Counterfactual Explanation at Will, with Zero Privacy Leakage

SHUAI AN, University of Edinburgh, UK
YANG CAO, University of Edinburgh, UK

While counterfactuals have been extensively studied as an intuitive explanation of model predictions, they still have limited adoption in practice due to two obstacles: (a) They rely on excessive access to the model for explanation that the model owner may not provide; and (b) counterfactuals carry information that adversarial users can exploit to launch model extraction attacks. To address the challenges, we propose CPC, a data-driven approach to counterfactual. CPC works at the client side and gives full control and right-to-explain to model users, even when model owners opt not to. Moreover, CPC warrants that adversarial users cannot exploit counterfactuals to extract models. We formulate properties and fundamental problems underlying CPC, study their complexity and develop effective algorithms. Using real-world datasets and user study, we verify that CPC does prevent adversaries from exploiting counterfactuals for model extraction attacks, and is orders of magnitude faster than existing explainers, while maintaining comparable and often higher quality.

CCS Concepts: • **Information systems** → **Middleware for databases**; • **Computing methodologies** → **Machine learning**.

Additional Key Words and Phrases: In-database Explanation; Database for Explainable Machine Learning; Model Explainability; Counterfactual; Privacy; Model Extraction

ACM Reference Format:
Shuai An and Yang Cao. 2024. Counterfactual Explanation at Will, with Zero Privacy Leakage. *Proc. ACM Manag. Data* 2, 3 (SIGMOD), Article 130 (June 2024), 29 pages. https://doi.org/10.1145/3654933

1 INTRODUCTION

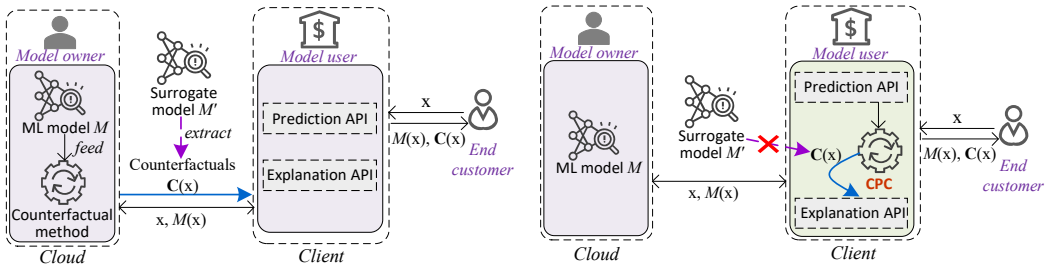
Machine Learning (ML) is increasingly prevalent in decision-making across high-stakes fields such as financial analysis [39], healthcare diagnosis [63] and policy making [27]. This increase underscores the importance and necessity of interpreting complex ML models, to ensure that stakeholders can thoroughly understand model decisions before taking any actions [74, 85, 96]. One particularly useful interpretation is *counterfactual explanation* [16, 31, 38, 85, 93, 114].

Counterfactual explanation indicates how features of a given instance need to be changed to achieve a different outcome, usually from an undesired prediction to a desired one [54, 85, 113]. It has been heavily studied [17, 54, 88, 109] as it offers actionable and plausible feedback to the customers.

Example 1: Banks often collaborate with third-party companies *e.g.,* Upstart [9] and ZestAI [10] that offer ML-based credit assessment and decision-making of loan applications from their customers. The ML model M is owned by the third party and considers features of customers of the bank, *e.g.,* income, credit score (cscore), request loan amount (lamount). As an ML service user, the bank submits its customer Alice's information - income ($5K), cscore (fair), lamount ($100K) - to the third party. However, her loan is unfortunately denied by M. To assist the bank in helping Alice make the necessary adjustments needed for approval, the third party may provide a counterfactual

Authors' addresses: Shuai An, shuai.an@ed.ac.uk, University of Edinburgh, UK; Yang Cao, yang.cao@ed.ac.uk, University of Edinburgh, UK.

ACM 2836-6573/2024/6-ART130
https://doi.org/10.1145/3654933



(a) **Prior approaches**: Counterfactuals are computed by model owner. Adversaries can exploit counterfactuals to extract a surrogate M' with a high fidelity to M.

(b) **CPC** computes counterfactuals at client without model access. It also prevents adversaries from exploiting counterfactuals.

Fig. 1. State-of-the-art counterfactual explanation Vs. CPC

explanation with x'_a and x'_b: x'_a suggests an increase in income by \$1K and an improvement in cscore to 'good', while alternatively x'_b proposes to reduce lamount from \$100K to \$20K. □

Open challenges. While there has been a host of work on computing high-quality counterfactual [31, 37, 38, 64, 70, 71], there are however open challenges that impede their applications.

Challenge I: Right-to-explain. Not all ML systems, in particular those hosted in the cloud as a remote ML service [6, 110, 117], support counterfactual explanation. Instead, it demands both dedicated cooperation from model owners and engineering effort from service providers.

More specifically, to compute counterfactual explanations for a given instance x, existing approaches [52, 89, 114] work by generating multiple candidate instances close to x and invoke the model to predict on them, to find out those that are valid counterfactual explanations for x. This relies on multiple additional accesses to the model other than the inference on x, which is completely at the discretion of the model owners and service providers. As a result, not all ML systems and services support counterfactual explanations, failing to uphold "the right to explanation" requested by the European Union General Data Protection Regulation (GDPR) [80, 104].

[Open question**]**. A pressing ethical open question is *whether we can allow all the* users *of ML services to exercise their right to access counterfactual explanations for arbitrary ML models, at all times, even when the model owners or service providers opt not to?*

Challenge II: Privacy leakage. Another factor that hinders the adoption of counterfactual explanation in especially those remote ML services is the associated privacy leakage risks, as already observed in prior studies [12, 90, 105, 117]. Counterfactual explanations are essentially purposefully generated instances and their predictions. Hence, they carry information that is not part of the model inference. As a result, providing them to the users of the ML service can unintentionally reveal sensitive information about the ML model and the training set that the model owners may want to protect.

Example 2: Continue with Example 1, the third-party companies typically aggregate customer information from various banks to form a more extensive training set for developing a high-performing loan assessment model M. In such scenario, it is crucial for the third-party to ensure M's privacy [26, 57] against potential misuse by the other malicious ML service users, *e.g.*, rival banks. Counterfactuals x'_a and x'_b generated in Example 1 may come from the entire feature space or even directly from the training set of M, unintentionally revealing sensitive information about M to the bank. □

This vulnerability of counterfactuals has been exploited to launch model extraction attacks [12, 83, 117] against ML models. As shown in Fig. 1a, by training over counterfactuals, an adversarial

user can reconstruct a surrogate model M' with a high fidelity to the original model M, *e.g.,* up to 98.6% for extracting a typical DNN model [12, 117].

[Open question**]**. Hence, a critical open question is *whether there exists a counterfactual explanation that leaks* no *privacy information, thereby preventing model extraction attacks over counterfactuals.*

Client-centric privacy-preserving counterfactuals. To address the challenges, we present CPC (Client-centric Privacy-preserving Counterfactual explanation), a framework for computing counterfactual explanations with the following guarantees: (a) For model users, it permits access to counterfactual of their predictions at will, even when the model owners or ML service providers opt not to. (b) For model owners, it assures that the counterfactuals do not leak any information about the model and training data, precluding model extraction attacks over counterfactuals.

Guarded counterfactuals. The enabling idea of CPC is *guarded counterfactuals*. Unlike traditional counterfactuals that are artificially generated from the entire feature space and are "open-ended", guarded counterfactuals are from a designated universe $\mathbf{U}$ of instances and are "closed-ended".

Counterfactuals guarded by a privacy-preserving universe $\mathbf{U}$ would prevent adversaries from exploiting counterfactuals to extract ML models. A natural choice for such universe, denoted by $\mathbf{U}_I$, is inference instances that are predicted during model serving and are also those to be explained via counterfactuals. With $\mathbf{U}_I$, guarded counterfactuals strictly reveal nothing during explanation.

To make it work in practice, a fundamental problem is to compute guarded counterfactuals while optimizing their quality. We consider two popular quality measures of counterfactuals, namely diversity [85] and succinctness [54]. We show that the problem is intractable. Indeed, optimizing either diversity or succinctness while bounding the other is already NP-Complete. Despite the intractability, we present efficient PTIME approximation algorithms with provable bounds, for both measures.

Client-centric explanation. In contrast to existing approaches that rely on the model owner to generate counterfactual explanations, CPC operates completely at the client side, controlled by the user that receives predictions from, *e.g.,* a remotely hosted black-box ML model with restricted access.

To do this, CPC uses a client-side cache to collect inference instances as $\mathbf{U_I}$ during model serving. It then computes counterfactuals guarded by $\mathbf{U_I}$. For those clients that do not have local memory to hold all the relevant inference instances or $\mathbf{U_I}$ is an online stream, CPC can operate in a streaming explanation mode, under which it treats $\mathbf{U_I}$ as a stream of instances coming online and it incrementally refines explanation as each new inference instance passes, without the need to store $\mathbf{U_I}$.

We develop two streaming algorithms for computing guarded counterfactuals in such a mode, while optimizing their diversity and succinctness, respectively; when optimizing one quality measure, we bound the other. For both cases, our algorithms are provably competitive when compared to the optimal batch explanation algorithm that knows the entire $\mathbf{U_I}$ in advance (Theorem 4).

Effectiveness. We implement CPC and evaluate its effectiveness. We find the following.

(1) We first show that, with CPC, users can still have access to counterfactual explanation when using *e.g.,* Amazon SageMaker [6] service that comes with no support for counterfactual explanation. This also applies to any other ML system, be it remote or local.

(2) We validate the benefit of guarded counterfactuals in defending against model extract attacks. By comparing with 4 state-of-the-art counterfactual methods over real-life datasets, explanations generated by CPC completely prevent model extraction attacks from exploiting counterfactuals, while existing methods contribute to 14.7% to 30.6% of privacy leakage exploited by the attacks.

(3) Despite unique benefits of (1) and (2), guarded counterfactuals produced by CPC are consistently comparable to and often better in quality than those of state-of-the-art methods, *e.g.,* on average

35.5% higher in diversity and 39.5% more succinct over the 4 datasets.

(4) Furthermore, on average CPC is 205.7 times faster than existing counterfactual explanation methods, up to 3 orders of magnitude.

(5) In the streaming explanation mode, CPC takes only 1.7% of the memory cost of that in the batch explanation mode with just 17.8% and 14.5% of dip in diversity and succinctness, respectively.

Contributions. In summary, we make the following contributions:

- We identify open challenges in counterfactual explanation and propose guarded counterfactuals as a response (Sections 2-3).
- Based on it, we propose CPC, a framework for client-centric privacy-preserving counterfactual explanation. We study key problems underlying CPC and settle the complexity (Section 4).
- We present approximation algorithms for computing guarded counterfactuals (Section 5).
- We also develop competitive online algorithms for CPC to operate in streaming mode (Section 6).
- We implement CPC and empirically verify its effectiveness for enabling user-controlled counterfactual explanation that can ward off model extraction attacks over explanation (Section 7).

Remark. CPC offers contextual explanability, by operating on a relation of model input and output during inference as a context. This continues with previous efforts towards the vision of in-database model explanation [15], as CPC can in principle be deployed on or incorporated into a typical DBMS (see more in Section 4.1). It also serves as a response from our data management community to address the recent call for actions regarding "explainability for everyone" instead of model owners [23].

Related Work. We categorize the related work as follows.

Explainable machine learning. There is a vast literature on explanation techniques to interpret black-box ML models [17, 52, 74, 96, 97, 119]. They can be classified into two classes: local explanations and global explanations. Local explanations interpret the prediction of an individual instance, and are in contrast to global explanations, which aim to address how the model's overall behavior affects decisions. Among these, local explanations have gained more traction as they are more tailored and comprehensible to the specific contexts and cases at hand [84, 96, 97]. Current families of mainstream local explanations include *feature importance*, *rule-based* and *instance-based* methods. Feature importance methods, such as LIME [96] and SHAP [74], quantify the contribution of each feature to a single prediction. This can be accomplished using a proxy or by computing Shapley values. Moving on to rule-based methods, *e.g.,* Anchor [97], they aim to generate human-friendly explanations in the form of dependencies between feature values and the prediction of a targeted instance [59]. Instance-based explanations, also known as counterfactuals [31, 93, 114], offer a more tangible understanding of model decisions by illustrating what changes in the features could alter the prediction of an instance and lead to a desirable outcome. In this work, we focus on counterfactual explanation.

Counterfactual explanation. There has been a host of work on counterfactual explanation of ML predictions [16, 31, 38, 52, 70, 71, 85, 93, 109, 114, 119]. Conceptually, they work by formulating the task as an optimization problem where the objective function captures the quality of candidate instances as counterfactuals for the targeted instance. Such function is often a weighted combination of multiple measures of counterfactuals, *e.g.,* a linear combination [37] of diversity [85], closeness [54], sparsity [31]. While they share the overall structure of their objective functions, the key differences lie in their strategies in searching the optimal solutions in the entire feature space [16, 38, 52, 71, 85, 109] or the training set [31, 93, 119], by adopting gradient descent [85, 109, 114] or heuristics [52, 71]. There has also been a vast and rich literature on the causality [14, 48, 49, 58, 62, 65, 87, 91, 92, 99, 113, 122]

and fairness [66, 68, 95, 99, 103] of counterfactuals. By employing given or learned probabilistic causal models, these approaches aim to ensure that counterfactuals follow natural cause-effect laws and address potential unfairness or bias in certain subpopulations *e.g.,* a particular race or gender.

Our work differs from existing methods in the following. (1) Existing methods assume unlimited access to the model or the training set, which CPC does not. A key implication of this difference is that existing approaches have to be deployed and supported by the model owners in order to provide counterfactual explanation, which raises the open Challenge I. In contrast, CPC works for and is operated by the model users, irrelevant to the model owners. (2) This further implies that existing counterfactuals carry information about the model or training set, making them vulnerable to model extraction attacks as described in open Challenge II. Instead, CPC computes counterfactuals guarded by inference instances, averting model extraction attacks from exploiting counterfactuals. (3) CPC supports inference time counterfactual explanation and monitoring, which are not supported by existing systems. (4) Built upon simple and intuitive algorithms, CPC is up to 3 orders of magnitude faster than existing approaches with comparable or even higher quality on public datasets.

Privacy attack via counterfactuals. In the realm of ML security, adversaries commonly launch model extraction attacks [12, 36, 60, 117] against ML models, in particular those deployed in remote cloud servers. The attack aims to train a substitute model that approximates the original model well in terms of accuracy and fidelity. While there has been a host of work on both counterfactual computation and privacy attacks against ML models separately, only recently attention has started been paid to model attacks via counterfactuals. A recent study [12] firstly showed how models can be easily stolen by treating counterfactuals as training instances. Based on the intuition that counterfactuals are close to the decision boundary of an ML model, [117] subsequently proposed a more effective model extraction attack that additionally uses the counterfactuals of counterfactuals, which they proved is able to extract a linear model with perfect fidelity.

In contrast to existing work that focuses on attacks that exploit counterfactuals, we aim to defend such attacks for explanations. CPC computes counterfactuals guarded by inference instances that naturally ward off such attacks, without even affecting the performance of explanation and the quality of counterfactuals. While CPC handles both model extraction attack and membership attack, in this work we focus on the former as it is more complex and robust.

Database interpretability. Related is also the work on interpretability from the database community [14, 51, 94, 111, 115, 116], largely motivated by the need for *e.g.,* understanding query results [77, 98, 100], debugging systems [45, 73], and responsible data analysis [95, 102, 103]. With these diverse purposes, a range of interpretability approaches and frameworks have emerged [77, 81, 101]. This includes (a) data provenance [21, 32, 33, 53, 81], which identifies and expresses how input tuples contribute to the output of a database query, (b) causality [47, 49, 77–79] that explains queries in the form of interventions (*i.e.,* the addition, deletion or modification of tuples), by calculating the responsibility of each tuple in producing a specific query result [100] or providing high-level abstract explanations (*i.e.,* predicates) [43, 120], and (c) cluster-based summarization [72, 101, 108] and observation-based statistical methods [118]. We refer to recent surveys [51, 94] for more technical details. Orthogonal to them, CPC targets new issues around counterfactuals.

Data selection. In essence, CPC reduces the problem of explaining model predictions to a data selection task within the explanation (inference) universe. While CPC adopts variants of minimum cover and maximum coverage algorithms [18, 19, 76, 112] as proof-of-concept, in principle it can work with a more diverse range of data selection techniques [24, 25, 29, 61, 86]. Intuitively, data selection refers to selecting a subset from a large set of candidate data that meets specific conditions, while optimizing *e.g.,* fairness [11, 86], diversity [24, 86], among others [25]. In particular, package

queries [28, 29, 29, 30, 75] have been proposed as a generic approach to supporting data selection atop a typical RDBMS. We remark that they can be well incorporated into CPC and extend the latter with additional objectives and constraints that apply to counterfactuals, *e.g.*, fairness [68], potentially opening an avenue towards a vision of in-database support of model explanation.

2 PRELIMINARIES

We review the basics of counterfactual explanation and justify the associated open challenges.

ML basics. Let $A_1, \ldots, A_d$ be d features, where each $A_i (i \in [1, d])$ draws values from a domain $\mathrm{dom}(A_i)$. The *feature space* is denoted by $\mathcal{X} = \mathrm{dom}(A_1) \times \cdots \times \mathrm{dom}(A_d)$, and a tuple x in $\mathcal{X}$ is called an *instance*. $\mathsf{x}[A_i]$ is the projection of x on feature A_i. Consider a blackbox ML model M in the context of supervised learning. For any instance $\mathsf{x} \in \mathcal{X}$, M returns a prediction (*a.k.a.* label) $M(\mathsf{x}) \in \mathcal{Y}$, where $\mathcal{Y}$ is the *label space*. M is typically trained by a supervised ML algorithm using a set of pre-labeled instances from $\mathcal{X}$; these instances are called *training instances* and the process is referred to as *model training*. Once M is trained, it can be used to generate predictions for new unknown instances. This is known as *model inference*, and such new instances are known as *inference instances*.

Counterfactuals. Given a model M and an instance $\mathsf{x} \in \mathcal{X}$ to be explained, a *counterfactual* x' refers to an instance from $\mathcal{X}$ such that $M(\mathsf{x}') \neq M(\mathsf{x})$. In practice, counterfactuals may also have to meet application-specific conditions, *e.g.,* validity [54], feasibility [113], sparsity [31] and similarity [37]. For example, validity assures that $M(\mathsf{x}')$ of counterfactual x' should be a desired or targeted outcome.

For simplicity of exposition, we assume that the counterfactuals have passed the checks of those application-specific conditions; however, as will be shown in the experimental study (Section 7), our approach is generic and does support all these additional properties.

Counterfactual explanation. Given model M and instance x, a *counterfactual explanation* for x is a set $\mathbf{C}$ of counterfactuals for x [85]. There are up to exponentially many counterfactual explanations for a given instance and we often want to compute those of high-quality. Two commonly used quality measures for counterfactual explanations are *diversity* [37, 85] and *succinctness* [54, 113].

Diversity. The diversity of a counterfactual explanation $\mathbf{C}$ for x, denoted by $\mathrm{diverse}(\mathbf{C})$, is defined as $|\bigcup_{\mathsf{x}' \in \mathbf{C}} S(\mathsf{x}', \mathsf{x})|$, where $S(\mathsf{x}', \mathsf{x})$ is the set of features that differ between x' and x. Intuitively, a counterfactual explanation with larger diversity carries more possible actions that can be taken to change the prediction of x [54].

Succinctness. The succinctness of counterfactual explanation $\mathbf{C}$, denoted by $\mathrm{succinct}(\mathbf{C})$, is the number of counterfactuals in $\mathbf{C}$. Intuitively, succinct explanation $\mathbf{C}$ is more digestible as it contains less instances and requires less effort to interpret [82].

An ideal counterfactual explanation should be both diverse and succinct [54, 64]. However, there is an inherent trade-off between the two measures [31, 113]. Intuitively, an explanation $\mathbf{C}$ with more counterfactuals might naturally boost diversity, but this can compromise succinctness, and vice versa. Moreover, it is not clear whether having multiple counterfactuals, each focusing on a single feature modification, is superior to fewer counterfactuals with changes over more diverse features. The interplay of these two properties suggests that neither dominates the other. The choice to prioritize one property over the other depends on the specific task at hand [31].

Model serving architecture. We consider the cloud-based model serving architecture, which is ubiquitous in real-life applications. As shown in Fig. 1, there are three different roles in such a workflow: model owner, model users and end customers. The model owner owns the ML model, which is deployed and hosted by some model serving service provider in the cloud. The service provider exposes a controlled interface to the ML model so that model users can access at the

Id	age	income	cscore	lamount	education	prediction
1	30	\$5K	fair	\$100K	Bachelor	No
2	30	\$7K	fair	\$50K	Bachelor	Yes
3	30	\$5K	good	\$100K	Bachelor	Yes
4	30	\$10K	fair	\$100K	Bachelor	Yes
5	30	\$6K	poor	\$100K	Bachelor	No
6	28	\$5K	fair	\$100K	Bachelor	Yes
7	30	\$6K	fair	\$50K	Master	Yes
8	30	\$20K	fair	\$200K	Bachelor	Yes

Fig. 2. An example inference universe

client-side, *e.g.*, providing an instance x for inference and obtaining a prediction $M(\mathsf{x})$ as a response.

Model users may use the predictions with some decorations to serve end customers, so that customers do not need to access model directly. In Example 1, the model owner is Upstart [9] and ZestAI [10] that assist banks with loan-decisions. The bank is a model user and Alice is a customer.

Challenge: right-to-explain. While almost all major cloud providers offer ML model hosting service [6–8], only few of them provide counterfactual explanations [119]. Despite the active research on counterfactual explanation [31, 37, 38, 64, 85, 88, 109], we are still in great need of an approach applicable to these remotely hosted ML models even when service providers have decided not to.

Model extraction attacks. Driven by the desire to circumvent training costs, adversarial users often launch model extraction attacks [36, 60, 110] to steal ML models that are typically proprietary.

Model extraction attacks aim to extract a functionally equivalent surrogate model M' that behaves similarly to the targeted model M. Specifically, the adversary queries M with q instances $\mathsf{x}_1, \ldots, \mathsf{x}_q$, obtaining q predictions $M(\mathsf{x}_1), \ldots, M(\mathsf{x}_q)$ from M, just as normal users. It then trains a replica model M' on the dataset $\{(\mathsf{x}_1, M(\mathsf{x}_1)), \ldots, (\mathsf{x}_q, M(\mathsf{x}_q))\}$ to minimize the loss over an evaluation set $\mathcal{X}_e \subseteq \mathcal{X}$: $\frac{1}{|\mathcal{X}_e|} \sum_{\mathsf{x} \in \mathcal{X}_e} I(M(\mathsf{x}) = M'(\mathsf{x}))$, where I is the indicator function.

Challenge: attacks over counterfactuals. Since counterfactuals tend to align closely with the manifold of training instances [37, 54, 113], ML systems supporting counterfactual explanation may be more susceptible to exposure. In particular, counterfactuals can be exploited by the adversaries as additional training data of high quality to learn M'. By asking the service provider counterfactual explanations $\mathbf{C}_1, \ldots, \mathbf{C}_q$ for inference instances $\mathsf{x}_1, \ldots, \mathsf{x}_q$, an adversary now have a set $\bigcup_{i \in [1,q]} \mathbf{C}_i$ of high quality "training instances" that assists its model extraction. Using counterfactual explanation, it has been shown that the attack can "recover" a typical DNN model with fidelity up to 98.6% [12, 117].

3 GUARDED COUNTERFACTUAL

To address the challenges, we propose *guarded counterfactual explanation* with the below guarantees:

(G1) it enables counterfactual explanation for model users even if model owners opt not to; and

(G2) it is privacy-preserving such that it cannot be exploited for model extraction attacks.

In contrast to prior methods that draw instances from the entire feature space or training set as counterfactuals, we consider counterfactuals guarded by a privacy-preserving explanation universe.

Explanation universe. Consider an ML model M with feature space $\mathcal{X}$. An *explanation universe* $\mathbf{U}$ for M is a subset of $\mathcal{X}$. Consider a set $\mathbf{I} \subseteq \mathcal{X}$ of inference instances processed during model serving and a set $\mathbf{P} \subseteq \mathcal{X}$ of instances that the owner of M wants to protect.

Universe $\mathbf{U}$ is *privacy-preserving* for M *w.r.t.* $\mathbf{I}$ and $\mathbf{P}$ if for any ML architecture $\mathcal{M}$ defined over $\mathcal{X}$, the trained model of $\mathcal{M}$ over $\mathbf{I}$, denoted by $M_{\mathbf{I}}$, and that trained over $\mathbf{I} \cup \mathbf{U}$, denoted by $M_{\mathbf{U}}$, satisfy the following (assuming the same training algorithm):

$$\Pr(M_{\mathbf{I}}(\mathsf{x}) = M(\mathsf{x})) = \Pr(M_{\mathbf{U}}(\mathsf{x}) = M(\mathsf{x})) \quad (\forall \mathsf{x} \in \mathbf{P})$$

where $\Pr(M_{\mathbf{I}}(\mathsf{x}) = M(\mathsf{x}))$ is the probability that $M_{\mathbf{I}}$ and M make the same prediction for x; similarly for $\Pr(M_{\mathbf{U}}(x) = M(\mathsf{x}))$. Intuitively, when $\mathbf{U}$ is privacy-preserving, exposing instances from $\mathbf{U}$ and their predictions by M to the model users does not give any advantage to them to extract sensitive information about M specified by $\mathbf{P}$, *i.e.,* the predictions of instances from $\mathbf{P}$ made by M.

Guarded counterfactuals. Let x be an inference instance in $\mathbf{I}$. A counterfactual x' of x is *guarded* by universe $\mathbf{U}$ if x' is in $\mathbf{U}$. Since, nevertheless, inference instances in $\mathbf{I}$ and their predictions will need to be provided to model users, by definition counterfactuals guarded by a privacy-preserving universe $\mathbf{U}$ will naturally give zero advantage to adversarial users to extract predictions of M over instances protected by $\mathbf{P}$. Hence, they completely ward off model extraction attacks over counterfactuals.

Inference universe and counterfactuals. A natural choice of privacy-preserving universes is the inference set $\mathbf{I}$ itself or a subset thereof, which we refer to as an *inference universe* and denote by $\mathbf{U_I}$. Note that for any set $\mathbf{U_I} \subseteq \mathbf{I}$, any $\mathbf{P}$ of instances to be protected and any $\mathsf{x} \in \mathbf{P}$, we have $\Pr(M_{\mathbf{I}}(\mathsf{x}) = M(\mathsf{x})) = \Pr(M_{\mathbf{U_I}}(\mathsf{x}) = M(\mathsf{x}))$ since $\mathbf{I} \cup \mathbf{U_I} = \mathbf{I}$. That is, $\mathbf{U_I}$ is privacy-preserving for M *w.r.t.* $\mathbf{I}$ and $\mathbf{P}$.

Counterfactuals of instance x for M guarded by an inference universe $\mathbf{U_I}$ for M are referred to as *inference counterfactuals* of x.

Example 3: Continue with Example 1. Consider the inference set $\mathbf{I}$ in Fig 2 with 8 loan applications and predictions by M. Applications 2-4 and 6-8 could serve as inference counterfactuals for application 1. Application 5 cannot as it was also rejected, *i.e.,* having the same prediction as application 1. □

In addition to the capability of preserving privacy, computing inference counterfactuals, *i.e.,* the instances and associated predictions, requires only knowledge of $\mathbf{I}$ and predictions of M over $\mathbf{I}$, which is already available for model users during model serving. As a result, this naturally confirms:

Proposition 1: *Explanation composed of inference counterfactuals upholds guarantees* G1 *and* G2. □

Remark. In practice, an adversarial model user can perform model extraction attacks using only the inference set [60, 83]; however, such attacks can be made much more effective when the adversary has access to counterfactuals [12, 117], as they carry abundant additional information about the model. Guarded counterfactuals aim to prevent adversaries from exploiting counterfactuals to assist in model extraction. Orthogonal to this, privacy-preserving ML [34, 117, 121] can help defend attacks against the inference set.

4 CLIENT-CENTRIC EXPLANATION

While Proposition 1 tells us that we can address challenges raised earlier with inference counterfactuals for explanation, we have however not discussed how to compute such explanations. To this end, we present CPC, a framework for computing high-quality explanations via inference counterfactuals from model user's perspective.

4.1 Framework

As shown in Fig. 1b, CPC co-locates with the client-side model interface that sends inference instances to the remote model and receives corresponding predictions. It collects a relation of model input and output that the user sends and receives during model serving. CPC uses such relation as the explanation universe $\mathbf{U_I}$ and computes counterfactuals guarded by $\mathbf{U_I}$, without any access to the remote model. Depending on how CPC reads $\mathbf{U_I}$, it can operate under two modes: *batch explanation* mode and *streaming explanation* mode.

Batch explanation. In this mode, CPC analyzes inference instances and their associated predictions in batches, decided and collected via a local cache, *e.g.,* a buffer pool or a Redis cache. For a given batch as the universe $\mathbf{U_I}$ and a targeted instance x to be explained, it computes an explanation $\mathbf{C}$ for x with counterfactuals guarded by $\mathbf{U_I}$, while optimizing its quality, *i.e.,* diversity and succinctness.

CPC offers a fine-grained and firm control over the quality of explanations, by allowing users to purposefully optimize one quality measure while strictly bounding the other. This is based on the study of the following two problems underlying CPC.

Succinctness-bounded explanation. We state the succinctness-bounded explanation problem (SBEP).

Input	an instance $\mathsf{x} \in \mathcal{X}(A_1, , \ldots, A_d)$, a batch $\mathbf{U_I}$ of inference instances from $\mathcal{X}$ and their associated predictions by a (blackbox) ML model M, an integer k.
Output	explanation $\mathbf{C}$ of counterfactuals guarded by $\mathbf{U_I}$ for x.
Objective	maximize diverse($\mathbf{C}$).
Constraint	succinct($\mathbf{C}$) $\leq k$.

Intuitively, SBEP is to identify the most diverse explanation $\mathbf{C}$ with up to k counterfactuals for x.

Diversity-bounded explanation. Similarly, the diversity-bounded explanation problem, denoted by DBEP, is to compute the most succinct explanation $\mathbf{C}$ (*i.e.,* maximize succinct($\mathbf{C}$)) with inference counterfactuals that has diversity above a given lower bound p.

Streaming explanation. In practice, the data input to an ML system may come in a streaming manner, *e.g.,* a stream of transactions from an online bank [56] or measurements from a sensor network [107]. These datasets are often too large to fit into memory, particularly for users with limited resources, *e.g.,* no memory for caching the entire $\mathbf{U_I}$. However, it is often necessary to take into account all relevant instances when explaining, to provide explanation of high quality globally.

To this end, CPC offers to operate in the streaming explanation mode. Under this mode, instead of collecting and batching the inference universe $\mathbf{U_I}$, CPC treats $\mathbf{U_I}$ as a stream with inference instances coming online one by one. Upon the arrival of each new instance x in $\mathbf{U_I}$, it incrementally refines the explanation it computes before the arrival of x as $\mathbf{U_I}$ grows to $\mathbf{U_I} \cup \{\mathsf{x}\}$, while maximizing diversity (for SBEP) or minimizing succinctness (for DBEP).

Note that in this mode CPC does not need to "remember" the inference universe as it does when operating in the batch mode. It requires less memory overhead and maintains explanations in real-time. However, due to the memoryless streaming access to the inference universe $\mathbf{U_I}$, the explanation quality can be lower than those found by CPC in the batch mode. This said, as will be shown in Sections 4.2 and 6, we give streaming algorithms for both sSBEP and sDBEP that are competitive to their counterparts in the batch mode.

Benefits. CPC brings several benefits. (1) As discussed in Section 3, with inference counterfactuals CPC provides privacy-preserving explanations at the disposal of model users. (2) CPC avoids accessing ML models, significantly improving the efficiency of explaining (see Section 7.4). (3) By varying $\mathbf{U_I}$, CPC can provide users with timely, dynamic and contextual explanations, offering more insight (see Section 7.2). (4) CPC reduces the problem of explaining into a data-driven task over $\mathbf{U_I}$, opening new possibilities of further extensions, *e.g.,* fair [24, 86] and causality-aware [48, 49, 99] explanation.

Assumptions and limitations. We assume the following.

(a) On model users. CPC targets model users with long-term high demand for 3rd party ML inference services, *e.g.,* banks that use credit assessment models [9, 10]. Such users often rely on predictions from the 3rd party models to serve large groups of customers (source of $\mathbf{U_I}$) who may require

reliable and real-time explanations that the model owners do not provide due to cost or privacy concerns. CPC cannot serve ad-hoc model users with limited inference instances.

(b) On explanation universe $\mathbf{U_I}$. To make practical use of CPC, we assume that instances in $\mathbf{U_I}$ and their predictions by M are causally consistent [58, 99], such that CPC could align well with real-world cause and effect specified by, *e.g.,* causal models [49, 91]. This may be done by incorporating causality-aware explanations [48, 49] and exploiting domain knowledge [65, 87, 92], to capture and enforce causal dependencies between features of instances in $\mathbf{U_I}$.

In addition, instances in $\mathbf{U_I}$ are assumed to be relevant and close to those to be explained. This is to assure that counterfactuals guarded by $\mathbf{U_I}$ can meet application-specific criteria, *e.g.,* sparsity, similarity (recall Section 2). Moreover, for counterfactuals to be actionable and realistic [64], $\mathbf{U_I}$ shall follow the same or a similar distribution of the training data [113], or come from the data manifold [62]. We remark that this is often the case as inference set and training set are supposedly close in typical learning setup [22, 110].

Although CPC by default uses inference universe $\mathbf{U_I}$, it is not the only option for CPC to function; application-specific, privacy-preserving universes picked by the users are also a viable option.

(c) On deployment. Due to the client-centric design, CPC has limitations on use. (1) It assumes that model predictions are deterministically determined by features provided by the model user. If the model also uses internal variables, their values must be passed to the client user along with the predictions of inference instances during model serving, so that CPC can reason properly. (2) CPC may suffer from a cold-start when there are no sufficient instances in $\mathbf{U_I}$. To remedy this, one may fall back on traditional methods for start. (3) Instances may get explanations of varying quality depending on their "representatives" in $\mathbf{U_I}$, which can lead to *e.g.,* unfairness [40]. One way to improve upon this is to incorporate fairness data selection [11, 86] into CPC, or augment $\mathbf{U_I}$ with selected instances sampled from feature space to improve quality and fairness. (4) While CPC prevents model user (*e.g.,* a bank) from extracting 3rd party models via counterfactuals, the explanation that an end customer of the user receives may still reveal information about other peers in the same $\mathbf{U_I}$. One can mitigate this by anonymizing [20, 46] $\mathbf{U_I}$ or using differential privacy [41, 42] as a post-hoc step.

4.2 Complexity of Client-Centric Explanation

We next settle the complexity of key problems underlying CPC.

Complexity. Both SBEP and DBEP are intractable when CPC operates in batch explanation mode.

Theorem 2: *Both* SBEP *and* DBEP *are* NP-Complete. □

Proof sketch: SBEP is in NP because one can verify in polynomial time whether a set of inference counterfactuals $\mathbf{C}$ satisfies $\text{diverse}(\mathbf{C}) \geq l$ and $\text{succinct}(\mathbf{C}) \leq k$. Clearly, the same also applies to DBEP. We next show that both SBEP and DBEP are NP-Hard.

(1) We show that SBEP is NP-Hard by a reduction from the maximum coverage (MCR) problem, which is NP-Complete [50]. Given an instance of MCR, *i.e.,* a set U of elements $e_1, \ldots, e_d$, a collection $\mathcal{F}$ of subsets $S_1, \ldots, S_m$ of U, and integers l and k, MCR is to decide whether there exists an l-coverage of U, *i.e.,* at least l elements of U can be covered by k sets in $\mathcal{F}$. Given U, $\mathcal{F}$, l and k, we construct an inference universe $\mathbf{U_I}$ over d features $A_1, \ldots, A_d$ that consists of $m+1$ instances $\mathsf{x}_1, \ldots, \mathsf{x}_m$ and x:

- $\mathsf{x} = (a_1, \ldots, a_d)$;
- for each $e_i \in U$ and $S_j \in \mathcal{F}$, set $\mathsf{x}_j[A_i] = a_i$ if $e_i \notin S_j$;
- for any $\mathsf{x}_j[A_i]$ not decided above, set it to a distinct constant;
- labels of $\mathsf{x}_1, \ldots, \mathsf{x}_m$ are all the same but different from that of x.

ALGORITHM 1: Algorithm dSBC

Input: inference universe $\mathbf{U_I} = \{\mathsf{x}_1, \ldots, \mathsf{x}_m\}$, an instance x_0, integer k.
Output: an counterfactual explanation $\mathbf{C}$ for x_0.

$\mathbf{C} \leftarrow \emptyset$; $\mathbf{A} \leftarrow \{1, \ldots, d\}$; // $\mathbf{A}$: *features of* $\mathcal{X} = \{A_1, \ldots, A_d\}$; *i refers to feature* A_i
$\mathsf{x}_i \leftarrow \arg\min_{\mathsf{x}_i \in \mathbf{U_I}} |\mathbf{A}[\mathsf{x}_i = \mathsf{x}_0]|$;
$\mathbf{C} \leftarrow \mathbf{C} \cup \{\mathsf{x}_i\}$; $\mathbf{U_I} \leftarrow \mathbf{U_I} \setminus \{\mathsf{x}_i\}$;
while $|\mathbf{C}| < k$ **do**
  $\mathsf{x}_i \leftarrow \arg\min_{\mathsf{x}_i \in \mathbf{U_I}} |\bigcap_{\mathsf{x}_j \in \mathbf{C}} \mathbf{A}[\mathsf{x}_j = \mathsf{x}_0] \cap \mathbf{A}[\mathsf{x}_i = \mathsf{x}_0]|$;
  $\mathbf{C} \leftarrow \mathbf{C} \cup \{\mathsf{x}_i\}$; $\mathbf{U_I} \leftarrow \mathbf{U_I} \setminus \{\mathsf{x}_i\}$;
return $\mathbf{C}$

We show that there exists $\mathbf{C} \subseteq \mathbf{U_I}$ with $\mathsf{diverse}(\mathbf{C}) \geq l$ and $\mathsf{succinct}(\mathbf{C}) = k$ for x *if and only if* the MCR instance has an l-coverage.

(2) We next prove that DBEP is Np-Hard by reduction from minimum set cover (MSC), a classic Np-Complete problem [50]. Given an MSC instance, namely, a set U of elements $e_1, \ldots, e_d$, a collection $\mathcal{F}$ of subsets $S_1, \ldots, S_m$ of U, and an integer k, MSC is to decide whether there exists a k-cover, *i.e.,* k sets in $\mathcal{F}$ that together cover all d elements of U. Given U, $\mathcal{F}$ and k, we build the same $\mathbf{U_I}$ as in (1). We then prove that there exists $\mathbf{C} \subseteq \mathbf{U_I}$ with $\mathsf{succinct}(\mathbf{C}) = k$ and $\mathsf{diverse}(\mathbf{C}) = d$ for x *if and only if* the MCR instance has a k-cover. □

Approximability. In light of Theorem 2, any practical algorithm for SBEP or DBEP has to be approximate or heuristic. Nonetheless, we show that, in the batch mode, there exist approximation algorithms for both SBEP and DBEP with performance guarantees.

Following standard notion of approximation ratios [112], we have:

Theorem 3: *(1) There exists a polynomial-time* $\frac{e}{e-1}$*-approximation algorithm for problem* SBEP.

(2) There is a polynomial-time $\ln p$*-approximation for* DBEP. □

We will prove Theorem 3 by giving such algorithms in Section 5.

Competitiveness. Using competitiveness analysis [13], we show that there exist algorithms for streaming SBEP and DBEP with bounded competitiveness against the *optimal batch* explanations.

More specifically, we say that an algorithm $\mathcal{A}_s$ for streaming SBEP is c-competitive if for any batch algorithm $\mathcal{A}_b$, any input to SBEP, $c \cdot \mathsf{diverse}(\mathbf{C}_s) \geq \mathsf{diverse}(\mathbf{C}_b)$, where $\mathbf{C}_s$ is the explanation computed by $\mathcal{A}_s$ that incrementally maintains $\mathbf{C}_s$ when the universe $\mathbf{U_I}$ is revealed one by one as a stream; $\mathbf{C}_b$ is computed by $\mathcal{A}_b$ that reads the entire $\mathbf{U_I}$ as a batch; similarly for streaming DBEP.

Theorem 4: *(1) There exists a 4-competitive algorithm for the streaming version of* SBEP.

(2) There exists a $4\sqrt{2d}$*-competitive algorithm for streaming* DBEP *assuming* $p \geq d$ *where* d *is the number of features.* □

This assures that even if the client has no memory to batch $\mathbf{U_I}$, users can still employ CPC to obtain guarded counterfactual explanations; moreover, the explanation quality is guaranteed close to the optimal that they would have hoped for in the batch mode with infinitely large memory to collect $\mathbf{U_I}$.

5 BATCH CLIENT-CENTRIC EXPLANATION

As a constructive proof of Theorem 3, we present algorithms for both SBEP and DBEP in this section.

Algorithm dSBC. We start with an algorithm for problem SBEP, denoted by dSBC and shown in Algorithm 1. Given an instance x_0 to be explained, an inference universe $\mathbf{U_I} = \{\mathsf{x}_1, \ldots, \mathsf{x}_m\}$ and

integer k, dSBC returns a diverse set $\mathbf{C}$ of at most k counterfactuals guarded by $\mathbf{U_I}$ as the explanation for x_0. For the ease of presentation, we assume that each instance in $\mathbf{U_I}$ is a qualified counterfactual for x_0; in practice this can be done by a linear scan of $\mathbf{U_I}$ which filters out those instances that do not comply with application-specific constraints on counterfactuals, *e.g.,* validity, sparsity or similarity.

Similar to the classic MCR algorithm [112], algorithm dSBC works by iteratively picking instances in $\mathbf{U_I}$ as counterfactuals in $\mathbf{C}$ until $\mathbf{C}$ grows to the limit specified by k. In each iteration, it selects one that has fewest identical features with x_0. More specifically, it first identifies an initial counterfactual (lines 1-3) and then iteratively picks more counterfactuals from $\mathbf{U_I}$ one by one (lines 4-6). Denote by $\mathbf{A}[\mathsf{x}_i = \mathsf{x}_0]$ the set of features where x_i and x_0 share the same value. Each time dSBC picks x_i among all the instances from $\mathbf{U_I}$ that are not yet in $\mathbf{C}$, such that $|\bigcap_{\mathsf{x}_j \in \mathbf{C}} \mathbf{A}[\mathsf{x}_j = \mathsf{x}_0] \cap \mathbf{A}[\mathsf{x}_i = \mathsf{x}_0]|$ is the minimum among all (line 5). The set $\mathbf{C}$ is returned once it contains k counterfactuals (line 7).

Algorithm sDBC. We next present an algorithm for DBEP, denoted by sDBC, to generate a counterfactual explanation $\mathbf{C}$ with diversity bounded by p while minimizing its succinctness. It works the same as dSBC except the termination condition of the iterations (line 4 of dSBC in Algorithm 1), similar to the classic MSC algorithm [112]. Specifically, the iterations now terminate when $|\bigcup_{\mathsf{x}_j \in C} \mathbf{A}[\mathsf{x}_j \neq \mathsf{x}_0]| \geq p$, where $\mathbf{A}[\mathsf{x}_j \neq \mathsf{x}_0]$ represents the set of features on which x_j and x_0 disagree. Intuitively, once counterfactuals in $\mathbf{C}$ differ from x_0 on p distinct features, sDBC terminates.

Example 4: Continue with Example 3. Assume that applications 2-4 are valid counterfactual candidates for application 1. (1) Given $k = 2$, dSBC initially picks application 2, as it shares only three features with application 1. dSBC further selects application 3 via lines 5-6, since it shares the fewest features with applications 1-2. Hence, applications 2 and 3 are returned as counterfactual explanation $\mathbf{C}$ for application 1. (2) Along the same lines, sDBC returns applications 2 and 3 as the explanation for application 1 when $p = 3$. □

Analysis. Both dSBC and sDBC can be implemented in $O(m^2 d)$-time, where m is the number of instance in $\mathbf{U_I}$ and d is the number of features of an instance. The algorithms warrant the following.

Lemma 5: *(1) For any k, the diversity of the optimal explanation is at most $(1 + \frac{1}{e-1})$-times of that of $\mathbf{C}$ returned by* dSBC.

(2) For any p, sDBC *always returns an explanation $\mathbf{C}$ with a succinctness that is at most $\ln p$-times of that of the optimal explanation.* □

Remarks. (1) One might be tempted to "reverse" the reduction in the proof of Theorem 2 to reduce SBEP to MCR, and use classic MCR algorithm [112] for SBEP. We remark that this is indeed a valid approach to SBEP; However, it can incur higher cost than dSBC as the reduction, if "reversed", would involve computing the complement of $\mathbf{A}[\mathsf{x}_i = \mathsf{x}_0]$ (corresponds to line 2 and 5 of dSBC), which can be expensive if the number of instances relevant to x_0 is high; sDBC is also similar. This said, both algorithms and the proof of Lemma 5 (omitted due to space limit) do largely follow the idea of MCR and MSC [112], but without explicitly implementing the reductions.

(2) CPC is extensible. It can be extended with alternative solutions to SBEP and DBEP, *e.g.,* data selection [24, 86], to take into account additional factors like fairness [11, 86], or package queries [29, 75] which would add the support of CPC in a typical database system.

6 STREAMING EXPLANATION

As a proof of Theorem 4, we give algorithms for streaming SBEP and DBEP, for CPC to operate in the streaming explanation mode. Unlike batch mode in Section 5, the approach to the streaming mode follows the "reverse" of reductions in the proof of Theorem 2, to adopt classic streaming

ALGORITHM 2: Algorithm $\widetilde{\text{dSBC}}$

Input: $\mathbf{U_I}$ as a stream with $\mathtt{x}_1, \mathtt{x}_2, \ldots$ arriving online, an instance $\mathtt{x}_0$, integer k.
Output: a counterfactual explanation $\mathbf{C}$ when $\mathbf{U_I}$ ends.

1 $\mathbf{C} \leftarrow \emptyset$;
Upon the arrival of instance $\mathtt{x}_t (t \geq 1)$ of $\mathbf{U_I}$:
2 **if** $|\mathbf{C}| < k$ **then** $\mathbf{C} \leftarrow \mathbf{C} \cup \{\mathtt{x}_t\}$;
3 $\mathbf{C} \leftarrow \mathtt{uCdiv}(\mathbf{C}, \mathtt{x}_t)$;
When the stream $\mathbf{U_I}$ ends:
4 **return** $\mathbf{C}$

Procedure $\mathtt{uCdiv}(\mathbf{C}, \mathtt{x}_t)$ // *Update* $\mathbf{C}$ *for each arriving* $\mathtt{x}_t$
5 $\quad \mathbf{A} \leftarrow \{1, \ldots, d\}$ // $\mathbf{A}$*: features of* $\mathcal{X}$*,* i.e.*,* i *corresponds to feature* A_i
6 $\quad \mathtt{x}_i \leftarrow \arg\min_{\mathtt{x}_i \in \mathbf{C}} |\mathbf{A}[\mathtt{x}_i \neq \mathtt{x}_0] \setminus \bigcup_{\mathtt{x}_j \in (\mathbf{C} \setminus \{\mathtt{x}_i\})} \mathbf{A}[\mathtt{x}_j \neq \mathtt{x}_0]|$;
7 $\quad A_u \leftarrow \bigcup_{\mathtt{x}_j \in \mathbf{C}} \mathbf{A}[\mathtt{x}_j \neq \mathtt{x}_0]$;
8 $\quad$ **if** $|A_u \setminus \mathbf{A}[\mathtt{x}_i \neq \mathtt{x}_0] \cup \mathbf{A}[\mathtt{x}_t \neq \mathtt{x}_0]| \geq (1 + \frac{1}{k})|A_u|$ **then**
9 $\quad\quad \mathbf{C} \leftarrow \mathbf{C} \setminus \{\mathtt{x}_i\}; \mathbf{C} \leftarrow \mathbf{C} \cup \{\mathtt{x}_t\}$;
10 $\quad$ **return** $\mathbf{C}$

MCR [18, 76, 123] and MSC [19, 44] for CPC. For completeness, we first describe the algorithms and then show that they implement the reductions, thus verifying Theorem 4.

Algorithm $\widetilde{\text{dSBC}}$. We start with streaming SBEP. We give an algorithm, denoted by $\widetilde{\text{dSBC}}$ and shown in Algorithm 2, that computes a counterfactual explanation $\mathbf{C}$ guarded by a streaming universe $\mathbf{U_I}$ It works by incrementally maintaining $\mathbf{C}$ (initially empty) upon the arrival of each inference instance in $\mathbf{U_I}$ (lines 2-3). It does not memorize arrived instances due to lack of local memory in streaming mode, and returns $\mathbf{C}$ after the stream ends (line 4). The goal is to maximize the diversity of $\mathbf{C}$ while subject to a succinctness bound k.

Upon the arrival of $\mathtt{x}_t$, if $\mathbf{C}$ contains less than k instances, $\widetilde{\text{dSBC}}$ adds $\mathtt{x}_t$ to $\mathbf{C}$ (line 2). Otherwise, it decides whether $\mathtt{x}_t$ should be added to $\mathbf{C}$ and if so which existing instance needs to be evicted from $\mathbf{C}$ to maintain succinctness bound k (via procedure uCdiv; line 3).

More specifically, uCdiv first identifies the existing instance $\mathtt{x}_i$ in $\mathbf{C}$ that contributes least to diverse($\mathbf{C}$) (line 6), measured as $|\mathbf{A}[\mathtt{x}_i \neq \mathtt{x}_0] \setminus \bigcup_{\mathtt{x}_j \in (C \setminus \{\mathtt{x}_i\})} \mathbf{A}[\mathtt{x}_j \neq \mathtt{x}_0]|$. Recall that $\mathbf{A}[\mathtt{x}_i \neq \mathtt{x}_0]$ is the set of features where $\mathtt{x}_i$ and $\mathtt{x}_0$ differ. Intuitively, this counts the number of "private" features in $\mathbf{C}$ that are exclusively present in $\mathtt{x}_i$, *i.e.,* no other instances in $\mathbf{C}$ differ from $\mathtt{x}_0$ over such features. With this, $\widetilde{\text{dSBC}}$ then replaces $\mathtt{x}_i$ with $\mathtt{x}_t$ only if the contribution of $\mathtt{x}_t$ to $\mathbf{C}$ satisfies $|A_u \setminus \mathbf{A}[\mathtt{x}_i \neq \mathtt{x}_0] \cup \mathbf{A}[\mathtt{x}_t \neq \mathtt{x}_0]| \geq (1 + \frac{1}{k})|A_u|$, where A_u refers to the set of total features in which the instances in $\mathbf{C}$ differ from $\mathtt{x}_0$ (lines 7-9). Otherwise, $\mathbf{C}$ remains unchanged.

Algorithm $\widetilde{\text{sDBC}}$. We next present an algorithm, denoted by $\widetilde{\text{sDBC}}$ (Algorithm 3), for the streaming version of DBEP. Similar to $\widetilde{\text{dSBC}}$, $\widetilde{\text{sDBC}}$ works by maintaining and updating an explanation $\mathbf{C}$ as instances of $\mathbf{U_I}$ pass online. It only updates $\mathbf{C}$ with new instances that bring benefits to the succinctness of $\mathbf{C}$ while maintaining diversity lower bound p. To measure the benefits, it maintains an integer effectiveness for each feature A_i, denoted by $\mathbf{e}[A_i]$, of which the value changes when a new instance $\mathtt{x}_t$ arrives. $\widetilde{\text{sDBC}}$ identifies a set T of features from $\mathbf{A}[\mathtt{x}_t \neq \mathtt{x}_0]$ for $\mathtt{x}_t$, each with an effectiveness strictly less than $\lceil \log |T| \rceil$. It then refreshes $\mathbf{C}$ according to T.

More specifically, $\mathbf{C}$ is empty initially and the effectiveness of each feature is set to a minimum value (lines 1-3). Upon the arrival of $\mathtt{x}_t$, $\widetilde{\text{sDBC}}$ computes its T with largest cardinality (via procedure uCsuc; line 6). If T is empty, it keeps $\mathbf{C}$ intact and waits for the next instance of the stream (line 7). Otherwise, it updates $\mathbf{C}$ by adding $\mathtt{x}_t$ and sets the effectiveness of each feature of T to $\lceil \log |T| \rceil$

ALGORITHM 3: Algorithm $\widetilde{\text{sDBC}}$

Input: $\mathbf{U_I}$ as a stream with $\mathrm{x}_1, \mathrm{x}_2, \ldots$ arriving online, an instance x_0, integer p.
Output: a counterfactual explanation $\mathbf{C}$ when $\mathbf{U_I}$ ends.

1 $\mathbf{C} \leftarrow \emptyset$;
2 $\mathbf{A} \leftarrow \{1, \ldots, d\}$; // $\mathbf{A}$: *features of* $\mathcal{X}$, i.e., i *corresponds to feature* A_i
3 **foreach** $i \in \mathbf{A}$ **do** $\mathbf{e}[A_i] = -\infty$; // $\mathbf{e}[A_i]$ *is the effectiveness of feature* A_i
Upon the arrival of instance $\mathrm{x}_t (t \geq 1)$ of $\mathbf{U_I}$:
4 $\mathbf{C}, \mathbf{e} \leftarrow \texttt{uCsuc}(\mathbf{C}, \mathbf{e}, \mathrm{x}_t, p)$;
When the stream $\mathbf{U_I}$ ends:
5 **return** $\mathbf{C}$

Procedure $\texttt{uCsuc}(\mathbf{C}, \mathbf{e}, \mathrm{x}_t, p)$ // *Update* $\mathbf{C}$ *for each arriving* x_t
6 $T \leftarrow \arg\max_T \{|T| \mid T \subseteq \mathbf{A}[\mathrm{x}_t \neq \mathrm{x}_0] \text{ and } \mathbf{e}[A] < \log|T| (\forall A \in T)\}$
7 **if** $|T| = \emptyset$ **then continue** ;
8 **else**
9 $\mathbf{C} \leftarrow \mathbf{C} \cup \{\mathrm{x}_t\}$;
10 **foreach** $A_i \in T$ **do** $\mathbf{e}[A_i] = \lceil \log|T| \rceil$;
11 **foreach** $\mathrm{x}_j \in (\mathbf{C} \setminus \{\mathrm{x}_t\})$ **do** // *start removing redundancy*
12 **if** $\mathbf{A}[\mathrm{x}_j \neq \mathrm{x}_0] \setminus T = \emptyset$ **then** $\mathbf{C} \leftarrow \mathbf{C} \setminus \{\mathrm{x}_j\}$;
13 $\mathrm{x}_i \leftarrow \arg\min_{\mathrm{x}_i \in \mathbf{C}} |\mathbf{A}[\mathrm{x}_i \neq \mathrm{x}_0] \setminus \bigcup_{\mathrm{x}_j \in (\mathbf{C} \setminus \{\mathrm{x}_i\})} \mathbf{A}[\mathrm{x}_j \neq \mathrm{x}_0]|$;
14 **while** $|\bigcup_{\mathrm{x}_j \in (\mathbf{C} \setminus \{\mathrm{x}_i\})} \mathbf{A}[\mathrm{x}_j \neq \mathrm{x}_0]| \geq p$ **do**
15 $\mathbf{C} \leftarrow \mathbf{C} \setminus \{\mathrm{x}_i\}$;
16 $\mathrm{x}_i \leftarrow \arg\min_{\mathrm{x}_i \in \mathbf{C}} |\mathbf{A}[\mathrm{x}_i \neq \mathrm{x}_0] \setminus \bigcup_{\mathrm{x}_j \in (\mathbf{C} \setminus \{\mathrm{x}_i\})} \mathbf{A}[\mathrm{x}_j \neq \mathrm{x}_0]|$;
17 **return** $\mathbf{C}, \mathbf{e}$

(lines 8-10). Intuitively, by line 6, the effectiveness $\mathbf{e}[A]$ of each feature A in T is increased at least by 1 since T is at least twice as the size of T' in the last iteration that included and updated A. It adds x_t to $\mathbf{C}$ only when its T is non-empty, *i.e.,* at least one feature of which the effectiveness needs to increase. To make $\mathbf{C}$ more succinct, $\widetilde{\text{sDBC}}$ removes redundant instances from $\mathbf{C}$ in two phases. Initially, it discards those that only differ from x_0 over the features in T (lines 11-12). This is because these features are essentially embodied by the most recent x_t that yields a greater benefit. In phase two, it iteratively deletes instances from $\mathbf{C}$ with the fewest "private" features (recall $\widetilde{\text{dSBC}}$) until the diversity of $\mathbf{C}$ reaches the lower bound p (lines 13-16).

Analysis. Upon the arrival of each x_t, $\widetilde{\text{dSBC}}$ and $\widetilde{\text{sDBC}}$ take $O(kd)$-time and $O(d^2)$-time, respectively, to maintain the counterfactual explanation $\mathbf{C}$, where d is the number of features. The space complexity of $\widetilde{\text{dSBC}}$ and $\widetilde{\text{sDBC}}$ is only $O(kd)$ and $O(d^2)$, respectively. Note that both k and d are quite small in practice. The negligible time and space complexity justifies that they can support CPC well for real-time applications with stringent resource constraints. Besides the practicality, they also correctly implement the reductions and offer performance guarantees for streaming CPC.

Lemma 6: *(1)* $\widetilde{\text{dSBC}}$ *is 4-competitive for streaming* SBEP. *(2)* $\widetilde{\text{sDBC}}$ *is* $4\sqrt{2d}$*-competitive for streaming* DBEP, *assuming* $p \geq d$. □

Indeed, one can verify that (a) procedures `uCdiv` and `uCsuc` correctly implement reductions from SBEP and DBEP to MCR and MSC, respectively, and (b) lines 6, 8 and lines 6, 10 of `uCdiv` and `uCsuc` respectively maintain invariants of upper bound conditions of streaming MCR [18] and MSC [19] (details omitted). (c) We further extend classic streaming MSC [19] with steps to eliminate potentially redundant counterfactuals (lines 11-16 of $\widetilde{\text{sDBC}}$) so as to optimize succinctness of $\mathbf{C}$;

datasets	#-instances	#-features	task
Adult [67]	45,222	13	predict whether an adult earns ≥ 50K a year
Compas [2]	6,172	7	score criminal defendant's likelihood of re-offending based on the COMPAS algorithm
Credit [1]	115,527	10	predict whether someone will experience finance distress in the next two years by financial information
Loan [4]	614	11	predict whether to approve a loan application

Table 1. Datasets and their prediction tasks

these steps do not affect the effectiveness of features, thereby maintaining the performance bounds.

Remarks. (1) We clarify that our main goal is to show that there exist effective algorithms to support CPC in the streaming mode, and we do this by "reversing" the reductions in the proof of Theorem 2 to adapt and optimize classic streaming MCR [18, 76] and MSC [19, 55] algorithms for CPC.

(2) CPC is versatile and can be used with other variants of streaming MCR (*e.g.,* [76, 123]) and MSC (*e.g.,* [55]). It remains robust when deployed with such alternative variants (see Section 7.5).

7 EXPERIMENTS

In this section, we experimentally evaluate the effectiveness of CPC. Below we first specify the settings (Section 7.1). We then evaluate the capability of CPC to find high quality counterfactual explanations while defending model extraction attacks (Section 7.2- 7.6).

7.1 Experimental Setting

Datasets. We used 4 real-world datasets from the Kaggle contest portal [3] and the UCI repository [5] that are commonly used for evaluating explanations [31, 54, 85] (see Table 1 for details). For each dataset, we designated non-actionable features to ensure that counterfactuals do not alter their values. Specifically, the non-actionable features are set as follows: {age, sex, race} for Adult, {age} for Credit, {age, sex, race} for Compas and {gender} for Loan.

Compared methods. As shown in Table 2, we compared with 11 state-of-the-art methods.

(1) Counterfactual explanation methods. We tested 4 popular counterfactual explanation methods, implemented using the Carla library [88] with the default configurations recommended by Carla. Among them, DICE is capable of generating multiple counterfactuals as an explanation for a query instance. In contrast, GS, FACE and CCHVAE only produce explanation of singleton counterfactual. To address this, we repeatedly executed these three methods multiple times with varied random seeds for their parameters; this yields explanation of multiple counterfactuals, consistent with DICE.

(2) CPC *variants*. We also tested 7 variants of CPC where we use alternative methods for SBEP and DBEP in CPC, as follows.

(a) Data selection. $\text{CPC}_{\text{MaxMin}}$ and $\text{CPC}_{\text{MaxSum}}$ are two variants of CPC that employ MaxMin and MaxSum, respectively, for selecting a diverse explanation from $\mathbf{U_I}$, where MaxMin (resp. MaxSum) is a recent data selection method that is shown effective in maximizing the minimum (resp. total) distance among the chosen data [24, 86].

(b) Package queries. We also employed package queries [29, 75] to select explanations from $\mathbf{U_I}$ in CPC. A package query returns a set of instances meeting predefined single and global constraints; it scales by employing ILP solvers structurally via query decomposition.

(c) Max-cover & Set-cover. Denote by CPC_{GSV}, CPC_{GOPS} and CPC_{DR} the CPC variants that use

State-of-the-art counterfactual explanation methods	
methods	**description**
DICE [85]	return a set of plausible and different counterfactuals by solving an optimization problem with various constraints to ensure feasibility and diversity
FACE [93]	obtain counterfactuals by building data graphs based on user-defined density metrics, and finding optimal paths consistent with data distribution
GS [70]	rely on a generative heuristic approach growing a sphere of synthetic instances around query instance to find the closest counterfactual
CCHVAE [89]	utilize an auto-encoder to generate counterfactuals in the latent space without needing a real-space distance function and a complex decoder
CPC *variant with Diverse Data Selection [24, 86]*	
methods	**description**
$\text{CPC}_{\text{MaxMin}}$	MaxMin [86]: select k instances and maximize the minimum pairwise distance
$\text{CPC}_{\text{MaxSum}}$	MaxSum [24]: similar to MaxMin, but maximize the sum of pairwise distances
CPC *variant with Package Queries [29]*	
method	**description**
CPC_{PQ}	use package queries [29] for solving SBEP and DBEP in CPC
CPC *variants with Max-Cover [76, 123] and Set-Cover [55]*	
methods	**description**
CPC_{GSV}	GSV [76]: a single-pass streaming algorithm for approximating maximum k-set coverage, using the "guess, subsample, and verify" framework
CPC_{GOPS}	GOPS [123]: a streaming MCR algorithm with the aid of computer simulation
CPC_{DR}	DR [55]: a streaming MSC algorithm based on iterative dimensionality reduction
CPC *variant with optimal search*	
method	**description**
CPC_{OPT}	optimal answers to SBEP and DBEP in CPC via brute-force enumeration

Table 2. Compared explanation methods and CPC variants

GSV [76], GOPS [123] and DR [55], respectively, for solving SBEP and DBEP when CPC operates in streaming explanation mode. Both GSV and GOPS are single-pass streaming algorithms for approximating maximum k-set coverage, while DR targets minimum set cover. The first two are viable alternatives to $\widetilde{\text{dSBC}}$ and DR is a practical substitute for $\widetilde{\text{sDBC}}$.

(d) Optimum. CPC_{OPT} is a CPC variant that uses brute-force enumeration to find optimal solutions to both SBEP and DBEP in CPC.

We remark that all these 4 counterfactual methods rely on the ML model M from model owners or service providers to deduce counterfactual explanations, *i.e.,* they are not client-centric and users have no control of whether and how they can obtain the explanation. CPC and all its variants run on explanation universe $\mathbf{U_I}$ that consists of instances already pass the application-specific constraints (*e.g.,* sparsity, similarity) on individual counterfactuals. $\text{CPC}_{\text{MaxMin}}$ and $\text{CPC}_{\text{MaxSum}}$ focus exclusively on optimizing diversity and operate in batch mode. On the other hand, CPC_{GSV}, CPC_{GOPS} and CPC_{DR} are tailored for streaming mode, with the first two focusing on maximizing diversity and the last one on optimizing succinctness.

Measures. We assessed the efficiency of each method by the average time it takes to generate the explanation for a single instance.

In addition, we also evaluated the quality of counterfactual explanations computed by all methods and the degree of privacy leakage in their counterfactuals *w.r.t.* model extraction attacks.

(1) Measuring explanation quality. We used four measures to quantify the quality of counterfactual explanations. Consider a counterfactual explanation $\mathbf{C}$ for an instance x. The first two measures assess the quality of $\mathbf{C}$ as a whole.

(a) Diversity [37, 85]. the diversity of $\mathbf{C}$ is the total number of distinct features of counterfactuals in $\mathbf{C}$ that differ from those of x.

(b) Succinctness [54, 113]: the number of counterfactuals in $\mathbf{C}$.

We also evaluated the quality of each individual counterfactual in the explanation. Consider a counterfactual $\mathsf{x}' \in \mathbf{C}$ for x.

(c) Similarity: the similarity between x and x' is measured by the Gower distance [37], which is defined as $\frac{1}{d}\sum_{i=1}^{d}\delta(\mathsf{x}[A_i], \mathsf{x}'[A_i])$. δ depends on the feature type: $\delta(\mathsf{x}[A_i], \mathsf{x}'[A_i]) = |\mathsf{x}[A_i] - \mathsf{x}'[A_i]|/\widehat{A_i}$ if A_i is numerical with $\widehat{A_i}$ as the value range of A_i. If A_i is categorical, $\delta(\mathsf{x}[A_i], \mathsf{x}'[A_i]) = I(\mathsf{x}[A_i] \neq \mathsf{x}'[A_i])$ where I is indicator function.

(d) Sparsity [31]: the sparsity of x is $\sum_{i=1}^{d} I(\mathsf{x}[A_i] \neq \mathsf{x}'[A_i])$.

The diversity of a counterfactual explanation should not exceed the number of actionable features in the dataset; similarly for the sparsity of an individual counterfactual. The similarity varies between 0 and 1. For the succinctness, similarity and sparsity, smaller values are preferable; conversely, the larger the better for diversity.

(2) Measuring privacy leakage. We start with the attacks.

Attack strategy. We adopted dualCF [117], a recent model extraction attack strategy that utilizes counterfactuals. It uses both the counterfactual and its subsequent counterfactual (referred to as CCF) as training pairs for training the substitute model M'. Specifically, dualDF first sends a query instance x to the counterfactual method and retrieves a counterfactual x'. Following this, x' itself is used as a new query to derive its CCF, x''. Using this procedure, dualDF can generate a set of counterfactuals and CCFs, which, together with the inference set, serve as a training set to create M'. Intuitively, the counterfactual and CCF sit on opposite sides of the decision boundary of the original M. Furthermore, both the counterfactual and CCF naturally have similar distances to the boundary as they are derived from the same counterfactual method. Together with inference set, this assists adversaries in training M' that closely resembles M.

Privacy metrics. To measure the privacy leakage of counterfactual explanations against model extraction attacks, we compare the performance of the surrogate model M' with that of the original model M over an evaluation set $\mathcal{X}_e \subseteq \mathcal{X}$. We use two metrics [12, 117].

(a) Agreement: the agreement of M' is defined as $\frac{1}{|\mathcal{X}_e|}\sum_{\mathsf{x}\in\mathcal{X}_e} I(M(\mathsf{x}) = M'(\mathsf{x}))$ where I is the indicator function, *i.e.,* 1 if $M(\mathsf{x}) = M'(\mathsf{x})$. It measures the prediction similarity between M and M' over $\mathcal{X}_e$. A higher agreement indicates that M' behaves more similarly to M.

(b) Discrepancy: the discrepancy of M' measures how different its predictions are to those of M over $\mathcal{X}_e$. It is defined as $\frac{1}{|\mathcal{X}_e|}|\sum_{x\in\mathcal{X}_e} I(M(x) = y) - \sum_{x\in\mathcal{X}_e} I(M'(x) = y)|$ where y is the true label for x.

Intuitively, discrepancy is an overall assessment of the quality of M' without requiring M' and M to make the same prediction for every instance. Instead, this specific requirement is exactly what the agreement measures. A higher agreement or lower discrepancy signifies a more successful model extraction attack by the adversary.

Configuration. We deployed CPC on a machine with an Intel Core i7-6700HQ CPU @2.6GHz and 32 GB of RAM as the client, and used AWS EC2 m5.2xlarge instance to host the remote model M.

For each dataset, a 60/30/10 ratio was used to split it into 3 disjointed sets: training, inference

	Income	CoIncome	LoanAmount	LoanTerm	Gender	Married	Credit	Urban	Prediction
x_0	6,000	0	20,000	360	Female	No	fair	No	Denied
				CPC					
x'_{01}	-	-	-	-	-	Yes	good	-	Approved
x'_{02}	-	3,000	-	-	-	-	good	-	Approved
x'_{03}	-	-	10,000	-	-	-	good	-	Approved
				DICE					
x'_{01}	-	-	-	39.2	-	-	good	-	Approved
x'_{02}	-	4,800	-	462.9	-	-	good	-	Approved
x'_{03}	-	-	-	312.8	-	-	good	-	Approved
				FACE					
x'_{01}	8,000	-	30,000	-	Male	Yes	good	Yes	Approved
x'_{02}	-	-	-	-	Male	Yes	good	Yes	Approved
x'_{03}	-	3,000	-	0	Male	Yes	good	-	Approved
				GS					
x'_{01}	6,007.8	4.9	20,067.1	360.1	-	-	good	-	Approved
x'_{02}	5,983.1	0.3	20,087.7	361.4	-	-	good	-	Approved
x'_{03}	5,972.2	7.6	20,658.9	355.9	-	-	good	-	Approved
				CCHVAE					
x'_{01}	6,108.1	2,154.9	30,370.9	287.3	Male	Yes	good	Yes	Approved
x'_{02}	5,766.4	2,677.9	23,418.2	333.7	Male	Yes	good	-	Approved
x'_{03}	5,676.3	1,949.7	27,404.6	195.1	Male	Yes	good	-	Approved

Table 3. Counterfactual explanations in the case study

and evaluation sets. The training set is used only for training the target model M. Specifically, we trained a Multilayer Perceptron (MLP) M for each dataset using the Adam optimizer to minimize the binary cross-entropy loss. For the Adult and Credit datasets, M comprises six hidden layers with 512, 288, 144, 72, 36, 9 and 3 neurons, while for the Compas and Loan datasets, M has three hidden layers with 36, 9 and 3 neurons. The inference set contains instances that were explained, and also those used by the adversary to extract the surrogate model M' via dualCF attacks. The evaluation sets are used to gauge the accuracy of both M and M', and to measure how well M' is aligned with M. Unless explicitly stated otherwise, CPC is equipped by default with dSBC and uses the complete inference set as the explanation universe.

All compared methods are tested with their default parameters. Moreover, all existing counterfactual explanation methods have unlimited access to the target model M and its training set, which CPC has none. In the tests, we set the similarity threshold for each counterfactual to 0.3, and sparsity to 4 for Credit and 2 for the other three datasets. The default succinctness and diversity bounds are 5 and 4, respectively, for all datasets. Each experiment was run three times, and the average is reported.

7.2 Case Study and User study

We first illustrated the benefits of CPC over the four counterfactual explainers via a case study. We then conducted a user study to evaluate how humans perceive their explanations in practice.

Case study. We start with a case study.

Right-to-explain. CPC gives users the right to access counterfactual explanation even when the model owner does not. As an example, we hosted the original model M over Loan dataset on Amazon SageMaker [6] as a remote model inference service that charges by the number of queries to the remote model. SageMaker provides neither counterfactual explanation nor APIs for computing

	Questions	Property
Practicality	**1** Considering feature changes and values in a *single* explanation, do the suggestions make sense for loan approval?	Similarity
	2 Given the number of changed features in a *single* explanation, do you think these explanations could be realistic to act?	Sparsity
	3 In your opinion, does the total number of distinct features in a *set* of counterfactuals provide you with sufficient options?	Diversity
	4 Given the instance and its all counterfactuals, how do you think they adhere to causal consistency?	Causal Consistency
	5 Which method do you prefer for a counterfactual explanation of the ML model's decision?	
Contextuality	**6** Do you accept the scenario where the same instance can have different counterfactuals at different times?	Acceptability
	7 Does the variability of counterfactual explanations across different times affect your trust in the model's decision?	Trustworthiness
	8 Do you think these varying counterfactuals provide enough stability for loan approval?	Stability
	9 Does the pattern reflected in explanations at different times help in guiding next loan strategy?	Guidance
	10 Do explanations based on a specified context provide more useful information and insights?	Insightfulness

Table 4. Questions in the user study

counterfactuals. Despite this, CPC still generates counterfactual explanations, *e.g.,* for x_0 of Table 3 at the client side, incurring no cost from SageMaker. In contrast, all other counterfactual methods require M to be open to "explanation queries" that invoke M over candidate counterfactuals, which also incurs additional cost even if the owner permits.

Privacy preservation. Since CPC ensures that its counterfactuals come solely from the inference set that is already known to the users, it does not reveal any extra information about the ML model. In contrast, counterfactuals from DICE, FACE, GS and CCHVAE can be seen as "new" training instances. Adversaries could incorporate these new instances into their existing inference set in their attacks, creating a surrogate model with high accuracy (see Section 7.3).

Quality. We further illustrate the quality of explanations for instance x_0 over Loan that has a denied decision. Assume a user wants to generate explanation C with 3 counterfactuals and the sparsity of each is no more than 2. To allow all methods to explain correctly, we deployed the model at the client-side and allowed unlimited model access for all methods except for CPC. Table 3 shows x_0 and its explanations by each method; only changed features are shown.

(1) Controllability. CPC offers strict controls on the validity of the counterfactuals. For instance, FACE, GS and CCHVAE do not meet the sparsity bound while CPC does. Moreover, FACE and CCHVAE modify immutable features, *e.g.,* `Gender`, which is clearly not viable.

(2) Relevance. CPC draws counterfactuals from real instances predicted during inference, which are more relevant than methods that generate counterfactuals from the feature space, which are artificial and unrealistic. For example, counterfactuals of GS and CCHVAE are less feasible since in financial transactions [39], `LoanTerm` cannot be arbitrarily specified, especially not as a random decimal.

(3) Quantification. The diversity of the explanation by CPC is 4, higher than 3 for DICE. The results

of FACE, GS and CCHVAE are invalid as they do not satisfy user specific requirements on sparsity.

User Study. We further conducted a user study to evaluate CPC and 4 counterfactual explanation methods regarding (1) *practicality*: which method produces counterfactuals that are easier for users to understand and accept, and if they are causally consistent, and (2) *contextuality*: how do users perceive counterfactuals that may vary over time or be dependent of user-specified contexts?

To this end, we designed an online questionnaire[1] using Google Forms. The first page (page 1) of the survey asked about the participants' professional background. Considering that counterfactual methods should be accessible to non-technical users without ML expertise, we recruited 38 participants, half familiar with ML terminologies and the other half less so. Page 2 introduced the basics of ML models and counterfactuals. We randomly selected 10 instances with denied ML decision from Loan dataset and presented explanations generated by each counterfactual method, similar to those in the case study. Each survey evenly and randomly included 3 instances. To ensure impartiality, the methods were anonymized so that participants were unaware of the specific methods used.

Design. We next introduce the remaining part of the survey in detail.

(1) Practicality. Following [82, 106], we designed 5 questions that are aligned with practicality properties (*i.e.,* similarity, sparsity, diversity and causal consistency), to assess counterfactuals' practicality.

As shown in Table 4, Questions 1-4 (Q1-Q4) are measured by a 5-point Likert scale (higher scores indicate better performance), and Question 5 is a single-choice query. Q1-Q4 were presented without direct comparison between different methods. We first evaluated explanations computed by CPC, then asked the same questions for explanations by DICE, FACE, GS and CCHVAE, in that order. This forms pages 3-7 of the survey. Page 8 included Question 5, which directly asked respondents to choose their preferred method.

(2) Contextuality. We then assessed how users perceive contextual counterfactuals by CPC for the same instance with different contexts as the explanation universe. To create these contexts, we categorized instances based on different time and features. For time-dependent contexts, we evenly divided Loan sorted in ascending order of `Income` into four non-overlapping batches (assuming that people's income increases over time). For the feature-grouped contexts, we created different batches by specifying multiple expected ranges for `Income` and `LoanAmount`. Following [69, 123], as shown in Table 4, we designed 5 questions from various aspects including acceptability, trustworthiness, stability, guidance and insightfulness. Q6-Q9 and Q10 were on page 9 and 10 of the survey, respectively. All the questions were evaluated using the 5-point Likert scale.

Results. From the gathered feedback, we found the following.

(1) Overall, out of the 38 answers, a significant 28 users chose CPC as their preferred method (Q5). For Q1, on average both CPC and FACE scored over 4 on a 5-point Likert scale, while DICE only received a score of 3.4, and GS and CCHVAE scored even less than 2. This indicates that users generally find counterfactuals generated by CPC and FACE to be relevant and meaningful. However, FACE scored only 1.7 in Q2, whereas CPC and DICE scored much higher, at 3.9 and 3.7 respectively. This highlights the importance of sparse explanations for practical user actions. Moreover, in Q3 and Q4, 89.5% of users felt that CPC provided a sufficiently diverse range of feature options (*i.e.,* scoring ≥ 4), and 86.1% of users believed these counterfactuals were causally consistent. These findings demonstrates that among all compared counterfactual explanation methods, CPC offers the most realistic and practical suggestions.

(2) When CPC's counterfactuals vary over time, on average the scores for Q6-Q8 were 4.1, 4.2 and

[1] https://forms.gle/DzRLW9hXTwAGUZM66

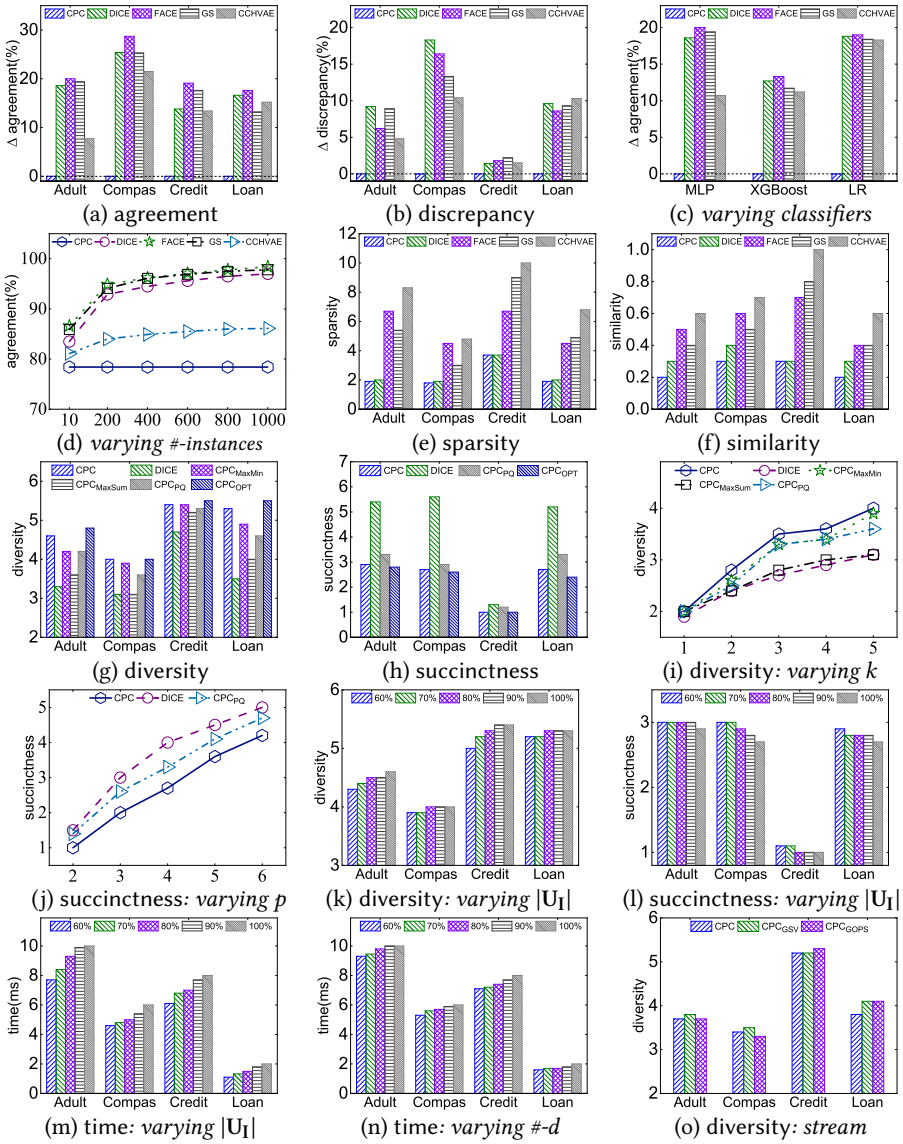


Fig. 3. The effectiveness and quality of guarded counterfactual

3.5, respectively. This indicates that most participants still accept and trust the system in such scenarios, even though some find CPC's explanations not somewhat stable. More importantly, a notable 71.1% of users believed the potential pattern reflected in explanations at different times could be beneficial in improving their future strategy (Q9).

(3) Users recognized the benefit of CPC's contextual counterfactuals that are tailored to scenarios with specific ranges of picked features. Specifically, almost all participants believed that these contextual counterfactuals delivered more useful information and deeper insights, with 35 out of 38 responses scored over 4 on Q10.

7.3 Defending Attacks over Counterfactuals

We next evaluated the effectiveness of all counterfactual methods in defending against model

extraction attacks over explanations.

Counterfactual contribution. Note that model extraction attacks also exploit inference set during model inference, even without the availability of counterfactuals. To eliminate the noise from inference instances and assess the privacy leakage of counterfactuals, we also extract a reference model M_r by using only the inference instances, and report the difference between the quality metrics (agreement and discrepancy) of the fully extracted model M' and those of M_r; We refer to such difference as *counterfactual contribution*.

Intuitively, M_r is the "best" privacy-preserving model that leaks zero information via explanation as it sees no counterfactuals at all; hence, the gap between M' and M_r quantifies the degree of contribution from counterfactual explanations to the quality of the extracted models. For both agreement and discrepancy, the lower the contribution of an explanation method is the better it can defend model extraction attacks. When training M' (and M_r), we use a default three hidden-layer MLP classifier, distinct from the original model M.

Defend. We first examined the counterfactual contribution to the agreement and discrepancy of extracted model M' by each method (denoted by Δagreement and Δdiscrepancy). As shown in Figures 3a- 3b, in all cases CPC has consistently zero contribution to both agreement and discrepancy of M' over reference model M_r, while on average that of DICE, FACE, GS and CCHVAE is 18.6% and 9.6%, 21.4% and 8.3%, 18.9% and 8.4%, and 14.5% and 6.8%, respectively. This shows that explanations by CPC are privacy-preserving and can prevent adversaries from exploiting counterfactuals to extract models, while existing methods cannot. This verifies Proposition 1.

Ablation study. We also evaluated the impact of test parameters.

(1) Classifier. In addition to MLP, we trained M' with XGBoost [35] and Logistic Regression (LR) as alternatives. Fig 3c shows counterfactual contribution to agreement by all methods over Adult.

(a) CPC is robust against the type of classifiers that the adversary employs to extract models, *e.g.,* contributing 0 consistently in all cases. In contrast, FACE contributes 17.4% in the agreement of the extracted models; similarly for other methods.

(b) The raw agreement of all compared methods is consistently above 88% for all classifiers (not shown in Fig 3c). This suggests that, regardless of the exact classifier, an adversarial user can always use counterfactuals to help effectively train M' that mirrors M's behavior, even with a classifier as primitive as LR. This finding underlines the pressing need of privacy preserving counterfactuals.

(2) Varying # of query instances. We tested how agreement is affected by the number of instances (#-instance) queried by the adversarial user. Fig 3d depicts the results on Adult. The result of CPC is constant as it does not query (*i.e.,* access) to the model for explanation. In contrast, the agreement of M' trained by the four compared methods increases as #-instance grows. Moreover, even with very low #-instance, the adversary can still achieve high agreement model extractions. For instance, on Adult with just 200 query instances, the M' trained using DICE, FACE, GS and CCHVAE attains agreement of 92.9%, 94.9%, 94.1% and 84.0%, respectively.

7.4 Quality and Efficiency

Quality. Using the four measures defined in Section 7.1, we next evaluated the quality of counterfactual explanations of all methods.

(1) Sparsity and similarity. As shown in Figures 3e- 3f, except for CPC and DICE, counterfactuals of FACE, GS and CCHVAE consistently violate the pre-defined similarity and sparsity thresholds. While the sparsity of DICE is comparable to that of CPC when they both optimize diversity, the similarity and diversity of DICE are both much worst, *e.g.,* on Loan, the Gower distance of

counterfactuals of DICE exceeds that of CPC by 33.3%. This is because DICE cannot flexibly optimize diversity while also bounding similarity; it works by using a higher weight to diversity in its objective function when maximizing diversity, which results in reduced similarity.

(2) Diversity and succinctness. CPC consistently outperforms all the compared methods on all datasets in diversity and succinctness. As depicted in Fig 3g, on average the diversity of CPC exceeds that of DICE and CPC variants, $\text{CPC}_{\text{MaxMin}}$, $\text{CPC}_{\text{MaxSum}}$ and CPC_{PQ}, by 33.7%, 5.1%, 23.3% and 9.4% across all datasets respectively. Moreover, its diversity gap from the optimal solution CPC_{OPT} is negligible, only 2.4% on average. On the other hand, as shown in Fig 3h, CPC (sDBC) is also the best when users prioritize succinctness. We remark that $\text{CPC}_{\text{MaxMin}}$ and $\text{CPC}_{\text{MaxSum}}$ are not suitable for optimizing succinctness. Compared to the strongest competitor, CPC_{PQ}, CPC's explanations are 12.1%, 6.9%, 16.7% and 18.2% more succinct over Adult, Compas, Credit and Loan respectively. Similarly, CPC is comparable to CPC_{OPT} in optimizing succinctness.

(3) Impact of k and p. We also evaluated the impact of the succinctness bound (desired counterfactuals k) and the diversity bound (desired changed features p) on the performance of CPC. Fig 3i depicts the diversity of CPC (dSBC) on Compas when varying k from 1 to 5. Fig 3j shows the succinctness of CPC (sDBC) when p ranges from 2 to 6 on Loan. We observe that for all values of k, CPC consistently outperforms all the competitors in diversity, and the benefit of CPC becomes even more evident when k increases. For instance, the gap between CPC and DICE grows from 0.4 to 0.8 when k varies from 2 to 5. Similarly, CPC is the most succinct. On average, the succinctness of CPC surpasses DICE and CPC_{PQ} by 27.0% and 18.5% respectively. This verifies that CPC can reliably deliver high-quality explanations, catering to various values of k and p.

(4) Impact of $|\mathbf{U_I}|$. Finally, we evaluated how the size $|\mathbf{U_I}|$ of inference universe $\mathbf{U_I}$ impacts the quality of counterfactuals of CPC. Varying $|\mathbf{U_I}|$ from 60% to 100% of the inference set of all datatsets, we tested the diversity and succinctness. As shown in Fig 3k-3l, we observe that (1) with the increase of $|\mathbf{U_I}|$, both diversity and succinctness improve as expected; (2) even when $|\mathbf{U_I}|$ is at 60%, on average CPC still outperforms all compared methods in diversity and succinctness by 12.7% and 22.1% respectively (Figures 3g- 3h); and (3) CPC is robust against varying $|\mathbf{U_I}|$. For instance, on average the diversity decreases by only 4.6% when $|\mathbf{U_I}|$ is reduced from 100% to 60%.

Efficiency. We also tested the efficiency. The average time (ms) for explaining an instance over all datasets is shown in the table below.

datasets	existing explainers					CPC variants			
	FACE	GS	DICE	CCHVAE	**CPC**	MaxMin	MaxSum	PQ	OPT
Adult	433	2012	143	8000	10	24	21	70	599
Compas	205	284	36	2989	6	19	18	62	476
Credit	210	956	81	9001	8	26	26	72	686
Loan	113	150	14	455	2	11	10	6	394

Compared to 4 existing counterfactual methods, on average across all datasets CPC is 40.1, 110.8, 9.4, and 662.7 times faster than DICE, FACE, GS and CCHVAE, respectively. This is because these methods need to access the ML model for counterfactual candidates, which is not required by CPC and its variants. Moreover, CPC is on average 30.9 times faster than its variants, up to 197 times. This shows that CPC can efficiently compute high-quality counterfactual explanations on-the-fly, making it ideal for real-time scenarios.

Impact of $|\mathbf{U_I}|$ and #-d. We further examined CPC's efficiency by varying (1) the inference universe size $|\mathbf{U_I}|$ from 60% to 100% of the inference set; and (2) the feature number #-d (rounded to the nearest whole number) from 60% to 100% of the total feature number of each dataset. The results are shown in Figures 3m-3n. As expected, the running time of CPC increases when $|\mathbf{U_I}|$ or #-d grows, but the increment is minor and acceptable. For instance, over the Adult dataset, the time

increases by only 11.6% when $|\mathbf{U_I}|$ is raised from 80% to 100%. This shows that CPC is scalable. We also observe that $|\mathbf{U_I}|$ has a greater impact on the performance of CPC than #-d, which is consistent with the time complexity analysis in Section 5.

7.5 Streaming Counterfactual Explanation

Finally, we evaluated the capability of counterfactual for resource-constrained clients that have no memory to cache relevant inference instances, by operating CPC in the streaming mode.

Using the same inference universe as in previous experiments, we compared CPC ($\widetilde{\text{dSBC}}$ and $\widetilde{\text{sDBC}}$) with its streaming variants, *i.e.,* CPC_{GSV}, CPC_{GOPS} and CPC_{DR}. We also examined CPC's streaming mode by comparing against its batch mode (dSBC and sDBC).

(1) CPC ($\widetilde{\text{dSBC}}$) is comparable to CPC_{GSV} and CPC_{GOPS} for diversity (Fig 3o); similarly for CPC ($\widetilde{\text{sDBC}}$) and CPC_{DR}. This highlights the versatility of CPC and verifies that it is robust when extended with varying methods. On the other hand, $\widetilde{\text{dSBC}}$ and $\widetilde{\text{sDBC}}$ is more memory-efficient, *e.g.,* $\widetilde{\text{dSBC}}$ uses on average only 44.2% of the memory used by GSV and 34.7% by GOPS across all datasets.

(2) Compared to the batch mode, even though the diversity of $\widetilde{\text{dSBC}}$ is lower than dSBC by 17.8% on average over all 4 datasets, it still surpasses DICE by 10.3%. Similarly, $\widetilde{\text{sDBC}}$ is more succinct than DICE by 33.3% over Adult, albeit CPC is 19.4% more succinct in its explanation with $\widetilde{\text{sDBC}}$. However, the slight dip in diversity and succinctness gives CPC a significant benefit in space cost, *e.g.,* $\widetilde{\text{dSBC}}$ saves 98.2% of memory cost of dSBC on average; similarly for $\widetilde{\text{sDBC}}$.

7.6 Summary

We find the following. (1) CPC offers model users the right to explain arbitrary models even when model owners opt not to. For cases that model owners offer the interface for counterfactual explanation, CPC is on average 205.7 times faster than existing methods. (2) Counterfactual explanation of CPC does not leak any privacy of the target model other than instances to be explained. (3) These benefits do not sacrifice the quality of counterfactuals: compared to state-of-the-art, CPC is 35.5% higher in diversity and 39.5% more succinct. (4) CPC is versatile. It can be well extended with existing methods to fit varying applications, including streaming scenarios.

8 CONCLUSION

We presented CPC, a data-driven approach to counterfactual explanation. It addresses two open challenges that hinder the adoption of counterfactuals in practice, by (a) ensuring model users the right to access counterfactual for arbitrary models, *e.g.,* those hosted at Amazon SageMaker, and (b) preventing model extraction attacks from exploiting counterfactuals. We formulated its properties and studied underlying computational problems. Using public datasets, we experimentally verified its effectiveness, efficiency and quality of explanation by comparing with state-of-the-art methods.

This work potentially opens a new, data-driven direction towards explainable AI. One topic for future work is to extend CPC with causality-aware explanations [48, 58, 91, 99]. Another direction is to adopt package queries [29, 75] for in-database support of CPC with extensions to take into account *e.g.,* fairness [40, 86]. A third topic is to give a formal treatment to contextual explanation and to characterize the implications of explanation universes on explainability.

ACKNOWLEDGMENTS

This work is supported by RAEng RF\201920\19\319 and the Huawei-Edinburgh Joint Lab.

REFERENCES

[1] 2020. Credit dataset. https://github.com/DrIanGregory/Kaggle-GiveMeSomeCredit.
[2] 2022. Compas dataset. https://www.kaggle.com/datasets/danofer/compass.
[3] 2022. Kaggle. https://www.kaggle.com/.
[4] 2022. Loan dataset. https://www.kaggle.com/datasets/vikasukani/loan-eligible-dataset.
[5] 2022. UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/index.php.
[6] 2023. Amazon SageMaker. https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-model-explainability.html.
[7] 2023. Google cloud. https://cloud.google.com/.
[8] 2023. Microsoft Azure Machine Learning. https://azure.microsoft.com/en-gb.
[9] 2023. Upstart. https://www.upstart.com/.
[10] 2023. Zest AI. https://www.zest.ai/.
[11] Raghavendra Addanki, Andrew McGregor, Alexandra Meliou, and Zafeiria Moumoulidou. 2022. Improved Approximation and Scalability for Fair Max-Min Diversification. In *ICDT*, Vol. 220. 7:1–7:21.
[12] Ulrich Aïvodji, Alexandre Bolot, and Sébastien Gambs. 2020. Model extraction from counterfactual explanations. *arXiv preprint arXiv:2009.01884* (2020).
[13] Susanne Albers. 2003. Online algorithms: a survey. *Mathematical Programming* 97, 1 (2003), 3–26.
[14] Emanuele Albini, Shubham Sharma, Saumitra Mishra, Danial Dervovic, and Daniele Magazzeni. 2023. On the Connection between Game-Theoretic Feature Attributions and Counterfactual Explanations. In *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*. 411–431.
[15] Shuai An and Yang Cao. 2024. Relative Keys: Putting Feature Explanation into Context. *Proc. ACM Manag. Data* 2, 1, Article 8 (mar 2024), 28 pages.
[16] Javier Antorán, Umang Bhatt, Tameem Adel, Adrian Weller, and José Miguel Hernández-Lobato. 2020. Getting a clue: A method for explaining uncertainty estimates. *arXiv preprint arXiv:2006.06848* (2020).
[17] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information fusion* 58 (2020), 82–115.
[18] Giorgio Ausiello, Nicolas Boria, Aristotelis Giannakos, Giorgio Lucarelli, and V Th Paschos. 2012. Online maximum k-coverage. *Discrete Applied Mathematics* 160, 13-14 (2012), 1901–1913.
[19] Michael Barlow, Christian Konrad, and Charana Nandasena. 2021. Streaming Set Cover in Practice. In *ALENEX*. 181–192.
[20] Roberto J Bayardo and Rakesh Agrawal. 2005. Data privacy through optimal k-anonymization. In *ICDE*. IEEE, 217–228.
[21] Nicole Bidoit, Melanie Herschel, and Katerina Tzompanaki. 2016. Refining SQL Queries based on Why-Not Polynomials. In *TAPP*.
[22] C Bishop. 2006. Pattern recognition and machine learning. *Springer google schola* 2 (2006), 35–42.
[23] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ B. Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. 2021. On the Opportunities and Risks of Foundation Models. *CoRR* abs/2108.07258 (2021).
[24] Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *SIGMOD*. 155–166.
[25] Tomas Borovicka, Marcel Jirina Jr, Pavel Kordik, and Marcel Jirina. 2012. Selecting representative data sets. In *ICDM*, Vol. 12. 43–70.
[26] El Bachir Boukherouaa, Mr Ghiath Shabsigh, Khaled AlAjmi, Jose Deodoro, Aquiles Farias, Ebru S Iskender, Mr Alin T Mirestean, and Rangachary Ravikumar. 2021. *Powering the Digital Economy: Opportunities and Risks of Artificial Intelligence in Finance*. International Monetary Fund.
[27] Tim Brennan and William L Oliver. 2013. Emergence of machine learning techniques in criminology: implications of complexity in our data and in research questions. *Criminology & Pub. Pol'y* 12 (2013), 551.
[28] Matteo Brucato, Azza Abouzied, and Alexandra Meliou. 2018. Package queries: efficient and scalable computation of high-order constraints. *The VLDB Journal* 27 (2018), 693–718.
[29] Matteo Brucato, Juan Felipe Beltran, Azza Abouzied, and Alexandra Meliou. 2016. Scalable Package Queries in

Relational Database Systems. *Proc. VLDB Endow.* 9, 7 (2016), 576–587.
[30] Matteo Brucato, Miro Mannino, Azza Abouzied, Peter J Haas, and Alexandra Meliou. 2020. sPaQLTooLs: a stochastic package query interface for scalable constrained optimization. *Proc. VLDB Endow.* (2020).
[31] Dieter Brughmans, Pieter Leyman, and David Martens. 2023. Nice: an algorithm for nearest instance counterfactual explanations. *Data Mining and Knowledge Discovery* (2023), 1–39.
[32] Peter Buneman, Sanjeev Khanna, and Wang Chiew Tan. 2001. Why and Where: A Characterization of Data Provenance. In *ICDT*.
[33] Peter Buneman and Wang Chiew Tan. 2007. Provenance in databases. In *SIGMOD*. ACM.
[34] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. 2022. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1897–1914.
[35] Tianqi Chen, Tong He, Michael Benesty, et al. 2015. Xgboost: extreme gradient boosting. *R package version 0.4-2* 1, 4 (2015), 1–4.
[36] Mark Craven and Jude Shavlik. 1995. Extracting tree-structured representations of trained networks. *Advances in neural information processing systems* 8 (1995).
[37] Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. 2020. Multi-objective counterfactual explanations. In *International Conference on Parallel Problem Solving from Nature*. Springer, 448–469.
[38] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. 2018. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. *NeurIPS* 31 (2018).
[39] Matthew F Dixon, Igor Halperin, and Paul Bilokon. 2020. *Machine learning in finance*. Vol. 1170. Springer.
[40] Jonathan Dodge, Q Vera Liao, Yunfeng Zhang, Rachel KE Bellamy, and Casey Dugan. 2019. Explaining models: an empirical study of how explanations impact fairness judgment. In *Proceedings of the 24th international conference on intelligent user interfaces*.
[41] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*. Springer, 1–19.
[42] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.
[43] Kareem El Gebaly, Parag Agrawal, Lukasz Golab, Flip Korn, and Divesh Srivastava. 2014. Interpretable and informative explanations of outcomes. *Proceedings of the VLDB Endowment* 8, 1 (2014), 61–72.
[44] Yuval Emek and Adi Rosén. 2016. Semi-streaming set cover. *ACM Transactions on Algorithms (TALG)* 13, 1 (2016), 1–22.
[45] Anna Fariha, Suman Nath, and Alexandra Meliou. 2020. Causality-guided adaptive interventional debugging. In *SIGMOD*. 431–446.
[46] Benjamin CM Fung, Ke Wang, and S Yu Philip. 2007. Anonymizing classification data for privacy preservation. *TKDE* 19, 5 (2007), 711–725.
[47] Sainyam Galhotra, Anna Fariha, Raoni Lourenço, Juliana Freire, Alexandra Meliou, and Divesh Srivastava. 2022. Dataprism: Exposing disconnect between data and systems. In *Proceedings of the 2022 International Conference on Management of Data*. 217–231.
[48] Sainyam Galhotra, Amir Gilad, Sudeepa Roy, and Babak Salimi. 2022. HypeR: Hypothetical Reasoning With What-If and How-To Queries Using a Probabilistic Causal Approach. In *SIGMOD*.
[49] Sainyam Galhotra, Romila Pradhan, and Babak Salimi. 2021. Explaining Black-Box Algorithms Using Probabilistic Contrastive Counterfactuals. In *SIGMOD*. ACM.
[50] M. R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.
[51] Boris Glavic, Alexandra Meliou, and Sudeepa Roy. 2021. Trends in explanations: Understanding and debugging data-driven systems. *Foundations and Trends® in Databases* (2021).
[52] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Counterfactual visual explanations. In *ICML*. 2376–2384.
[53] Todd J. Green, Gregory Karvounarakis, and Val Tannen. 2007. Provenance semirings. In *PODS*. ACM, 31–40.
[54] Riccardo Guidotti. 2022. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery* (2022), 1–55.
[55] Sariel Har-Peled, Piotr Indyk, Sepideh Mahabadi, and Ali Vakilian. 2016. Towards tight bounds for the streaming set cover problem. In *PODS*. 371–383.
[56] Johannes Haug, Alexander Braun, Stefan Zürn, and Gjergji Kasneci. 2022. Change Detection for Local Explainability in Evolving Data Streams. In *CIKM*. 706–716.
[57] Haoran He, Zhao Wang, Hemant Jain, Cuiqing Jiang, and Shanlin Yang. 2023. A privacy-preserving decentralized credit scoring method based on multi-party information. *Decision Support Systems* 166 (2023), 113910.

[58] Hans G Herzberger. 1979. Counterfactuals and consistency. *The Journal of Philosophy* 76, 2 (1979), 83–88.
[59] Alexey Ignatiev, Yacine Izza, Peter J. Stuckey, and João Marques-Silva. 2022. Using MaxSAT for Efficient Explanations of Tree Ensembles. In *AAAI*. 3776–3785.
[60] Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. 2020. High accuracy and high fidelity extraction of neural networks. In *29th USENIX security symposium (USENIX Security 20)*. 1345–1362.
[61] Tyler B Johnson and Carlos Guestrin. 2018. Training deep models faster with robust, approximate importance sampling. *NeurIPS* 31 (2018).
[62] Shalmali Joshi, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. 2019. Towards Realistic Individual Recourse and Actionable Explanations in Black-Box Decision Making Systems. *CoRR* (2019).
[63] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature* 596, 7873 (2021), 583–589.
[64] Amir-Hossein Karimi, Bernhard Schölkopf, and Isabel Valera. 2021. Algorithmic recourse: from counterfactual explanations to interventions. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*. 353–362.
[65] Amir-Hossein Karimi, Julius Von Kügelgen, Bernhard Schölkopf, and Isabel Valera. 2020. Algorithmic recourse under imperfect causal knowledge: a probabilistic approach. *Advances in neural information processing systems* 33 (2020), 265–277.
[66] Niki Kilbertus, Mateo Rojas Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. 2017. Avoiding discrimination through causal reasoning. *Advances in neural information processing systems* 30 (2017).
[67] Ron Kohavi. 1996. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. In *SIGKDD*. 202–207.
[68] Matt J. Kusner, Joshua R. Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual Fairness. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 4066–4076.
[69] Markus Langer, Daniel Oster, Timo Speith, Holger Hermanns, Lena Kästner, Eva Schmidt, Andreas Sesing, and Kevin Baum. 2021. What do we want from Explainable Artificial Intelligence (XAI)?–A stakeholder perspective on XAI and a conceptual model guiding interdisciplinary XAI research. *Artificial Intelligence* 296 (2021), 103473.
[70] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. 2018. Comparison-based inverse classification for interpretability in machine learning. In *IPMU*. 100–111.
[71] Thai Le, Suhang Wang, and Dongwon Lee. 2020. Grace: generating concise and informative contrastive sample to explain neural network model's prediction. In *SIGKDD*. 238–248.
[72] Seokki Lee, Bertram Ludäscher, and Boris Glavic. 2020. Approximate Summaries for Why and Why-not Provenance. *Proc. VLDB Endow.* 13, 6 (2020), 912–924.
[73] Raoni Lourenço, Juliana Freire, and Dennis Shasha. 2020. Bugdoc: Algorithms to debug computational processes. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 463–478.
[74] Scott M. Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *NeurIPS*. 4765–4774.
[75] Anh Mai, Matteo Brucateo, Azza Abouzied, Peter J Haas, and Alexandra Meliou. 2024. Scaling Package Queries to a Billion Tuples via Hierarchical Partitioning and Customized Optimization. *Proc. VLDB Endow.* (2024).
[76] Andrew McGregor and Hoa T Vu. 2019. Better streaming algorithms for the maximum coverage problem. *Theory of Computing Systems* 63 (2019), 1595–1619.
[77] Alexandra Meliou, Wolfgang Gatterbauer, Katherine F. Moore, and Dan Suciu. 2010. The Complexity of Causality and Responsibility for Query Answers and non-Answers. *Proc. VLDB Endow.* (2010).
[78] Alexandra Meliou, Wolfgang Gatterbauer, Suman Nath, and Dan Suciu. 2011. Tracing data errors with view-conditioned causality. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. 505–516.
[79] Alexandra Meliou, Sudeepa Roy, and Dan Suciu. 2014. Causality and explanations in databases. *Proceedings of the VLDB Endowment* 7, 13 (2014), 1715–1716.
[80] Michelle Miao. 2022. Debating the Right to Explanation: An Autonomy-Based Analytical Framework. *SAcLJ* 34 (2022), 864.
[81] Zhengjie Miao, Qitian Zeng, Boris Glavic, and Sudeepa Roy. 2019. Going beyond provenance: Explaining query answers with pattern-based counterbalances. In *Proceedings of the 2019 International Conference on Management of Data*. 485–502.
[82] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence* 267 (2019), 1–38.
[83] Smitha Milli, Ludwig Schmidt, Anca D Dragan, and Moritz Hardt. 2019. Model reconstruction from model explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. 1–9.

[84] Christoph Molnar. 2020. *Interpretable machine learning*. Lulu. com.
[85] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*. 607–617.
[86] Zafeiria Moumoulidou, Andrew McGregor, and Alexandra Meliou. 2021. Diverse Data Selection under Fairness Constraints. In *ICDT*, Vol. 186. 13:1–13:25.
[87] Harsh Parikh, Carlos Varjao, Louise Xu, and Eric Tchetgen Tchetgen. 2022. Validating causal inference methods. In *ICML*. 17346–17358.
[88] Martin Pawelczyk, Sascha Bielawski, Johannes van den Heuvel, Tobias Richter, and Gjergji Kasneci. 2021. CARLA: A Python Library to Benchmark Algorithmic Recourse and Counterfactual Explanation Algorithms. In *NeurIPS*.
[89] Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. 2020. Learning model-agnostic counterfactual explanations for tabular data. In *Proceedings of the web conference 2020*. 3126–3132.
[90] Martin Pawelczyk, Himabindu Lakkaraju, and Seth Neel. 2023. On the privacy risks of algorithmic recourse. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 9680–9696.
[91] Judea Pearl et al. 2000. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress* 19, 2 (2000), 3.
[92] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. 2016. *Causal inference in statistics: A primer*. John Wiley & Sons.
[93] Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. 2020. FACE: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 344–350.
[94] Romila Pradhan, Aditya Lahiri, Sainyam Galhotra, and Babak Salimi. 2022. Explainable AI: Foundations, Applications, Opportunities for Data Management Research. In *Proceedings of the 2022 International Conference on Management of Data*. 2452–2457.
[95] Romila Pradhan, Jiongli Zhu, Boris Glavic, and Babak Salimi. 2022. Interpretable data-based explanations for fairness debugging. In *SIGMOD*. 247–261.
[96] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *KDD*. 1135–1144.
[97] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-Precision Model-Agnostic Explanations. In *AAAI*. 1527–1535.
[98] Sudeepa Roy, Laurel J. Orr, and Dan Suciu. 2015. Explaining Query Answers with Explanation-Ready Databases. *Proc. VLDB Endow.* 9, 4 (2015), 348–359.
[99] Sudeepa Roy and Babak Salimi. 2022. Causal Inference in Data Analysis with Applications to Fairness and Explanations. In *Reasoning Web*.
[100] Sudeepa Roy and Dan Suciu. 2014. A formal approach to finding explanations for database queries. In *SIGMOD*. 1579–1590.
[101] Sandhya Saisubramanian, Sainyam Galhotra, and Shlomo Zilberstein. 2020. Balancing the tradeoff between clustering value and interpretability. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 351–357.
[102] Babak Salimi, Corey Cole, Peter Li, Johannes Gehrke, and Dan Suciu. 2018. HypDB: a demonstration of detecting, explaining and resolving bias in OLAP queries. *Proceedings of the VLDB Endowment* 11, 12 (2018), 2062–2065.
[103] Babak Salimi, Bill Howe, and Dan Suciu. 2020. Database repair meets algorithmic fairness. *ACM SIGMOD Record* 49, 1 (2020), 34–41.
[104] Andrew Selbst and Julia Powles. 2018. “Meaningful information” and the right to explanation. In *conference on fairness, accountability and transparency*. 48–48.
[105] Reza Shokri, Martin Strobel, and Yair Zick. 2021. On the privacy risks of model explanations. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. 231–241.
[106] Nina Spreitzer, Hinda Haned, and Ilse van der Linden. 2022. Evaluating the Practicality of Counterfactual Explanations. In *NeurIPS*.
[107] Bram Steenwinckel, Dieter De Paepe, Sander Vanden Hautte, Pieter Heyvaert, Mohamed Bentefrit, Pieter Moens, Anastasia Dimou, Bruno Van Den Bossche, Filip De Turck, Sofie Van Hoecke, et al. 2021. FLAGS: A methodology for adaptive anomaly detection and root cause analysis on sensor data streams by fusing expert knowledge with machine learning. *Future Generation Computer Systems* 116 (2021), 30–48.
[108] Balder ten Cate, Cristina Civili, Evgeny Sherkhonov, and Wang-Chiew Tan. 2015. High-level why-not explanations using ontologies. In *Proceedings of the 34th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 31–43.
[109] Tommaso Teofili, Donatella Firmani, Nick Koudas, Vincenzo Martello, Paolo Merialdo, and Divesh Srivastava. 2022. Effective explanations for entity resolution models. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2709–2721.
[110] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing machine learning

models via prediction {APIs}. In *25th USENIX security symposium (USENIX Security 16)*. 601–618.
[111] Guy Van den Broeck, Anton Lykov, Maximilian Schleich, and Dan Suciu. 2022. On the tractability of SHAP explanations. *Journal of Artificial Intelligence Research* 74 (2022), 851–886.
[112] Vijay V Vazirani. 2001. *Approximation algorithms*. Vol. 1. Springer.
[113] Sahil Verma, Varich Boonsanong, Minh Hoang, Keegan E Hines, John P Dickerson, and Chirag Shah. 2020. Counterfactual explanations and algorithmic recourses for machine learning: A review. *arXiv preprint arXiv:2010.10596* (2020).
[114] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.* 31 (2017), 841.
[115] Xiaolan Wang and Alexandra Meliou. 2019. Explain3D: Explaining Disagreements in Disjoint Datasets. *Proc. VLDB Endow.* (2019).
[116] Yongjie Wang, Qinxu Ding, Ke Wang, Yue Liu, Xingyu Wu, Jinglong Wang, Yong Liu, and Chunyan Miao. 2021. The skyline of counterfactual explanations for machine learning decision models. In *CIKM*. 2030–2039.
[117] Yongjie Wang, Hangwei Qian, and Chunyan Miao. 2022. Dualcf: Efficient model extraction attack from counterfactual explanations. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*. 1318–1329.
[118] Yuhao Wen, Xiaodan Zhu, Sudeepa Roy, and Jun Yang. 2018. Interactive summarization and exploration of top aggregate query answers. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, Vol. 11. 2196.
[119] James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda Viégas, and Jimbo Wilson. 2019. The what-if tool: Interactive probing of machine learning models. *IEEE transactions on visualization and computer graphics* 26, 1 (2019), 56–65.
[120] Eugene Wu and Samuel Madden. 2013. Scorpion: Explaining Away Outliers in Aggregate Queries. *Proc. VLDB Endow.* 6, 8 (2013), 553–564.
[121] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*. IEEE, 268–282.
[122] Brit Youngmann, Michael J. Cafarella, Babak Salimi, and Anna Zeng. 2023. Causal Data Integration. *Proc. VLDB Endow.* 16, 10 (2023), 2659–2665.
[123] Huiwen Yu and Dayu Yuan. 2013. Set coverage problems in a one-pass data stream. In *ICDM*. SIAM, 758–766.

Received October 2023; revised January 2024; accepted February 2024