Blameless Users in a Clean Room: Defining Copyright Protection for Generative Models

Aloni Cohen
Department of Computer Science
University of Chicago
aloni@uchicago.edu

Abstract

Are there any conditions under which a generative model's outputs are guaranteed not to infringe the copyrights of its training data? This is the question of "provable copyright protection" first posed in [VKB23]. They define *near access-freeness* (NAF) and propose it as sufficient for protection. This paper revisits the question and establishes new foundations for provable copyright protection—foundations that are firmer both technically and legally. First, we show that NAF alone does not prevent infringement. In fact, NAF models can enable verbatim copying, a blatant failure of copy protection that we dub being *tainted*. Then, we introduce our *blameless copy protection* framework for defining meaningful guarantees, and instantiate it with *clean-room copy protection*. Clean-room copy protection allows a user to control their risk of copying by behaving in a way that is unlikely to copy in a counterfactual "clean-room setting." Finally, we formalize a common intuition about differential privacy and copyright by proving that DP implies clean-room copy protection when the dataset is *golden*, a copyright deduplication requirement.

1 Introduction

A user of a generative model is worried about unwitting copyright infringement. She is worried that the model's outputs might resemble copyrighted works in the training data through no fault of her own, exposing her to legal liability. The user would like some assurance that this won't happen. We ask: **What assurances can the model provider give, and under what conditions?**

Ideally, the generative model would never reproduce copyrighted work. That's unrealistic. Typical models can be prompted to generate copyrighted output: `print the following ____` [VKB23, LCG24]. It's also unnecessary. In the example, copying is clearly the user's fault. Instead, we should protect blameless users from inadvertent infringement. If the model reproduces copyrighted work, it should be because the user induced it to (or was very unlucky). This guarantee should satisfy a careful, honest user. As long as the user's use of the model does not itself cause the infringement, consciously or subconsciously, then the result is unlikely to infringe.

Our goal is to formalize such a guarantee—in a mathematically and legally rigorous way—and to study its feasibility. We're after "provable copyright protection" at deployment time, first studied by Vyas, Kakade, and Barak [VKB23]. They propose *near access-freeness* (NAF) as an answer.

Our contributions This paper establishes new foundations for provable copyright protection—foundations that are firmer both mathematically and legally than prior work.

1. **We prove that *near access-free* (NAF) models can enable verbatim copying.** NAF is the first (and only other) attempt at a mathematical definition intended to offer "provable copyright protection"

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

 https://doi.org/10.52202/085713-1903

for generative models [VKB23]. The proof leverages NAF's lack of protection against multiple prompts or data-dependent prompts.

2. **We define *tainted models*, which enable users to reproduce verbatim training data** despite knowing nothing about the underlying dataset, a blatant failure of copy protection (Section 5). A meaningful definition of provable copyright protection must exclude tainted models. NAF does not.

3. **We introduce a framework for defining copy protection guarantees, called *blameless copy protection*** (Section 6). It protects *blameless* users—those who don't themselves induce infringement—from unwitting copying.

4. **We define *clean-room copy protection* ((κ, β)*-clean*), a first instantiation of our framework** (Section 7), drawing inspiration from *clean-room design*. A training algorithm is (κ, β)-clean if, for every user who copies in a (counterfactual) "clean-room environment" with probability $\leq \beta$, the probability of copying in the real world is $\leq \kappa$. Clean-room copy protection lets users choose their tolerance for risk κ and then tune β accordingly. Clean-room copy protection also excludes tainted models under mild assumptions.

5. **We prove that *differential privacy* (DP) implies clean-room copy protection**, formalizing a common intuition about DP and copyright (Section 8). This holds when the dataset is *golden*, a copyright deduplication requirement. Thus, DP provides a way to bring copyrighted expression "into the clean room" without tainting the model.

This paper does not offer a legal analysis of where liability would lie under existing law. Instead,our paper develops a mathematical framework for preventing unwitting infringement—where holding the user liable would be unjust—and proposes a rule for assigning fault when copying does inevitably occur (Section 9).

Related work There is a lot of recent work on copyright questions for generative AI [CLG+23, LCG24, HLJ+23], and the first generation of cases are working their way through the courts (e.g., *New York Times v Microsoft*). 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—near access-freeness (NAF)—aimed at "preventing deployment-time copyright infringement" [VKB23]. Elkin-Koren, Hacohen, Livni, and Moran argue that copyright cannot be "reduced to privacy" [EKHLM24]. 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. Li, Shen, and Kawaguchi propose and empirically evaluate an attack called VA3 against NAF [LSK24]. They provide good evidence that the CP-k algorithm of [VKB23] may not prevent infringement, but some uncertainty remains (see Appendix B.3). We discuss these three papers at length. There is also work on new NAF algorithms [GAZ+24, CKOX24].

Scheffler, Tromer, and Varia [STV22] give a complexity-theoretic account of substantial similarity, and a procedure for adjudicating disputes. In contrast, we treat substantial similarity as a black box. That differential privacy might protect against infringement has been suggested in [BLM20, HLJ+23, VKB23, EKHLM24, CKOX24]. Livni, Moran, Nissim, and Pabbaraju [LMNP24] study the problem of attributing credit to inputs of an algorithm that influence its output, motivated by copyright. They define a condition under which a given input need not be credited (with credit required otherwise). Under their definition, a differentially private algorithm never has to credit its inputs—a manifestation of the common intuition that differential privacy prevents copying which we discuss in Section 8.

1.1 Notation

$\mathcal{W}$ is the domain of *works* protected by copyright law. For example, $\mathcal{W}$ might be the space of possible images, texts, or songs. We assume $\mathcal{W}$ is discrete. We usually use $w \in \mathcal{W}$ to denote an individual work, along with $c, y, z \in \mathcal{W}$ as described in the following. We denote by $\mathcal{C} \subseteq \mathcal{W}$ the set of all in-copyright works—those currently under copyright protection—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 works with multiplicities allowed. We sometimes abuse notation and treat D as a set. A *conditional generative model (model)* p is a mapping from a prompt x to a probability distribution $p(\cdot|x)$ over $\mathcal{W}$ which samples $y \in \mathcal{W}$ with probability $p(y|x)$. A training algorithm Train maps a dataset D to a model p. A user u is an algorithm which

uses black-box access to a model p, along with *auxiliary input* $\mathsf{aux} \in \{0,1\}^* \cup \{\perp\}$, and outputs a work $z \in \mathcal{W}$ (see Section 4).

Symbol	Definition
$\mathcal{W}$; w, y, z	Domain of works (e.g., images, text); individual works
D	Dataset $D = (w_1, \ldots, w_n)$
Train	Training algorithm mapping datasets D to models p
p; $p(y \mid x)$	Conditional generative model; the probability of generating y on prompt x
aux	Auxiliary information given to the user (usually $\mathsf{aux} = \mathsf{ideas}(w)$)
u	User taking input aux and access to p and outputting a work $z \leftarrow \mathsf{u}^p(\mathsf{aux})$
τ; $\tau(E; \mathsf{aux})$	Real-world distribution; the probability of event $z \in E$ in the probability experiment $p \leftarrow \mathsf{Train}(D)$, $z \leftarrow \mathsf{u}^p(\mathsf{aux})$
$\mathcal{C}$; c	Set of **in-copyright** works; an in-copyright work
$\mathsf{SubSim}(w)$	Set of works **substantially similar** to w
$\mathsf{ideas}(w)$	Non-copyrightable **ideas** associated with w
"w' stems from w"	w' is a **derivative work** of w, represented as edges in graph G
$\mathcal{C}_D$; $\mathcal{C}_{-D}$	In-copyright works from which some work ($\mathcal{C}_D$) / no work ($\mathcal{C}_{-D}$) in D stems
$\mathsf{scrub}(D, c)$	Sub-dataset of works in D that don't stem from c
τ_{-c}; $\tau_{-c}(E; \mathsf{aux})$	Clean-room distribution, defined like τ but replacing D with $\mathsf{scrub}(D, c)$

Table 1: Common symbols and their definitions. Bolded terms are exogenously (legally) defined.

2 The legal-technical interface

To use legal concepts within a mathematical formalism, we assume some exogenously-defined functions reflecting and operationalizing those concepts. They are the interface between math and law, enabling mathematical study of legal concepts without formalizing the concepts themselves. Three legal concepts are needed for this paper: *substantial similarity*, *ideas*, and *access*. This section defines functions SubSim and ideas for the first two. Section 7.1 operationalizes access.

To prove copying, a plaintiff (copyright holder) must prove two things: *substantial similarity* and *access*. The former requires the defendant work's to be substantially similar to the original, with vague tests varying by jurisdiction. Access requires the defendant to have had a reasonable opportunity to view the original (e.g., it's online). Independent creation of substantially similar works is allowed. The function SubSim defines what it means for one work to be substantially similar to another.

Definition 2.1 (SubSim)**.** *For a work $w \in \mathcal{W}$, the set $\mathsf{SubSim}(w) \subseteq \mathcal{W}$ contains all works w' that are substantially similar to w. We assume w substantially similar to itself: $w \in \mathsf{SubSim}(w)$. For a set of works $W \subseteq \mathcal{W}$, we define $\mathsf{SubSim}(W) = \cup_{w \in W} \mathsf{SubSim}(w)$.*

Copyright does not protect *ideas*, only *original expression*. For example, the text and images in a cookbook may be copyrighted, but not the method of cooking a dish. To be afforded protection, there must be a minimum of creativity beyond the ideas. The function ideas defines the non-copyrightable *ideas* of a work, as opposed their expression. We represent these ideas as a string which one can view as encoding a set or any other data type.

Definition 2.2 (ideas)**.** *For a work $w \in \mathcal{W}$, the string $\mathsf{ideas}(w) \in \{0,1\}^* \cup \{\perp\}$ is (some representation of) the ideas contained in w. For a set of works $W \subseteq \mathcal{W}$, we define the set of all their ideas $\mathsf{ideas}(W) = \{\mathsf{ideas}(w) : w \in \mathcal{W}\}$.*

We never actually compute SubSim nor ideas. This is a feature: it means we can be agnostic as to the specific instantiation. One might view them as capturing the "true" legal concept. Or one may take a more practical view. For example, interpreting $\mathsf{SubSim}(w)$ as reflecting the opinion of a typical judge or jury. However the reader wishes to interpret these functions is fine (up to any assumptions stated where used). **In particular, our results hold even for $\mathsf{SubSim}(w) = \{w\}$ and $\mathsf{ideas}(w) = \perp$.** In contrast, a model provider (or data provider) would need to actually work with the copyright dependency graph in Section 7. We discuss this limitation in Section 9.

3 Near access-free models do not provide provable copyright protection

Near access-freeness (NAF) is a mathematical definition for "provable copyright protection" [VKB23]. This section describes near access-freeness and its limitations. We prove that models can enable

 https://doi.org/10.52202/085713-1903

verbatim copying while still satisfying NAF. While NAF provides protection against a single prompt that is independent of the training data, it makes no guarantees against many prompts (composition) [LSK24], nor a single prompt derived from non-copyrightable ideas (not expression). We emphasize that our focus is *definitions* for provable copyright protection, where NAF falls short. Practically, NAF may still provide useful protection in many natural settings. For space, we defer algorithms, proofs, and additional discussion to Appendix B.

3.1 NAF Background

NAF informally *Near access-freeness* (NAF) is the first (and only other) attempt at a mathematical definition intended to offer "provable copyright protection" for generative models [VKB23]. Motivated by the legal requirement of access, NAF is meant to show "that the defendant's work is close to a work which was produced without access to the plaintiff's work" [VKB23]. NAF is closely related to DP. But surprisingly, any training algorithm Train can be used in a black-box way to construct a NAF model without the extra noise that is the hallmark of DP (Theorem 3.3).

The definition of 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 $k \geq 0$: smaller k means closer models. A corollary of the NAF requirement is that for any prompt x, the probability p generates something substantially similar to c is at most 2^k-times greater than safe doing so. We heuristically expect the latter probability to be miniscule, if safe and x are independent of c. If so, this bounds the probability that p's output infringes on prompt x.

Known limitations of NAF Prior works discuss and critique the NAF framework in three ways: questioning NAF's technical effectiveness [LSK24], its legal applicability [LCG24, EKHLM24], and its real-world practicality [HLJ$^+$23, EKHLM24, LCG24, CKOX24].

$*$ *Technically*, Li, Shen, and Kawaguchi propose and empirically evaluate an attack called VA3 against NAF [LSK24]. See Appendix B.3 for a full discussion. They show that a black-box attacker, can reliably induce a model to produce outputs that infringe on a target work c^*. They also give a white-box prompt writing algorithm for diffusion models that greatly improves performance. VA3 provides good evidence that the CP-k algorithm of [VKB23] may not prevent infringement. Some uncertainty remains because the algorithm implemented in VA3 deviates from the original. The paper leaves open whether k-NAF (with fixed k) prevents copyright infringement, formalizes a flawed attack model, and avoids the underlying definitional questions almost entirely.

$*$ *Legally*, Elkin-Koren et al. [EKHLM24] and Lee, Cooper, and Grimmelman [LCG24] argue that NAF is wrong on the law. See Appendix B.4 for a detailed discussion of these arguments.

$*$ *Practically*, the main concern is that deduplicating data in the way needed to make NAF effective is infeasible [EKHLM24, LCG24, HLJ$^+$23]. It is unrealistic to produce enough clean training data to pretrain foundation models, advances in deduplication notwithstanding. Moreover, NAF learning algorithms are computationally expensive and may lag in performance [CKOX24]. These concerns are justified and also apply to DP. Still, there may be settings where golden data is practical (Sec. 9), to say nothing of the value of theory.

3.2 NAF definition and the Copy Protection (CP) algorithm

Near access-freeness (NAF) is defined with respect to a function safe. The safe function maps a copyrighted data point $c \in \mathcal{C}$ to a generative model $\mathsf{safe}_c \in \mathcal{P}$ trained without access to c. An example $\mathsf{sharded\text{-}safe}$ is in Appendix B (Alg. 1). Thm. 3.3 presents the main feasibility result: any training algorithm Train can be used as a black-box to construct an NAF model CP, short for *copy protection*.

Definition 3.1 (Max KL divergence)**.** *For distributions p, q, $\Delta_{\max}(p\|q) := \max_{y \in \mathrm{Supp}(p)} \log \frac{p(y)}{q(y)}$.*

Definition 3.2 (k_x-NAF [VKB23])**.** *Fix a set $\mathcal{C}$ and function $\mathsf{safe} : \mathcal{C} \to \mathcal{P}$. 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 $k_x \leq k$.*

NAF bounds the probability that p's output copies c relative to the probability under safe. If p is k_x-NAF safe on prompt x with respect to $\mathcal{C}$ and safe, then for any $c \in \mathcal{C}$:

$$p(\mathsf{SubSim}(c) \mid x) \leq 2^{k_x} \cdot \mathsf{safe}_c(\mathsf{SubSim}(c) \mid x). \tag{1}$$

Hence, bounding k_x prevents copying c, assuming $\mathsf{safe}_c(\mathsf{SubSim}(c)|x)$ is negligibly small (in $|c|$).

Theorem 3.3 (Copy Protection algorithm [VKB23])**.** *Let p be the model returned by* CP *(Algorithm 2 in Appendix B), and q_1 and q_2 be the models returned by* sharded-safe*. Let $k_x \leq -\log\big(1 - d_{\mathrm{TV}}\big(q_1(\cdot|x), q_2(\cdot|x)\big)\big)$. Then p is k_x-NAF for x with respect to $\mathcal{C}$ and* sharded-safe*.*

3.3 Failures of NAF models

We now describe two ways NAF fails to prevent copying, leveraging its lack of protection against multiple prompts (i.e., composition) and data-dependent prompts. In each case, a user reproduces training data $c \in D$ verbatim. In Section 3.3.1, the model generates c when prompted with $\mathsf{ideas}(c)$—a prompt that depends on c but not on any copyrightable expression thereof. In Section 3.3.2, the user issues a fixed sequence of prompts, recovering k bits of the dataset with each query.

3.3.1 CP can regurgitate training data

We show that CP—the main NAF algorithm of [VKB23]—can fail to protect against copying. Using a prompt containing no copyrightable expression, a user can cause the NAF model returned by CP to regurgitate copyrighted training data. This is based on an observation of Thomas Steinke [Ste23].

Observe that CP is a black-box transformation from an underlying training algorithm Train. Thus, CP is really a family of algorithms—one per Train. The following theorem states that there exists Train^* such that the resulting NAF algorithm CP^* fails in the following way. On prompt $x = \mathsf{ideas}(c)$ reproduces training datum c verbatim. We defer the proof of (a slight generalization) of this theorem to Appendix B.2. It uses the training algorithm $\mathsf{Train}^* = \mathsf{Train}_{\mathsf{kv}}$ (Alg. 3) described in Example 5.2.

Theorem 3.4. *For model training algorithm Train^*, let CP^* be the k_x-NAF algorithm from Theorem 3.3, and let $p^* \leftarrow \mathsf{CP}^*$. For all* ideas*, there exists a training algorithm Train^* such that for all datasets D and for all works $c \in D$ whose ideas are distinct in D (i.e., $\forall w \in D, w \neq c : \mathsf{ideas}(w) \neq \mathsf{ideas}(c)$): $p^*(c \mid \mathsf{ideas}(c)) = 1$.*

3.3.2 k-NAF does not prevent full reconstruction

Our next theorem shows that k-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. In more detail, we give a family of models that are k-NAF with respect to pure noise, yet which 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. The proof is deferred to Appendix B.2.

Theorem 3.5. *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 $\mathrm{poly}(L, 1/k)$ black-box queries to $p_{k,D}$ and outputs D with probability > 0.99.*

4 The interaction between user and model

People interact with generative models in complex ways: refining prompts and using results to create a final product whose author is not solely human nor machine. Meaningful copyright protection must cover such cases. To that end, we make the interaction between a user and the model explicit.

Let u be a (randomized) algorithm called the *user*, p be a model, and $\mathsf{aux} \in \{0,1\}^* \cup \{\perp\}$ be an *auxiliary input*. In cryptography, auxiliary inputs allow one to define security guarantees even when an algorithm has instance-specific side information. In our setting, the algorithm is the user, and the side information is the non-copyrightable ideas of an in-copyright work. We denote by $\mathsf{u}^p(\mathsf{aux})$ the

https://doi.org/10.52202/085713-1903

algorithm u run with input aux and black-box access to p. The result is a work $z \in \mathcal{W}$ distributed as $z \leftarrow \mathsf{u}^p(\mathsf{aux})$. Together with a training algorithm Train and a dataset D, this process induces probability measure τ over $\mathcal{W}$.

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

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

The probability is taken over the randomness of the algorithms Train*,* u*, and* p*. Note that* τ *depends on* Train*,* u*, and* D*. The notation elides these dependencies to reduce clutter.*

Legally, the user's output infringes on the copyright of a work $c \in \mathcal{C}$ only if it is substantially similar to c. Preventing substantial similarity prevents infringement. This motivates our first desideratum.

Desideratum 1. *We want* $\tau(\mathsf{SubSim}(\mathcal{C}); \mathsf{aux})$ *to be as small as possible for as many users as possible.*

By considering the user's output distribution, we require protection against multiple prompts (composition) and data-dependent prompts (when aux contains the ideas of work in the training data, as below). This fixes the shortcomings of NAF that enabled the attacks described in Section 3.3.

5 Tainted models enable 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. We say a training algorithm Train is tainted if there is a fixed algorithm u that copies work from the training dataset using only trained model and unprotected ideas. The order of quantifiers is important: the user u is fixed while the dataset D is arbitrary. Such a user cannot possibly be at fault—it cannot know a work in all possible datasets D.

Definition 5.1 (Tainted training)**.** *We say* Train *is* tainted *with respect to* ideas *if there exists a user* u *such that for all datasets* D *and all works* $w \in D$*:* $\tau\big(\mathsf{SubSim}(D);\ \mathsf{ideas}(w)\big) > 0.99$.

If the goal is to prevent copying, being tainted is as bad as it gets. Thus, we get our next desideratum. We will see that NAF fails this test but Definition 7.6 passes.

Desideratum 2. *A meaningful definition of provable copyright protection should bar tainted models.*

As an example, we show that any training algorithm can be made tainted (Example 5.2). This is used to prove Theorem 3.5. Note E.1 presents a negative example: a training algorithm that always returns a single fixed model is not tainted, under the mild assumption that it is possible for two works to express the same ideas so differently that no work is similar to both.

Example 5.2. *Algorithm 3 in the Appendix 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* id *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* $I[x] = D_x$ *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* $\mathsf{u}^p(\mathsf{aux})$ *that returns a sample from* $p(\cdot|\mathsf{aux})$*. Fix* D *and* $w \in D$*, and let* $p \leftarrow \mathsf{Train}_{\mathsf{kv}}(D)$*. By construction, the user always outputs an element of* $D_{\mathsf{ideas}(w)}$*. Therefore,* $\tau(\mathsf{SubSim}(D); \mathsf{ideas}(w)) \geq \tau(D_{\mathsf{ideas}(w)}; \mathsf{ideas}(w)) = 1$.

5.1 Tainted NAF models

Desideratum 2 lays out a necessary condition for provable copyright protection: exclude tainted training algorithms. NAF fails this test. This is captured by the following corollaries of Theorem B.5 (slightly generalizing Theorem 3.4) and Theorem 3.5.

Corollary 5.3 (of Thm. B.5)**.** *For any* ideas*, the NAF algorithm* CP^* *given by Thm. 3.4 is tainted.*

Proof. Consider the user $\mathsf{u}^p(\mathsf{aux})$ that returns a sample from $p(\cdot|\mathsf{aux})$. Fix D and $w \in D$, and let $p^* \leftarrow \mathsf{CP}^*(D)$. By Theorem B.5, $\tau(\mathsf{SubSim}(D);\ \mathsf{ideas}(w)) \geq \tau(D_{\mathsf{ideas}(w)};\ \mathsf{ideas}(w)) = 1$. □

Corollary 5.4 (of Thm 3.5)**.** *For any* k*,* ideas*, there exists a tainted algorithm* Train *such that for all datasets* D*, the model* $p \leftarrow \mathsf{Train}(D)$ *is* k*-NAF with respect to* coin_ℓ*.*

6 Blameless copy protection: a definitional framework

Our goal is to define provable copyright protection. The previous section gives a negative answer: provable copyright protection should bar tainted models. This section and those that follow give a positive answer: provable copyright protection should protect blameless users.

We can't hope to guarantee that a generative model never reproduces copyrighted work, even works it was not trained on. Malicious users can always induce copying. (Formally, if the set of in-copyright works $\mathcal{C}$ is non-empty, there exists u that always infringes: $\tau(\mathsf{SubSim}(\mathcal{C}); \perp) = 1$.) Instead, we wish to protect *blameless* users—who don't themselves induce infringement—from unwitting copying.

This suggests a framework for defining meaningful guarantees, which we call *blameless copy protection*. First, define a class of blameless users $\mathfrak{B}$. Second, guarantee that for blameless users, the probability of copying $\tau(\mathsf{SubSim}(\mathcal{C});\ \mathsf{aux})$ at most some small κ.

Definition 6.1 (Blameless copy protection)**.** *Fix $\kappa > 0$ and a class $\mathfrak{B}$ of* blameless users*. We say* Train *is $(\kappa, \mathfrak{B})$*-copy protective *if for all blameless $\mathsf{u} \in \mathfrak{B}$ and all aux, $D \in \mathcal{W}^*$, $\mathcal{C} \subseteq \mathcal{W}$:*

$$\tau(\mathsf{SubSim}(\mathcal{C});\ \mathsf{aux}) < \kappa.$$

The missing piece is $\mathfrak{B}$: What makes a user blameless? Different answers will yield different versions of blameless copy protection. So Definition 6.1 is less a definition than a framework for definitions. Instantiating it may require additional assumptions. As we cannot perfectly capture legal blamelessness, we should err on the side of protecting more users rather than less. In Section 7, we offer a first instantiation of blameless copy protection, inspired by clean-room design. But there may be very different ways to formalize blamelessness, which we leave for future work. The cryptographic notion of extraction offers an intriguing approach: a user is blameworthy if a copyright-infringing work can be efficiently extracted from the user itself.

7 Defining clean-room copy protection

This section formalizes what it means to have *access* to a copyrighted work (Section 7.1) and instantiates the blameless copy protection framework (Section 7.2)—drawing inspiration from clean-room design. In copyright law, a clean room is "a process of producing a product under conditions guaranteeing independent design and foreclosing the possibility of copying" [Elk90]. The clean room keeps contamination out, like keeping dust out of a semiconductor lab. The idea comes from cases involving reverse engineering. Roughly, a team is given a description of design specs of the product (ideas) but not the product itself (expression), and tasked with producing a compatible product. There is no access, as long as the team itself was not spoiled by prior familiarity with the product or its design. As such, the outcome is constructively non-infringing. Even substantial similarities will be the product of independent creation.

Clean-room design is the inspiration for a counterfactual *clean-room distribution* where a user interacts with a model trained without access to certain in-copyright works (Definition 7.4). We quantify the *blamelessness* of a user as the probability β that the user—in the clean-room distribution—would have produced something that is substantially similar to the excluded works (Definition 7.5). *Clean-room copy protection* provides a corresponding bound κ on the probability of copying any work in the real distribution (Define 7.6), where κ depends on β.

7.1 Scrubbing a dataset removes access

A clean room is a setting where a user does not have access to a particular copyrighted work. To pin this down, we first operationalize the legal concept of access.

The copyright dependency graph We need a way to say whether a work w' is a derivative of another work w a copyright-relevant way. Such dependencies are directed, with later works stemming from earlier works. The *copyright dependency graph* encodes this relation. It is assumed to capture the legal concept of access in the following sense: *For purposes of copyright law, a dataset D only gives access to a work w if there is some $w' \in D$ that stems from w.*

Definition 7.1 (Copyright dependency graph)**.** *The* copyright dependency graph *is a directed graph $G = (\mathcal{W}, E)$ whose edges reflect dependencies relevant for copyright. If $(w, w') \in E$, we say that w'* stems from *w. We assume that every w stems from itself: $(w, w) \in E$ for all w.*

 https://doi.org/10.52202/085713-1903

It is useful to refer to the set of all copyrighted works for which access is or isn't implied by access to a dataset D. We denote these sets by $\mathcal{C}_D$ and $\mathcal{C}_{-D}$, respectively. For the sake of copyright analysis, access to D does not constitute access to any copyrighted $c \in \mathcal{C}_{-D}$.

Definition 7.2 ($\mathcal{C}_D$ and $\mathcal{C}_{-D}$)**.** *We define $\mathcal{C}_D = \{c \in \mathcal{C} : \exists w \in D, (c, w) \in E\}$ as the set of copyrighted works from which any work in D stems. We denote its complement $\mathcal{C}_{-D} = \mathcal{C} \setminus \mathcal{C}_D$.*

The scrub **function** We define the function scrub to operationalize the legal concept of access. It removes from a dataset any work that stems from a target copyrighted work c. That is, any w for which $c \to w$ is an edge in the copyright dependency graph. The result is a new dataset $\mathsf{scrub}(D, c) \subseteq D$. For the sake of copyright analysis, access to $\mathsf{scrub}(D, c)$ does not constitute to access to c.

Definition 7.3 (scrub)**.** *Fix copyright dependency graph $G = (\mathcal{W}, E)$, dataset D and work c. The dataset $\mathsf{scrub}(D, c) = (w \in D : (c, w) \notin E)$ is the sub-dataset of D of works not stemming from c.*

Observe that $\mathcal{C}_D = \{c \in \mathcal{C} : \mathsf{scrub}(D, c) \neq D\}$ and $\mathcal{C}_{-D} = \{c \in \mathcal{C} : \mathsf{scrub}(D, c) = D\}$

7.2 Clean training algorithms: copy protection for blameless users in a clean room

This section gives a clean-room inspired formulation of copy protection. Informally, we say that a training algorithm is (κ, β)-*clean* if for all users u: either (i) u would have copied in a clean-room setting with probability at least β; or (ii) u copies in the real world with probability at most κ. Making this precise, we define the user's *clean-room output distribution* (Definition 7.4), *blameless users in the clean room* (Definition 7.5), and (κ, β)-*clean-room copy protection* (Definition 7.6).

7.2.1 The user's clean-room distribution

We define a user's *clean-room distribution*, a counterfactual to its true distribution τ. For a copyrighted work c, we denote by τ_{-c} 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 τ (Definition 4.1) is that the model is trained on $\mathsf{scrub}(D, c)$ instead of D.

Definition 7.4 (User's clean-room distribution)**.** *For user* u*, training algorithm* Train*, dataset D, work c, and auxiliary information* aux*, we define the user's* clean-room distribution τ_{-c} *for c as:*

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

This is closely related to NAF. The model $p \leftarrow \mathsf{Train}(\mathsf{scrub}(D, c))$ is NAF's safe model safe_c for an appropriately defined function safe. The clean-room distribution τ_{-c} is the distribution over outputs that results when the user given aux interacts with safe_c.

7.2.2 Clean-room blamelessness and copy protection

We postulate that *blameless users can control their risk of copying when using a model trained in a clean room*. Given $\beta > 0$, a blameless user can guarantee that they copy a work c that was "outside the clean room" with probability at most β. This should hold even if the user is exposed to c in some other than through the model. (For a weaker assumption, see Note E.3.) People are constantly exposed to copyrighted works, yet we somehow to manage to avoid copying every day. Using a model trained without access to c shouldn't change that.

We call these users β-*blameless in a clean room*, or β-*blameless* for short. Definition 7.5 formalizes this idea. Recall that $\mathcal{C}_{-D}$ is the set of copyrighted works from which no element of D stems (Definition 7.2). Thus, the set of works "outside the clean room" is $\mathcal{C}_{-D} \cup \{c\}$.

Definition 7.5 (Blameless in the clean room (β-blameless))**.** *For $0 \leq \beta \leq 1$, a user* u *is* β-blameless in the clean room (β-blameless) *with respect to D, $\mathcal{C}$,* Train *if for all $c \in \mathcal{C}$ and all* aux*:*[1]

$$\tau_{-c}\Big(\mathsf{SubSim}\big(\mathcal{C}_{-D} \cup \{c\}\big);\ \mathsf{aux}\Big) \leq \beta.$$

Instantiating our framework (Definition 6.1), we now define *clean training algorithms* as those that provide copy protection to users who are blameless in the clean room.

[1] Equivalently, $\mathfrak{B}_\beta^{\mathsf{clean}}(D, \mathcal{C}, \mathsf{ideas}) := \{\mathsf{u} : \forall c \in \mathcal{C}, \forall \mathsf{aux}, \tau_{-c}\big(\mathsf{SubSim}\big(\mathcal{C}_{-D} \cup \{c\}\big);\ \mathsf{id}\big) \leq \beta\}$.

Definition 7.6 (Clean-room copy protection; (κ, β)-clean)**.** *For $\kappa, \beta > 0$, we say* Train *is* (κ, β)-clean *if for all $\mathcal{C} \subseteq \mathcal{W}$, $D \in \mathcal{W}^*$, all users* u *who are β-blameless (w.r.t. $\mathcal{C}, D,$* Train*), and all* aux*:*

$$\tau(\mathsf{SubSim}(\mathcal{C});\ \mathsf{aux}) \leq \kappa.$$

If this only holds for datasets D in an admissible set $\mathfrak{D} \subseteq \mathcal{W}^*$, we say Train *is* (κ, β)-clean *for* $\mathfrak{D}$.

Remark 7.7 (How small is β?)**.** We think of β as the probability that a "truly blameless" user (whatever that means) nevertheless produces something substantially similar to a copyrighted work. There is a limit to how small this can be. Even a monkey on a typewriter—truly blameless if anyone is—has a non-zero β. We conjecture that $\beta \approx 10^{-6}$ or 10^{-9} is achievable by a conscientious user.

7.2.3 Clean-room copy protection is not tainted

By Desideratum 2, a good definition of provable copy protection should exclude tainted training algorithms. Theorem D.1 shows that clean-room copy protection passes this test, under a reasonable assumption on ideas and SubSim. Roughly, the assumption is that there exist at least n distinct ideas each of which can be expressed in $m \gg n$ ways satisfying a strong dissimilarity property: that for any distinct w and w', the set of works substantially similar to both w and w' is empty.

8 Differential privacy's protection against copying

Many works have suggested that differential privacy [DMNS06] (DP) should protect against copying if the training data are deduplicated [BLM20, HLJ+23, VKB23, EKHLM24, CKOX24, LMNP24]. This section turns the suggestion into a theorem. For background on DP, see Appendix C.

At a high level, we show that DP training blameless users from copyright infringement (in the sense of Definition 7.6) if the training dataset is *golden*, a copyright-deduplication condition defined below. If so, the probability of copying is at most $\approx e^{\varepsilon}\beta N_D$, where β is a user's probability of copying in the clean room and $N_D = |\mathcal{C}_D|$ is the number of copyrighted works to which D grants access. To guarantee the risk is at most κ, the user sets $\beta \approx \kappa / e^{\varepsilon} N_D$. The linear dependence on N_D is not ideal, but may sometimes be good enough or improved (Note E.4.)

This allows a user choose their appetite for copyright risk κ and then act to set β accordingly. Stephen King writing his next bestseller has a much lower risk tolerance ($\kappa = 10^{-9}$) than Joe Schmoe writing a wedding toast ($\kappa = 10^{-1}$). Most users are somewhere in between. For $\varepsilon = 5$ and $N_D = 200,000$ Stephen King needs $\beta \approx 10^{-15}$ while Joe Schmoe only needs $\beta \approx 10^{-6}$. If $\beta \approx 10^{-15}$ is too small (Remark 7.7), Stephen King can instead forgo using the LLM altogether.

A dataset D is *golden* if at most one item $w \in D$ stems from any copyrighted work c.[2] That item could be c itself or a derivative work. No protected original expression appears in more than one element of a golden dataset. For example, it can contain one copy or parody of the *Abbey Road* album cover (not both), and many parodies of the out-of-copyright *Mona Lisa*. Note E.5 explains why golden datasets will typically remain golden as the set of in-copyright works evolves over time.

Definition 8.1 (Golden dataset)**.** *Let $G = (\mathcal{W}, E)$ be a copyright dependency graph, and $\mathcal{C} \subseteq \mathcal{W}$ be the set of in-copyright works. A dataset D is* golden *(with respect to G, $\mathcal{C}$) if for all $c \in \mathcal{C}$ there is at most one $w \in D$ such that $(c, w) \in E$.*

Theorem 8.2. *Let* Train *be (ε, δ)-differentially private for $\varepsilon > 0, \delta \geq 0$. Let $N_D = |\mathcal{C}_D|$. Then* Train *is (κ, β)-clean for golden datasets D, for all $\beta \geq 0$ and $\kappa \geq (e^{\varepsilon} N_D + 1)\beta + N_D \delta$.*

Proof. Fix $\mathcal{C} \subseteq \mathcal{W}$, D, and user u that is β-blameless (Def. 7.5). We must show that for all aux, $\tau(\mathsf{SubSim}(\mathcal{C});\ \mathsf{aux}) \leq \kappa$. Because D is golden, $|D \setminus \mathsf{scrub}(D, c)| \leq 1$. Hence, datasets $\mathsf{scrub}(D, c)$ and D are either equal or neighboring for all $c \in \mathcal{C}$. Applying DP and post-processing, we have $\tau(E) \leq e^{\varepsilon} \cdot \tau_{-c}(E) + \delta$ for all events E and all $c \in \mathcal{C}$.

$$\begin{aligned}
\tau(\mathsf{SubSim}(\mathcal{C});\ \mathsf{aux}) &\leq \tau(\mathsf{SubSim}(\mathcal{C}_{-D});\ \mathsf{aux}) + \sum_{c \in \mathcal{C}_D} \tau(\mathsf{SubSim}(c);\ \mathsf{aux}) && \text{(union bound)} \\
&\leq \beta + N_D \delta + e^{\varepsilon} \cdot \sum_{c \in \mathcal{C}_D} \tau_{-c}(\mathsf{SubSim}(c);\ \mathsf{aux}) && \text{(DP; Prop. 8.3 below)} \\
&\leq (e^{\varepsilon} N_D + 1)\beta + N_D \delta \leq \kappa && (\beta\text{-blameless}) \quad \square
\end{aligned}$$

[2] The term is borrowed from [VKB23] who use it only informally and with a different meaning.

https://doi.org/10.52202/085713-1903

In the proof above, we need to bound the probability $\tau(\mathsf{SubSim}(\mathcal{C}_{-D});\ \mathsf{aux})$ of similarity with some work in $\mathcal{C}_{-D}$ in the real distribution τ. Blamelessness only gives us a bound in the clean-room distribution τ_{-c}. Proposition 8.3 shows that the latter implies the former.

Proposition 8.3. *Let* u *be* β*-blameless w.r.t.* D, $\mathcal{C}$, Train. *For all* aux*:* $\tau\left(\mathcal{C}_{-D};\ \mathsf{aux}\right) \leq \beta$.

Proof. By definition of $\mathcal{C}_{-D}$, $\mathsf{scrub}(D, c) = D$ and hence $\tau_{-c} = \tau$. Also, $\mathcal{C}_{-D} = \mathcal{C}_{-D} \cup \{c\}$. Thus, $\tau(\mathcal{C}_{-D}; \mathsf{aux}) \leq \tau_{-c}\left(\mathsf{SubSim}\left(\mathcal{C}_{-D} \cup \{c\}\right); \mathsf{aux}\right) \leq \beta$, with the last inequality by blamelessness. □

9 Discussion: provable copyright protection in practice?

What if copying occurs? People copy without generative models, and will continue copying with them. One way to evaluate the usefulness of a copyright-mitigation measure is to consider what happens when copying does occur. Who should pay?

It is hard to assign culpability on the basis of NAF alone, as tainted NAF models can induce a user acting appropriately to infringe. Clean-room copy protection gives a clearer theoretical account of who is culpable if training is differentially private. The provider is culpable if the data wasn't golden; the user is culpable if not blameless; possibly both are culpable, or neither. However, this test is impossible to apply as blamelessness cannot in general be checked.

Even so, a simple indemnification rule is possible: The model provider pays for infringement if the copied work appears too often in the training data. This is a question of fact that legal process and forensic experts are well-suited to resolve. Experts for each party can examine the training data and testify before the finder of fact. Our indemnification rule gives users what they want. Users are indemnified when there is any possibility that the model is to blame, and they can control their risk tolerance by tuning β. The rule also gives model providers what they want. Besides offering a valuable protection for users, the provider can trade off their overall exposure to copyright penalties with their effort spent cleaning the data. Perhaps the provider can even pass the liability to third-party data providers contracted to provide golden data. Many generative AI services already indemnify their users against copyright liability, but impose restrictions that could require a complex forensic analysis of the creation of the infringing artifact to decide whether the indemnification applies. In contrast, our indemnification rule imposes no restrictions on the user and reduces the question of liability to a clear question of fact.

How reasonable are the requirements for clean room protection? Theorem 8.2 requires DP models, golden data, and blameless users. Are these requirements reasonable?

Constructing a golden dataset large enough to pre-train a foundation model is hopeless. Making a golden dataset is much harder than deduplication (already a major challenge [LIN+21]): it depends on squishy legal standards and on data outside the dataset. Fine-tuning is much more promising. Even with differential privacy, tens of thousands of training data can suffice to train useful models. Creating a golden dataset with tens of thousands of items seems feasible but expensive. One only needs to be able to determine if two items stem from a common in-copyright work (i.e., siblings in the copyright dependency graph). For a given pair of items, doing so with confidence is plausible with appropriate domain expertise. Another approach would be to create or commission new work with known copyright dependencies. There are non-technical ways to address the challenge of golden datasets. Suppose the trainer license the data from a data provider. They could require metadata to include copyright dependencies, or that the data provider provide golden data. Either way, 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 the data is imperfect. The guarantees will still hold for any work satisfying the golden condition.

Blamelessness presents a greater challenge: it is not checkable, not even by the user. Still, we believe that diligent users who are attentive to the possibility of inadvertent infringement can guarantee β-blamelessness for β small enough to make Theorem 8.2 meaningful. This is the crux of the matter. Why do I 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. That I could productively use ChatGPT-4o to produce a story or image that is wholly original, bearing no resemblance even to stories and images with which I am intimately familiar. People are able to be truly creative, despite constant exposure to copyrighted work. Somehow you and I manage to avoid copying every day.

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. Theorem 3.4 is based on an observation of Thomas. We thank Sarah Scheffler, Randy Picker, Lior Strahilevitz, James Grimmelmann, anonymous reviewers, and 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

[BLM20] Olivier Bousquet, Roi Livni, and Shay Moran. Synthetic data generators–sequential and private. *Advances in Neural Information Processing Systems*, 33:7114–7124, 2020.

[CKOX24] Wei-Ning Chen, Peter Kairouz, Sewoong Oh, and Zheng Xu. Randomization techniques to mitigate the risk of copyright infringement. *arXiv preprint arXiv:2408.13278*, 2024.

[CLG+23] A Feder Cooper, Katherine Lee, James Grimmelmann, Daphne Ippolito, Christopher Callison-Burch, Christopher A Choquette-Choo, Niloofar Mireshghallah, Miles Brundage, David Mimno, Madiha Zahrah Choksi, et al. Report of the 1st workshop on generative ai and law. *arXiv preprint arXiv:2311.06477*, 2023.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006.

[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`.

[Elk90] David S Elkins. NEC v. Intel: A guide to using "clean room" procedures as evidence. *10 Computer L.J. 453*, 1990.

[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`.

[LIN+21] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*, 2021.

[LMNP24] Roi Livni, Shay Moran, Kobbi Nissim, and Chirag Pabbaraju. Credit attribution and stable compression. *Advances in Neural Information Processing Systems*, 37:2663–2685, 2024.

[LSK24] Xiang Li, Qianli Shen, and Kenji Kawaguchi. Va3: Virtually assured amplification attack on probabilistic copyright protection for text-to-image generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12363–12373, 2024.

 https://doi.org/10.52202/085713-1903

[Ste23] Thomas Steinke. Personal communication, 2023.

[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.

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

A 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. The claim that "copyright can be reduced to privacy" is a straw man. 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 possibility that one work's copyrighted expression appears in many other works.

B Deferred material on near access-free models (Section 3)

This section presents our detailed treatment of near access-free models. It describes near access-freeness and its limitations. We prove that models can enable verbatim copying while still satisfying NAF. While NAF provides protection against a single prompt that is independent of the training data, it makes no guarantees against many prompts [LSK24], nor a single prompt derived from non-copyrightable ideas (not expression).

To make this section self-contained, we repeat material from Section 3, quoting freely.

B.1 Definitions and main results from [VKB23]

NAF is defined with respect to a function safe. The safe function maps a copyrighted data point $c \in \mathcal{C}$ to a generative model $\mathsf{safe}_c \in \mathcal{P}$ trained without access to c. An example safe function is $\mathsf{sharded\text{-}safe}$. We simplify the notation of [VKB23] by fixing the divergence to maximum KL divergence.

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

Definition B.2 (k_x-NAF [VKB23])**.** *Fix a set $\mathcal{C}$ and function $\mathsf{safe} : \mathcal{C} \to \mathcal{P}$. 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 $k_x \leq k$.*

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.

 https://doi.org/10.52202/085713-1903

Lemma B.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 $E \subseteq \mathcal{Y}$:*

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

Letting $E = \mathsf{SubSim}(c)$ be the event that y is substantially similar to c, we get

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

As copying requires substantial similarity, bounding k_x suffices to prevent violation whenever $\mathsf{safe}_c(\mathsf{SubSim}(c)|x)$ is negligibly small. Heuristically, we expect $\mathsf{safe}_c(\mathsf{SubSim}(c)|x)$ to be negligible in $|c|$. It may sometimes be large, e.g., when x contains the copyrighted work c [VKB23].

Theorem 3.3 presents the main feasibility result of [VKB23]: any training algorithm Train can be used as a black-box to construct an NAF model CP, short for *copy protection*.

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

$$k_x \leq -\log\Big(1 - d_{\mathrm{TV}}\big(q_1(\cdot|x), q_2(\cdot|x)\big)\Big). \tag{2}$$

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

Algorithm 1: sharded-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

Algorithm 2: CP: Copy-Protection [VKB23]

Input: Dataset D
Learn: Run $\mathsf{sharded\text{-}safe}(D)$ to obtain q_1, q_2 as in Algorithm 1
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.2 Failures of NAF models

B.2.1 CP can regurgitate training data

We show that CP—the main NAF algorithm of [VKB23]—can fail to protect against copying. Using a prompt containing no copyrightable expression, a user can cause the NAF model returned by CP to regurgitate copyrighted training data. This is based on an observation of Thomas Steinke [Ste23].

The following claim is a slight generalization of Theorem 3.4. In words, instantiating $\mathsf{CP} = \mathsf{CP}_{\mathsf{kv}}$ with the (tainted) training algorithm $\mathsf{Train}_{\mathsf{kv}}$ (Algorithm 3) described in Example 5.2, causes training data regurgitation.

Note 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 $p(y|x) > 0$.

Theorem B.5. *Let D be a dataset. 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*

Algorithm 3: $\mathsf{Train}_{\mathsf{kv}}$ (for Example 5.2 and Theorem B.5)

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

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

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 sampled uniformly from W;
    else return y sampled from $q_0(\cdot|x)$;

Result: q_{kv}

in Algorithms 3, 1, and 2, defined with respect to Train_0 *(and the preceding algorithms). Let* $p \leftarrow \mathsf{CP}_{\mathsf{kv}}(D)$. *Then for all* $w \in D$*:*

$$p\left(D_{\mathsf{ideas}(w)} \mid \mathsf{ideas}(w)\right) = 1.$$

In particular, for any copyrighted work $c \in D$ whose unique ideas are unique, we get

$$p(c \mid \mathsf{ideas}(c)) = 1.$$

Theorem B.5 does not contradict the CP theorem (Theorem 3.3). This is because CP is only k_x-NAF, where k_x depends on the prompt x. In our construction, the bound on k_x given by Theorem 3.3 is not vacuous for $x \in \mathsf{ideas}(D)$.

Proof of Theorem B.5. 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 $w \in D_1$. Let $\mathsf{id} = \mathsf{ideas}(w)$.

By construction, $q_1(D_{\mathsf{id}}|\mathsf{id}) = 1$ (as in Example 5.2). Thus, for all outputs $y \in \mathcal{W}$:

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

The distribution $p(\cdot|\mathsf{id})$ is well-defined if there exists $y \in D_{\mathsf{id}}$ such that $q_2(y|\mathsf{id}) > 0$. The model $q_2(\cdot|\mathsf{id})$ returns an element of $D_2 \cap D_{\mathsf{id}}$ 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, $q_2(y|\mathsf{id}) > 0$ for all $y \in D_2 \cap D_{\mathsf{id}}$.

The claim follows immediately:

$$p(D_{\mathsf{id}} \mid \mathsf{id}) = 1 - p(\overline{D}_{\mathsf{id}} \mid \mathsf{id}) = 1. \qquad \square$$

B.2.2 k-NAF does not prevent full reconstruction

Our next theorem shows that k-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.

We give a family of models that are k-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.

Theorem (Theorem 3.5)**.** *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* $\ell = \max\{1, \lfloor k \rfloor\}$.

 https://doi.org/10.52202/085713-1903

- *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*.*

Remark B.6. The theorem and proof can be adapted to more realistic safe by encoding D in the bias of a hash of p's outputs. But the added complexity would obscure the technical idea used in the proof: biasing safe can reveal D without violating NAF.

Proof of Theorem 3.5. We parse D as a bit string of length L, and let $D[j]$ be its jth bit. We prove the result separately for $k \geq 1$ and $0 < k < 1$.

For $k \geq 1$, Train outputs the model $p_{k,D}$ as follows:

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

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 $\beta = 2^k - 1$ and outputs the model $p_{k,D}$ as follows:

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

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

To determine $D[j]$ with greater than $> 1 - \frac{1}{100L}$, the user u makes $\mathrm{poly}(L, \log 1/\beta) = \mathrm{poly}(L, 1/k)$ 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. □

B.3 *VA3*: an empirical attack on NAF [LSK24]

Li, Shen, and Kawaguchi propose and empirically evaluate a "Virtually Assured Amplification Attack" against NAF [LSK24]. At a high level, [LSK24] shows that an attacker, interacting with an NAF model in a black-box manner, can reliably induce a model to produce outputs that infringe on a target work c^*. They also give a white-box prompt writing algorithm for diffusion models (Anti-NAF) that greatly improves the performance of their attacks.

Overall, the work provides good evidence that the algorithms proposed by [VKB23] may not prevent infringement. Some uncertainty remains because the algorithm implemented in [LSK24] deviates from the original, as explained below. The paper leaves open whether k-NAF (with fixed k) prevents copyright infringement, formalizes a flawed attack model, and avoids the underlying definitional questions almost entirely.

We now describe [LSK24] in more detail, offering a somewhat different interpretation than the original.

The paper's main focus is the Amplification Attack. Given a target copyrighted work c^*, an attacker repeatedly queries a model with some prompt $x = x(c^*)$. It returns the generation most similar to c^*. This process amplifies the one-shot probability of infringement. For example, from 0.40% to 13.64% when x is the original caption of c^* in the training data, or from 8.52% to 77.36% using the Anti-NAF prompt generator. In our view, this not actually the paper's main negative result for NAF.

More significant (for our purposes) is the existence of prompts x whose one-shot probability of infringement is non-negligible: 0.40% or 8.52% in the previous example. The message is similar to our Theorem B.5. Namely, that the NAF constructions of [VKB23] admit models for which prompts x derived from non-copyrightable aspects of c^* can produce infringing generations. Based on the top half of [LSK24, Table 1], the prompts trivialize NAF's protection by making $k_x \approx \log(1/\mathsf{safe}(\mathsf{SubSim}(c^*) \mid x))$.

Though the paper does not speculate, we suspect that the mechanism for the failure is similar to our construction in Theorem B.5. The model, repeatedly fine-tuned on c^*, regurgitates c^* with high enough probability that it survives the reweighting of CP and $\mathsf{CP\text{-}k}$ [LSK24, Fig. 8].

Now we turn to the paper's limitations.

First, the attack model has a conceptual flaw, highlighting the need for our definitions-first approach. The attacker knows c^* and is actively trying to induce a similar generation [LSK24, Sec. 4.1]. Indeed, the Amplification Attack *requires* knowing c^*. This setup allows trivial attacks, like prompting `return` c^*. Such an attack would not be meaningful. Any infringement would be the users' fault, not something we care to prevent. Still, the flaw in the attack model doesn't affect the paper's experiments. Because they use a text-to-image model, he prompts used in the attack are just a few words long and contain nothing copyrightable.

Second, the results say nothing about k-NAF, where a fixed k upper-bounds $k_x \leq k$ for every prompt x. At best, the experiments are negative results for the particular k_x-NAF algorithm CP-k given in [VKB23], where k_x depends on x. But as we explain next, it is not entirely clear. (Our Theorems 3.4 and 3.5 are for k_x-NAF and k-NAF algorithms, respectively.)

Third, the algorithm in [LSK24] deviates from CP-k in an important way. As originally defined, CP-k is a rejection-sampling version of CP (Algorithm 2). Given a model p, a safe model safe, and constant k, prompt x, and generation y, let $\rho(y|x) = \log\left(p(y|x)/\mathsf{safe}(y|x)\right)$. Oversimplifying, CP-k repeatedly samples $y \leftarrow p(\cdot|x)$ until $\rho(y|x) \leq k$, and returns the final sample. [VKB23] proves that CP-k is k_x-NAF for $k_x = k + \log(1/\nu(x))$, where $\nu(x) = \Pr_y[\rho(y|x) \leq k]$.

[LSK24]'s implementation differs. Instead of fixing k, they fix $\nu(x)$. The experiments sample many generations $y \leftarrow p(\cdot|x)$, and return the $\nu(x) = 5\%, 10\%, \ldots$ with smallest $\rho(y|x)$. This strikes us as a meaningful difference. At a minimum, the CP-k theorem doesn't apply as is. Despite the gap, we view the results of [LSK24] as strong evidence of weaknesses of CP-k. We conjecture that there is a value of k_x for which the implemented version achieves k_x-NAF with high probability, but did not attempt to prove it.

B.4 On NAF's legal relevance

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). Its 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 it's 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 miniscule. Hence, substantial similarity between the defendant's and plaintiff's works is exceedingly unlikely.

Elkin-Koren et al. [EKHLM24] correctly argue that copyright cannot be "reduced to privacy." However, this is a straw man of version of [VKB23] and the present paper (see Appendix A for additional discussion).

The sharpest criticism is by Lee, Cooper, and Grimmelman who argue that NAF is simply wrong on the law [LCG24]. "[NAF] is explicitly inspired by copyright's concept of access, but copyright law itself does not work that way. Just as two authors can independently create identical works and each hold a copyright in theirs, 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."

We agree 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

 https://doi.org/10.52202/085713-1903

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.[3] 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.

C Differential privacy background [DMNS06]

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 C.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 C.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:

$$\Pr[f(M(D), \mathsf{aux}) \in S] \leq e^{\varepsilon} \cdot \Pr[f(M(D', \mathsf{aux})) \in S] + \delta. \tag{3}$$

D Clean-room copy protection is not tainted

Theorem D.1. *Fix $\beta < 1$, $n \geq 1$, and $m \geq \frac{n}{\beta}$. Suppose there exists $W = (w_{i,j}) \in \mathcal{W}^{m \times n}$ such that for all $i \neq i' \in [m]$ and all $j \in [n]$:*

$$\mathsf{ideas}(w_{i,j}) = \mathsf{ideas}(w_{i',j}) \quad \textit{and} \quad \mathsf{SubSim}(w_{i,j}) \cap \mathsf{SubSim}(w_{i',j}) = \emptyset.$$

If Train *be tainted with respect to* ideas*, then* Train *is not (κ, β)-clean.*

The proof uses Lemma D.2, stated below.

Proof of Theorem D.1. Fix ideas and let Train be tainted with respect to ideas. Let u be the user guaranteed by taintedness of Train, and τ its output distribution. We must show Train is not (κ, β)-clean. Namely, that there exist $\tilde{u}, \tilde{\mathcal{C}}$, and $\tilde{D}$ for which (i) $\tilde{u}$ is β-blameless, and (ii) there exists $\tilde{\mathsf{aux}}$ such that $\tilde{\tau}(\mathsf{SubSim}(\tilde{\mathcal{C}});\ \tilde{\mathsf{aux}}) \geq \kappa$. Here, $\tilde{\tau}$ is $\tilde{u}$'s output distribution.

Let $\mathsf{id} = \mathsf{ideas}(w_{1,1})$. Define $\tilde{\mathsf{u}} = \mathsf{u}_{\mathsf{id}}$ to be the user with id hard-coded that simulates $\mathsf{u}(\mathsf{id})$, always ignoring its auxiliary input aux. By construction, $\tilde{\tau}(E; \mathsf{aux}) = \tau(E; \mathsf{id})$ for all aux and all events E. That is, the event E is independent of aux. Because $\tilde{\mathsf{u}}$ ignores its auxiliary input, we can invoke Lemma D.2. Thus, there exists $\tilde{D}$ such that $\tilde{\mathsf{u}}$ is β-blameless with respect to $\tilde{D}$, $\tilde{\mathcal{C}} = \tilde{D}$, and Train (condition (i) above). Moreover, $w_{i,1} \in \tilde{D}$ for some $i \in [n]$.

Let $\mathsf{id}' = \mathsf{ideas}(w_{i,1})$. By hypothesis, $\mathsf{id} = \mathsf{id}$ and therefore $\tilde{\mathsf{u}} = \mathsf{u}_{\mathsf{id}'}$. Taintedness of Train implies condition (ii) above:

$$\tilde{\tau}(\mathsf{SubSim}(\mathcal{C}); \perp) = \tau(\mathsf{SubSim}(\mathcal{C}); \mathsf{id}') = \tau(\mathsf{SubSim}(D); \mathsf{id}') > 0.99 \geq \kappa. \qquad \square$$

Lemma D.2. *Fix $\beta < 1$, $n \geq 1$, and $m \geq \frac{n}{\beta}$. Suppose there exists $W = (w_{i,j}) \in \mathcal{W}^{m \times n}$ such that for all $i \neq i' \in [m]$ and all $j \in [n]$: $\mathsf{SubSim}(w_{i,j}) \cap \mathsf{SubSim}(w_{i',j}) = \emptyset$. Let* u *be a user that ignores its auxiliary input: $\mathsf{u}^p(\mathsf{aux}) = \mathsf{u}^p(\perp)$ for all* aux*. For all* Train*, there exists $x \in [m]^n$ such that* u *is β-blameless with respect to $D^{(x)} = \left(w_{x_j,j}\right)_{j \in [n]}$, $\mathcal{C} = D^{(x)}$, and* Train*.*

[3] "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`.

Proof of Lemma D.2. Because u ignores aux, we omit it throughout. Take $\mathcal{C} = D$, which implies $\mathcal{C}_{-D} = \emptyset$. Hence, user u is β-blameless if for all $w \in D$: $\tau_{-w}(\mathsf{SubSim}(w)) \le \beta$.

Let $D^{(x)}_{-j} = \mathsf{scrub}(D^{(x)}, w_{x_j,j}) := (w_{x_{j'},j'})_{j' \neq j}$. (For a counter example, we can choose the copyright dependency graph and hence scrub.) Define

$$p_j(x) = \tau^{(x)}_{-w_{x_j,j}}(\mathsf{SubSim}(w_{x_j,j})) = \Pr_{\substack{p \leftarrow \mathsf{Train}(D^{(x)}_{-j}) \\ z \leftarrow \mathsf{u}^p(\perp)}}[z \in \mathsf{SubSim}(w_{x_j,j})].$$

We must show that:

$$\exists x \in [m]^n, \forall j \in [n]: \quad p_j(x) \le \beta. \tag{$\star$}$$

For $x \in [m]^n$, define $j^*(x) = \mathrm{argmax}_{j \in [n]} p_j(x)$. Define the sets $Z_j = \{x \in [m]^n : j^*(x) = j\}$. One of the Z_j contains at least m^n/n distinct strings. Moreover, there is a subset $X^* \subseteq Z_j$ of at least $(m^n/n)/m^{n-1} = m/n$ strings that agree on all but the jth coordinate.

Summarizing, the set X^* satisfies three properties. (1) For all $x, x' \in X^*$, $j^*(x) = j^*(x') =: j^*$. (2) All $x, x' \in X^*$ disagree at the $j^*(x)$th coordinate, and agree on all other coordinates. (3) $|X^*| \ge m/n$.

By (2), $D^{(x)}_{-j^*} = D^{(x')}_{-j^*}$ for all $x, x' \in X^*$. It follows that $\tau^{(x)}_{-w_{x_{j^*},j^*}} = \tau^{(x')}_{-w_{j^*},x'_{j^*}} =: \tau^*$ for all $x, x' \in X^*$. Consider the events $E_x = \mathsf{SubSim}(w_{x_{j^*},j^*})$ for $x \in X^*$. By hypothesis and by (2), these events are disjoint. Hence, $\sum_{x \in X^*} \tau^*(E_x) \le 1$. By (3), there exists $x \in X^*$ such that $p_{j^*}(x) = \tau^*(E_x) \le n/m \le \beta$. By (1) and definition of $j^*(x)$, we have for all $j \in [n]$, $p_j(x) \le p_{j^*}(x) \le \beta$. This proves ($\star$) and completes the proof. □

E Deferred remarks

Note E.1 (A fixed model is not tainted)**.** The following claim states that any training algorithm that always returns a single fixed model is not tainted, under the mild assumption that it is possible for two works to express the same ideas so differently that no work is similar to both.

Claim E.2. *Suppose there exists a pair of works $w_0, w_1 \in \mathcal{W}$ such that $\mathsf{ideas}(w_0) = \mathsf{ideas}(w_1)$ and $\mathsf{SubSim}(w_0) \cap \mathsf{SubSim}(w_1) = \emptyset$. Then for any fixed model q, the constant algorithm $\mathsf{Train}_q(D) = q$ is not tainted.*

Proof. 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 $\tau^*(w_i) \le 1/2$ for one of $i = 0, 1$. Equation (**??**) is violated for the corresponding D_i. □

Note E.3 (Blameless users' auxiliary information)**.** One can weaken this assumption (making more users blameless) by restricting aux in Definitions 7.5 and 7.6. For example, by allowing aux to include $\mathsf{ideas}(c)$ but not original expression of works in $\mathcal{C}_{-D} \cup \{c\}$. This would undermine Desideratum 1, yielding qualitatively weaker protection. DP would still suffice (Theorem 8.2) with this change.

Note E.4 (On the linear dependence on N_D)**.** Notice that $\kappa > N_D \cdot (\beta + \delta)$ grows linearly with $N_D = |\mathcal{C}_D|$. This only yields a meaningful bound if $N_D \ll 1/(\beta + \delta)$. Along with the difficulty of creating enormous golden datasets (Section 9), this is another reason that our approach is most practical when only medium-sized datasets are required (i.e., fine-tuning).

On the other hand, the factor of $N_D = |\mathcal{C}_D|$ from the union bound is unreasonably pessimistic. Ideas sometimes differ so greatly that the risk of substantial similarity is essentially 0: a photorealistic bird will never resemble Dr. Seuss's Cat in the Hat. If the user is targeting a fixed set of ideas id (conditioning τ on id), one might instead take the union bound over $\mathcal{C}^{\mathsf{id}}_D = \{c \in \mathcal{C}_D : \mathsf{id} \in \mathsf{ideas}(\mathsf{SubSim}(c))\}$. We expect that typically $N^{\mathsf{id}}_D := |\mathcal{C}^{\mathsf{id}}_D| \ll |\mathcal{C}_D| = N_D$.

https://doi.org/10.52202/085713-1903

Note E.5 (Golden datasets remain golden over time)**.** Theorem 8.2 states that (ε, δ)-differentially private models are (κ, β)-clean for *golden datasets* (and appropriate κ and β). The set of golden datasets $\mathfrak{D}$ depends on the set of in-copyright works $\mathcal{C}$. For example, every D is golden when nothing is in-copyright ($\mathcal{C} = \emptyset$).

This presents a challenge: Will a golden dataset remain golden as the set of works protected by copyright evolves? As copyrights expire and new works are created, the sets of in-copyright works today $\mathcal{C}$ and tomorrow $\mathcal{C}'$ will diverge. For Definition 7.6 to offer lasting protection, we need $\mathfrak{D} \subseteq \mathfrak{D}'$.

Fortunately, it will typically be true that datasets remain golden as the set of in-copyright works evolves. This requires three observations. First, expiring copyrights don't affect goldenness. If D is golden for $\mathcal{C}$, then D is golden for $\mathcal{C} \cap \mathcal{C}'$. Second, a work is afforded copyright protections as soon as it is fixed in a tangible medium. This means that if a work currently exists will ever be in-copyright (in $\mathcal{C}'$), then it already in-copyright (in $\mathcal{C}$). Hence, $\mathcal{C}' \setminus \mathcal{C}$ contains only works that do not yet exist. The exception is when a work becomes newly eligible for copyright protections, and the change in eligibility is applied retroactively. For example, if a court extends copyright to a new class of works. Third, no work can stem from a work created later. Combined with the previous observation, D is golden with respect to $\mathcal{C}' \setminus \mathcal{C}$. Putting it all together, we have that $D \in \mathfrak{D}$ implies that D is golden with respect to $(\mathcal{C} \cap \mathcal{C}') \cup (\mathcal{C}' \setminus \mathcal{C}) = \mathcal{C}'$. That is, $D \in \mathfrak{D}'$.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The introduction includes an overview of the organization of the paper with pointers to all the main claims.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In addition to fully stating assumptions throughout, particular discussion of limitations are in Section 3.1,Section 9, and Appendix A,

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

https://doi.org/10.52202/085713-1903

Answer: [Yes]

Justification: Full assumptions and proofs of all theoretical results are included in the body or appendix. (And if I thought the proofs were incorrect, you wouldn't be reading this.)

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [NA]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [NA]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

 https://doi.org/10.52202/085713-1903

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [NA]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The various provisions in the Code either don't apply to this very theoretical work, or are addressed in the next question.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Based on the examples in the Guidelines below, this very theoretical paper does not present the sort of societal impacts this question is asking about. Of course, how copyright law applies to AI has great potential to impact society. Even so, any societal impacts of this paper would be tenuous.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

 https://doi.org/10.52202/085713-1903

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

https://doi.org/10.52202/085713-1903