,*

$$\Delta_{\max}\Big(p(\cdot|x) \parallel \mathsf{safe}_c(\cdot, x)\Big) \leq k_x.$$

A model p is k-NAF *with respect to $\mathcal{C}$ and safe if for all $x \in \mathcal{X}$, it is k_x-NAF for some $k_x \leq k$.*

[10] The only change we make to the original definitions of [VKB23] is that we fix the divergence measure to the maximum KL divergence $\Delta_{\max}$, simplifying the notation by dropping the dependence on $\Delta_{\max}$.

The appeal of this definition is that it can be used to bound the probability that model p produces outputs that violate the copyright of a work c, relative to the probability under safe. This is made precise in the following lemma.

Lemma A.3 (k-NAF event bound [VKB23])**.** *Suppose model p is k_x-NAF on prompt x with respect to $\mathcal{C}$ and safe. Then for any $c \in \mathcal{C}$ and any event $E \subseteq \mathcal{Y}$:*

$$p(E|x) \leq 2^{k_x} \cdot \mathsf{safe}_c(E|x).$$

Let $\mathsf{SubSim}(c)$ be the event that y is substantially similar to c. Then

$$p(\mathsf{SubSim}(c)|x) \leq 2^{k_x} \cdot \mathsf{safe}_c(\mathsf{SubSim}(c)|x). \tag{3}$$

As a copyright violation requires substantial similarity, bounding k_x suffices to prevent violation whenever $\mathsf{safe}_c(\mathsf{SubSim}(c)|x)$ is negligibly small. Note however that $\mathsf{safe}_c(\mathsf{SubSim}(c)|x)$ may be large, for example if the prompt x itself contains the copyrighted work c [VKB23].

Finally, [VKB23] gives algorithms for achieving k_x-NAF. Their main algorithm is called CP for *copy protection* (Algorithm 3). Theorem A.4 states that CP is k_x-NAF with respect to $\mathsf{sharded\text{-}safe}$ (Algorithm 2). VKB remarks that the theorem gives "strong bounds on the probability of sampling protected content."

Theorem A.4 ([VKB23])**.** *Let p be the model returned by CP, and q_1 and q_2 be the models returned by $\mathsf{sharded\text{-}safe}$. Then p is k_x-NAF with respect to $\mathcal{C}$ and $\mathsf{sharded\text{-}safe}$, with*

$$k_x \leq -\log\Big(1 - d_{\mathrm{TV}}\big(q_1(\cdot|x), q_2(\cdot|x)\big)\Big). \tag{4}$$

Observe that $\mathsf{CP}(D)$ is well-defined iff $\forall x$, $\exists y$: $q_1(y|x) > 0$ and $q_2(y|x) > 0$.

Algorithm 3: CP: Copy-Protection [VKB23]

Input: Dataset D

Learn: Run $\mathsf{sharded\text{-}safe}(D)$ to obtain q_1, q_2 as in Algorithm 2

Result: The model p with

$$p(y|x) = \frac{\min\{q_1(y|x), q_2(y|x)\}}{Z(x)}$$

where $Z(x)$ is a normalization constant that depends on x.

B Tainted NAF models

We now describe two tainted training algorithms that produce NAF models. In each case, a user can cause the the model to produce training data verbatim. In one case, the model generates w when queried with the prompt $\mathsf{ideas}(w)$ — a query that depends on w but not on any copyright aspects of w. In the other, the user queries a fixed set of prompts, recovering k bits of the dataset with each query.

B.1 The CP algorithm may be tainted

We now show that CP—the main NAF algorithm of [VKB23]—may be tainted. Using a prompt that is independent of any copyrightable expression, the user can cause the NAF model returned by CP to regurgitate its training data.[11] To do so, the user picks a prompt x for which the k_x-NAF guarantee of Theorem A.4 is vacuous. This example is based on an observation of Thomas Steinke.

[11] This doesn't violate Theorem A.4. We use the fact that the CP algorithm is not actually k-NAF for any constant k. Applied to our construction, the bound on $k = k_x$ given by the theorem is not meaningful for $x \in \mathsf{ideas}(D)$. [VKB23] also give a version of their algorithm, CP-k, that is k-NAF with a uniform k for all prompts. Adapting our construction to that algorithm yields a k-NAF model that either regurgitates training data or fails to terminate.

Technically, we leverage the fact that $\mathsf{sharded\text{-}safe}$—and hence also CP—depends on an underlying training algorithm Train at its core. As such, it actually describes a family of algorithms, one for each instantiation of Train. We show that using $\mathsf{Train}_{\mathsf{kv}}$ in Algorithm 1 (see Example 4.4), the induced $\mathsf{CP}_{\mathsf{kv}}$ enables retrieval of training data just like in Example 4.4.

Recall that $\mathsf{Train}_{\mathsf{kv}}$ is itself built from some underlying training algorithm Train_0. The only requirement we need of Train_0 is that it produces models with *full support*. That is, for every dataset D, model $p_0 \leftarrow \mathsf{Train}_0(D)$, prompt x, and output $y \in \mathcal{W}$, we require $p(y|x) > 0$.

Claim B.1. *Let D be a dataset and $w \in D$. For ideas $\mathsf{id} \in \mathcal{I}$, let $D_{\mathsf{id}} = \{w' \in D : \mathsf{ideas}(w') = \mathsf{id}\}$. Let Train_0 produce models with full support. Let $\mathsf{Train}_{\mathsf{kv}}$, $\mathsf{sharded\text{-}safe}_{\mathsf{kv}}$ and $\mathsf{CP}_{\mathsf{kv}}$ be as defined in Algorithms 1, 2, and 3, defined with respect to Train_0 (and the preceding algorithms). Let $p \leftarrow \mathsf{CP}_{\mathsf{kv}}(D)$. Then $p\left(D_{\mathsf{ideas}(w)} \mid \mathsf{ideas}(w)\right) = 1$.*

In particular, for any copyrighted work $c \in D$ whose unique ideas are unique, we get that $p(c \mid \mathsf{ideas}(c)) = 1$. Note that Claim B.1 does not contradict any claims in[VKB23]. Applying Theorem A.4 and Lemma A.3 yields the correct but trivial bound $p(c|\mathsf{ideas}(c)) \leq 1$.

Proof of Claim B.1. Unrolling the algorithms, $\mathsf{CP}_{\mathsf{kv}}(D)$ calls $\mathsf{sharded\text{-}safe}_{\mathsf{kv}}$, which shards the data into D_1 and D_2 and trains $q_i \leftarrow \mathsf{Train}_{\mathsf{kv}}(D_i)$. Without loss of generality, suppose $w \in D_1$.

By construction, $q_1(D_{\mathsf{ideas}(w)}|\mathsf{ideas}(w)) = 1$ (as in Example 4.4). Thus for all outputs $y \in \mathcal{W}$:

$$p(y|\mathsf{ideas}(w)) \propto \min\{q_1(y|\mathsf{ideas}(w)), q_2(y|\mathsf{ideas}(w))\} = \begin{cases} q_2(y|\mathsf{ideas}(w)) & \text{if } y \in D_{\mathsf{ideas}(w)} \\ 0 & \text{if } y \notin D_{\mathsf{ideas}(w)} \end{cases}$$

The distribution $p(\cdot|\mathsf{ideas}(w))$ is well defined if there exists $y \in D_{\mathsf{ideas}(w)}$ such that $q_2(y|\mathsf{ideas}(w)) > 0$. The model $q_2(\cdot|\mathsf{ideas}(w))$ returns an element of $D_2 \cap D_{\mathsf{ideas}(w)}$ if it has non-empty intersection, or it returns a sample from an underlying model with full support (the model returned by Train_0). Either way, $q_2(y|\mathsf{ideas}(w)) > 0$ for all $y \in D_2 \cap D_{\mathsf{ideas}(w)}$.

The claim follows immediately:

$$p(D_{\mathsf{ideas}(w)} \mid \mathsf{ideas}(w)) = 1 - p(\overline{D_{\mathsf{ideas}(w)}} \mid \mathsf{ideas}(w)) = 1. \qquad \square$$

Corollary B.2. $\mathsf{CP}_{\mathsf{kv}}$ *is tainted.*

Proof. Consider the user $u^p(\mathsf{aux})$ that returns a sample from $p(\cdot|\mathsf{aux})$. Fix D and $w \in D$, and let $p \leftarrow \mathsf{CP}_{\mathsf{kv}}(D)$ and $\mathsf{aux} = \mathsf{ideas}(w)$. By Claim B.1, $\tau(\mathsf{SubSim}(D)) \geq \tau(D_{\mathsf{ideas}(w)}) = 1$. $\square$

B.2 NAF does not prevent full reconstruction

NAF may allow training data to be reconstructed, even if k is arbitrarily small and independent of x. This is because the k-NAF guarantee does not compose across a user's many queries, and each query may leak up to k bits of training data.

To make this concrete, we describe a family of models that are NAF with respect to a model that returns pure noise, yet enable a user to reconstruct the dataset verbatim. For any $\ell \geq 1$, let $\mathsf{coin}_\ell(\cdot|x)$ be uniform over $\{0,1\}^\ell$ for all prompts x. As a generative model, $\mathsf{coin}_\ell(\cdot|\cdot)$ is clearly a "safe" instantiation of safe_c for any copyrighted work c.

Claim B.3. *Fix $\mathcal{C} \subseteq \mathcal{W} \subseteq \{0,1\}^*$. For $D \in \mathcal{W}^*$, let L be total the length of D in bits. There exists a (deterministic) training algorithm $\mathsf{Train} : (D,k) \mapsto p_{k,D}$ satisfying the following.*

- *For all D and $k > 0$: $p_{k,D}$ is k-NAF with respect to $\mathcal{C}$ and coin_ℓ for $\ell = \max\{1, \lfloor k \rfloor\}$.*
- *There exists a user u such that for all D and $k > 0$: u makes $\mathrm{poly}(L, 1/k)$ queries to $p_{k,D}$ and outputs D with probability > 0.99.*

The following corollary is immediate.

Corollary B.4. *For any k, there exists a tainted training algorithm k such that* $\mathsf{Train}(D)$ *is k-NAF.*

Proof of Claim B.3. We parse D as a bitstring of length L, and let $D[j]$ be its jth bit. For $k \geq 1$, Train outputs the model $p_{k,D}$ as follows:

$$p_{k,D}(y|x) = \begin{cases} (D[x], D[x+1], \ldots, D[x+\ell-1]) & \text{if } x \in [L-\ell+1] \\ y \leftarrow\!\!\$\ \{0,1\}^\ell & \text{otherwise} \end{cases}$$

The model $p_{k,D}$ is k-NAF with respect to $\mathcal{C}$ and coin_ℓ: $\forall x$, $\Delta_{\max}\big(p_{k,D}(\cdot|x) \parallel \mathsf{safe}_c(\cdot, x)\big) \leq k$. To reconstruct D, the user u queries $p_\ell(\cdot|x)$ for $x = i\ell + 1$ for $i = 0, 1, \ldots, (L-1)/\ell$.

For $0 < k < 1$, Train sets $\alpha = 2^k - 1$ and outputs the model $p_{k,D}$ as follows:

$$p_{k,D}(y|x) = \begin{cases} y \leftarrow \mathsf{Bernoulli}(\frac{1}{2} + \alpha(D[x] - \frac{1}{2})) & \text{if } x \in [L] \\ y \leftarrow \mathsf{Bernoulli}(\frac{1}{2}) & \text{otherwise} \end{cases}$$

The intuition is that $p_{k,D}$ encodes $D[x]$ in the bias of the output of $p_{k,D}(\cdot|x)$, with the magnitude of $\alpha \in (0,1)$ controlling the strength of the bias. The model $p_{k,D}$ is k-NAF with respect to coin_1: $\forall x$, $\Delta_{\max}\big(p_{k,D}(\cdot|x) \parallel \mathsf{safe}_c(\cdot, x)\big) = \log \frac{1/2+\alpha(D[x]-1/2)}{1/2} \leq \log(1+\alpha) \leq k$.

To determine $D[j]$ with greater than $> 1 - \frac{1}{100L}$, the user u makes $\mathrm{poly}(L, \log 1/\alpha) = \mathrm{poly}(L, 1/k)$ queries to the model, and stores the majority. The user does this for each $j \in [L]$ and outputs the result. By a union bound, the user's output is equal to D with probability greater than 0.99. □

Remark B.5. *This claim and the construction can be adapted to more realistic* safe*. Say, by encoding D in the bias of a hash of p's outputs. But the added complexity would obscure the main idea in the proof: that biasing* safe *can reveal D without violating NAF.*