Navigating the Structured What-If Spaces: Counterfactual Generation via Structured Diffusion

***Abstract*—Generating counterfactual explanations is one of the most effective approaches for uncovering the inner workings of black-box neural network models and building user trust. While remarkable strides have been made in generative modeling using diffusion models in domains like vision, their utility in generating counterfactual explanations in structured modalities remains unexplored. In this paper, we introduce *Structured Counterfactual Diffuser* or SCD, the first plug-and-play framework leveraging diffusion for generating counterfactual explanations in structured data. SCD learns the underlying data distribution via a diffusion model which is then guided at test time to generate counterfactuals for any arbitrary black-box model, input, and desired prediction. Our experiments show that our counterfactuals not only exhibit high plausibility compared to the existing state-of-the-art but also show significantly better proximity and diversity.**

I. Introduction

As AI models become more capable and widespread, the issue of trust becomes critical [1]. While traditional software is transparent—allowing tracing its control flow and easily resolving trust concerns—modern AI is built upon neural networks that are not transparent. Their underlying control flow is not understood, making it difficult to trust in high-risk settings such as loan or hiring decisions. Although the remarkable power and flexibility of neural networks have allowed building systems that achieve capabilities not possible with traditional software alone [2], [3], this lack of transparency and trust becomes a significant hurdle in realizing the full potential of neural networks [4]–[6].

To address concerns about trust, one needs to answer *why* a model behaves in a certain way. One of the most promising directions to answer this is via *what-if* scenarios or counterfactuals [6]. For instance, consider a model which declines a loan for `[Female, Earns $100K]`. To answer *why*, it is of interest to discover counterfactuals for which the same model approves loans. For instance, if the model approves the loan for a counterfactual instance `[Male, Earns $100K]`, this suggests that the model may be making decisions based on potentially problematic criteria, prompting model developers to investigate and fix the problem. Additionally, counterfactuals can also provide actionable insights to the end-users on how to achieve a different outcome [7]. In our previous example, if the model approves the loan for a counterfactual instance `[Female, Earns $110K]`, it explains what the applicant might need to do to obtain approval.

While [6] originally introduced the idea of counterfactual explanations, the idea has gained significant attention in recent years [7]–[11]. Ideally, counterfactuals should possess the following characteristics: *1)* they should maintain *proximity* to the original input, *2)* they should attain the desired counterfactual label to ensure its *validity*, *3)* they should be *diverse* and capture a wide range of distinct scenarios and *4)* they should be *plausible*. While proximity, validity, and diversity criteria have been studied extensively, there has been little focus on the plausibility of the generated counterfactuals, i.e., ensuring that the generated counterfactuals are realistic and conform to the underlying data distribution. Previous works have approached plausibility in a minimal sense, e.g., enforcing values to lie in legal ranges or applying user-designed constraints [7], [8].

Recently, in the visual domain, diffusion models [12] have been successfully used to acquire the underlying data distribution for generating plausible counterfactual explanations [13]–[16]. However, in the domain of tabular or structured data, counterfactual explanation methods have largely ignored these recent advances in diffusion modeling raising another important question: *"Can diffusion models, which are known for their remarkable generation capabilities in vision, help generate high-quality plausible counterfactuals in the structured domain?"*

To answer this question, in this work, we propose a novel counterfactual explainer called *Structured Counterfactual Diffuser* or *SCD*. SCD is the first plug-and-play framework leveraging diffusion modeling for generating counterfactual explanations for structured data. SCD works by learning the underlying data distribution via a diffusion model [12], [17]. At test time, the diffusion model is used to perform guided iterative denoising to generate counterfactuals for any given input and black-box model in a plug-and-play manner. In experiments, we show that our counterfactual explainer not only exhibits high plausibility compared to the state-of-the-art approaches but also shows significantly better proximity and diversity scores of the generated counterfactuals. In our analysis, we also find that our method, due to its unique stochastic denoising process, does not require explicit incentives to generate diverse counterfactuals, unlike the previous counterfactual explainers for structured data.

II. Preliminaries

Structured Data. A table or structured data consists of rows or instances. Each instance is a tuple with a value for each column or attribute. The entire space of such instances can be described as $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_C$. Here, C denotes the number of columns or attributes in the table, and each $\mathcal{X}_c$ denotes the space of possible values for column c. For example, a possible instance from a 4-column table is `[female, 40, doctoral, married]`. Here, $\mathcal{X}_1$ can

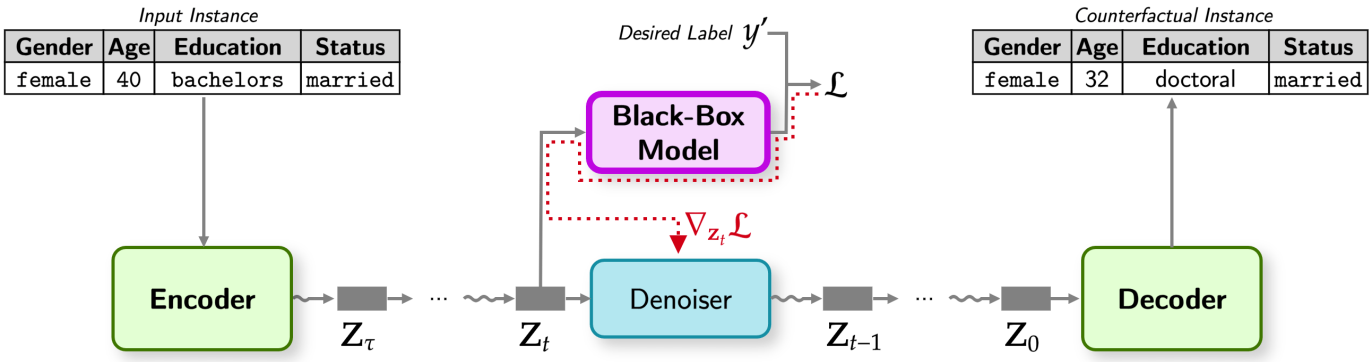

Fig. 1. **Overview of Our Counterfactual Generation Process.** The process starts by encoding the given human-readable instance or row into an embedding by performing a look-up on a dictionary of learned embeddings. Next, we iteratively apply denoising steps while incorporating the gradient information from the given black-box model to minimize the disparity between the model's prediction and the desired label. At the end of the denoising process, we obtain an embedding which is then decoded via a reverse look-up on the dictionary to obtain the counterfactual instance.

represent gender categories, $\mathcal{X}_2$ can represent the possible age values, and so forth. We will use $\mathbf{x}$ to denote an instance and $\mathbf{x}^c$ to denote c-th column or attribute within the instance.

Black-Box Model. A black-box model is a model $f : \mathcal{X} \to \mathcal{Y}$ that maps an input instance $\mathbf{x} \in \mathcal{X}$ to a label $y \in \mathcal{Y}$. However, the model is *black-box* in the sense that its inner workings are not understood and explainability tools are required to shed light on it. In the rest of the paper, we will use the term *model* and *black-box model* interchangeably.

A. Structured Counterfactual Explanations

As highlighted by [6], counterfactuals help identify alternative scenarios where a slight change in the original input $\mathbf{x}$ to a counterfactual input $\mathbf{x}'$ would have changed the outcome from y to y' by a black-box model f. By analyzing the change in prediction on counterfactual inputs, one can uncover if the model is making decisions based on potentially problematic or undesired criteria.

Counterfactual Explainer. Formally, a counterfactual explainer can be described as a system or framework that, given an input $\mathbf{x}$, a model f, and a counterfactual label y' (where y' is different from the original label y), produces a set of B counterfactuals $\mathbf{X}'$.

$$\mathbf{X}' = \{\mathbf{x}'_1, \ldots, \mathbf{x}'_B\} = \text{CounterfactualExplainer}(f, \mathbf{x}, y').$$

Here, each counterfactual $\mathbf{x}'_b \in \mathbf{X}'$ should achieve the counterfactual label y' on the given black-box model f with minimal change to the original input $\mathbf{x}$.

Desired Characteristics of Counterfactuals. There are 4 fundamental characteristics that counterfactuals in $\mathbf{X}'$ should possess:

1) *Validity:* Should achieve the label y'.
2) *Proximity:* Should be close to the original input $\mathbf{x}$.
3) *Diversity:* Should be diverse and not collapse to a single instance.
4) *Plausibility:* Should be plausible, i.e., should capture realistic instances from the input space.

While [6] originally introduced the validity and proximity desiderata, [7] introduced the desiderata of diversity. Plausibility, on the other hand, has not been given much attention in the community. Some existing works primarily focus on only keeping generated values within legal ranges, disregarding the complex relationships that values of various columns have [8] or require costly user-defined plausibility constraints [7]. In this work, we take a significant step forward in alleviating this concern.

III. SCD: Structured Counterfactual Diffuser

In this section, we present our proposed model *Structured Counterfactual Diffuser* or *SCD*. SCD learns a diffusion model through training on a structured dataset or table $\mathcal{D}$. Via training on $\mathcal{D}$, SCD learns about the underlying data distribution which enables it to generate plausible counterfactuals. Once the diffusion model is trained, SCD can be used in a plug-and-play manner to obtain counterfactual explanations for any given black-box model. We now describe SCD in detail.

Row Embedding. To train the diffusion model, we first map the raw human-readable instances or rows $\mathbf{x}$ of the table $\mathcal{D}$ into embeddings. The diffusion model shall be trained to model the distribution in this embedding space. We maintain a learned dictionary of embeddings $\text{Embedding}_c : \mathcal{X}_c \to \mathbb{R}^d$ for each column c. To encode a row, we lookup the embedding for each of the C columns and concatenate these embeddings to obtain a row embedding $\mathbf{z}$ as follows:

$$\mathbf{z} = [\text{Embedding}_1(\mathbf{x}^1), \ldots, \text{Embedding}_C(\mathbf{x}^C)] \in \mathbb{R}^{C \times d}$$

where d is the size of the embedding per column.

A. Diffusion Modeling

Via diffusion modeling, we seek to learn a distribution $p_\theta(\mathbf{z})$ over the row embeddings. In diffusion modeling, the

distribution $p_\theta(\mathbf{z})$ consists of T denoising steps:

$$p_\theta(\mathbf{z}_0) = \int p(\mathbf{z}_T) \prod_{t=T,\ldots,1} p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t, t) d\mathbf{z}_{1:T}$$

Here, $p(\mathbf{z}_T)$ represents standard Gaussian, the sequence $\mathbf{z}_T, \ldots, \mathbf{z}_1$ consists of iteratively cleaner samples, finally producing the desired sample $\mathbf{z}_0$; and $p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t, t)$ is a one-step denoising distribution. The $p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t, t)$ is parametrized in the following manner:

$$\mathcal{N}\left(\gamma_{1,t}\hat{\mathbf{z}}_0 + \gamma_{2,t}\mathbf{z}_t, \beta_t \mathbf{I}\right)$$

where $\hat{\mathbf{z}}_0 = g_\theta(\mathbf{z}_t, t)$, and the coefficients $\gamma_{1,t}$ and $\gamma_{2,t}$ are given by:

$$\gamma_{1,t} = \frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t}, \qquad \gamma_{2,t} = \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\bar{\alpha}_t}$$

Employing standard notations, we utilize a variance schedule $\beta_1, \ldots, \beta_T$, where $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$. We use a cosine schedule in our implementation.

Unconditional Sampling. To obtain unconditional samples from the learned diffusion model, we start with random Gaussian noise, $\mathbf{z}_T \sim p(\mathbf{z}_T)$. Next, using the trained one-step denoising distribution $p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t, t)$, we iteratively denoise the samples until $\mathbf{z}_0$, the desired sample, is obtained.

Learning: The training procedure involves first introducing noise to the input $\mathbf{z}_0$, creating its noisy version $\mathbf{z}_t$.

$$\mathbf{z}_t = \sqrt{\bar{\alpha}_t}\mathbf{z}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, \qquad \text{where } \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

Subsequently, a neural network predictor is trained that takes $\mathbf{z}_t$ as input and aims to predict the original input $\mathbf{z}_0$ by generating a prediction $\hat{\mathbf{z}}_0 = g_\theta(\mathbf{z}_t, t)$. The learning objective is $\mathcal{L}_{\text{diffusion}}(\theta) = \mathcal{E}(\hat{\mathbf{z}}_0, \mathbf{z}_0)$ where $\mathcal{E}$ is an error function.

B. Generating Counterfactuals via Guided Diffusion

Given the trained denoising distribution $p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t, t)$, we are now ready to generate counterfactuals for a black-box model f given an input instance $\mathbf{x}$ and a desired label y'. The process works by performing guided diffusion starting from the embedding of the given input instance. For this, we first encode $\mathbf{x}$ to its row embedding $\mathbf{z} \in \mathbb{R}^{C\times d}$. Since we seek to sample B counterfactuals, we copy the row embedding B times and stack the copies together to construct an embedding $\mathbf{Z} \in \mathbb{R}^{B\times C\times d}$. Next, we add Gaussian noise to $\mathbf{Z}$ to facilitate diversity among the B generated samples.

$$\mathbf{Z}'_\tau \leftarrow \sqrt{\bar{\alpha}_\tau}\mathbf{Z} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, \qquad \text{where } \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

Next, we perform τ guided diffusion steps. We iteratively and alternatingly apply the following two steps:

1) *Denoising Step:* This step involves sampling $\mathbf{Z}'_{t-1} \sim p_\theta(\mathbf{Z}'_{t-1}|\mathbf{Z}'_t, t)$.
2) *Guiding Step:* This step involves performing a gradient step on $\mathbf{Z}'_{t-1}$ with respect to a guiding loss $\mathcal{L}$ as:

$$\mathbf{Z}'_{t-1} \leftarrow \mathbf{Z}'_{t-1} - \eta\nabla_{\mathbf{Z}'_{t-1}}\mathcal{L}$$

where η is the step size for the update. One of the things that $\mathcal{L}$ measures is how well the black-box model f produces the counterfactual label y' on the samples $\mathbf{Z}'_{t-1}$ of the current step. We describe the exact formulation of $\mathcal{L}$ in detail in a later section.

From this iterative process, we obtain a series of progressively cleaned embeddings $\mathbf{Z}'_\tau, \ldots, \mathbf{Z}'_0$. Next, we take the generated $\mathbf{Z}'_0$, perform reverse look-up using the learned embeddings and obtain the human-readable counterfactual instances $\mathbf{X}' = \{\mathbf{x}'_1, \ldots, \mathbf{x}'_B\}$. In Fig. 1, we illustrate this process.

Guiding Loss: We now describe the terms in our guiding loss $\mathcal{L}$. Following [7], we include 3 terms in our loss capturing validity, proximity, and diversity of the samples. Formally this loss can be described as:

$$\begin{aligned}\mathcal{L}(\mathbf{Z}', \mathbf{x}, f, y') = &\lambda_{\text{validity}}\mathcal{L}_{\text{validity}}(\mathbf{Z}', f, y') \\ &+ \lambda_{\text{proximity}}\mathcal{L}_{\text{proximity}}(\mathbf{Z}, \mathbf{Z}') \\ &+ \lambda_{\text{diversity}}\mathcal{L}_{\text{diversity}}(\mathbf{Z}')\end{aligned}$$

1) *Validity Loss.* We use the cross-entropy loss of the black-box model f with respect to the desired prediction y' as our validity loss.

$$\mathcal{L}_{\text{validity}}(\mathbf{Z}', f, y') = \text{CrossEntropy}(f(\mathbf{Z}'), \texttt{target} = y').$$

2) *Proximity Loss.* We use a simple L2 loss between $\mathbf{Z}$ the embedding of the original input and $\mathbf{Z}'$ the generated embedding at the current step of the guided diffusion.

$$\mathcal{L}_{\text{proximity}}(\mathbf{Z}, \mathbf{Z}') = ||\mathbf{Z} - \mathbf{Z}'||^2.$$

3) *Diversity Loss.* We use the negative of L2 loss between all pairs of counterfactual instances

$$\mathcal{L}_{\text{diversity}}(\mathbf{Z}') = \frac{-2}{B(B-1)} \sum_{i=1}^{B-1} \sum_{j=i+1}^{B} ||\mathbf{z}'_i - \mathbf{z}'_j||^2.$$

C. Discussion

Our method has multiple benefits. First, our method operates in a plug-and-play manner once the diffusion model is trained. That is, no training is required during sampling of counterfactual explanations:

$$\mathbf{X}' = \text{CounterfactualExplainer}_\theta(f, \mathbf{x}, y').$$

Second, our experiments shall show that our method can produce diverse samples without requiring an explicit diversity term in the guiding loss, distinguishing it from previous methods like DiCE [7] which require an explicit diversity term in the loss. Third, our experiments shall show that our method can inherently preserve the contents of the original input without requiring an explicit proximity term in the guiding loss—another attractive aspect of our method. Fourth, not requiring proximity and diversity terms removes the burden of tuning the coefficients $\lambda_{\text{proximity}}$ and $\lambda_{\text{diversity}}$ which can be quite brittle in the previous methods.

TABLE I

COMPARISON OF PLAUSIBILITY, PROXIMITY, DIVERSITY, AND VALIDITY SCORES OF WACHTER, SCD AND DICE ON VARIOUS DATASETS. FOR VALIDITY, PROXIMITY, AND DIVERSITY SCORES, HIGHER IS BETTER. FOR THE PLAUSIBILITY SCORE, LOWER IS BETTER SINCE IT CAPTURES THE NEGATIVE LOG-LIKELIHOOD OF THE GENERATED SAMPLES.

Dataset	**Plausibility** (↓)			**Proximity** (↑)		
	Wachter *et al.*	**DiCE**	**SCD**	**Wachter *et al.***	**DiCE**	**SCD**
Adult Income	108.7	121.0	**21.21**	0.685	0.5764	**0.6173**
UCI Bank	168.3	166.7	**42.37**	0.226	0.2141	**0.3000**
Housing Price	102.8	109.5	**42.91**	0.375	0.3055	**0.3417**
Dataset	**Diversity** (↑)			**Validity** (↑)		
	Wachter *et al.*	**DiCE**	**SCD**	**Wachter *et al.***	**DiCE**	**SCD**
Adult Income	0.002	0.3837	**0.4008**	0.9400	**0.9776**	0.7511
UCI Bank	0.041	0.4165	**0.5498**	0.9900	**0.9686**	0.8600
Housing Price	0.03	0.4289	**0.5986**	0.9999	**0.9908**	0.8526

TABLE II

COUNTERFACTUAL SAMPLES IN ADULT INCOME DATASET. GIVEN THE INPUT ROW WITH THE ORIGINAL LABEL "$\leq 50K$", WE ASK OUR METHOD SCD AND THE BASELINE DICE TO GENERATE COUNTERFACTUAL INSTANCES THAT FLIP THE LABEL TO "$> 50K$" WITH RESPECT TO A BLACK-BOX INCOME PREDICTOR. WE NOTE THAT SCD GENERATES PLAUSIBLE SAMPLES WHILE DICE STRUGGLES. SPECIFICALLY, WE NOTE THAT DICE CREATES COUNTERFACTUALS CONTAINING *Divorced* AND *Husband* WITHIN THE SAME ROW WHICH IS CONTRADICTORY AND IMPOSSIBLE (HIGHLIGHTED IN RED). IN COMPARISON, SCD CREATES PLAUSIBLE COUNTERFACTUALS WHERE THE MARITAL STATUS, RELATIONSHIP AND GENDER COLUMNS CORRECTLY CONFORM WITH EACH OTHER (HIGHLIGHTED IN GREEN).

Method	**Age**	**Workclass**	**Education**	**Ed. No.**	**Marital Status**	**Occupation**	**Relationship**	**Race**	**Gender**	**Hr/W**	**Country**
Input	39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	40	US
Ours	31	Self-emp-inc	Bachelors	13	Married-civ-spouse	Adm-clerical	Wife	White	Female	40	US
	34	Self-emp-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	55	US
	39	Federal-gov	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	White	Male	40	US
DiCE	39	State-gov	Bachelors	16	Divorced	Transport-moving	Husband	White	Male	40	US
	39	State-gov	Bachelors	16	Divorced	Transport-moving	Husband	AME	Male	40	US
	39	Without-pay	Some-college	16	Divorced	Transport-moving	Husband	White	Male	40	US

IV. RELATED WORK

Explainable AI. Explainable AI (XAI) has received significant attention over the past few years [18]. Several methods seek saliency maps as a way of explanation [19], [20]. The notion of generating explanations has been well studied in image domain [21], [22], and [23]. Several methods focus on perturbing the input instance, however, these perturbations are not optimized to achieve a counterfactual prediction under the given black-box model [24]–[26]. Another line of work on generating adversarial instances of an input instance has been well studied [27]–[30]. However, unlike ours, these methods are not concerned with the plausibility of the samples. Other classes of methods include approximation-based ones which learn local or global decision boundaries to generate explanations [4], [5], [31]. However, these are not counterfactual explainers. Another way to achieve interpretability has been to introduce disentanglement within the neural network layers [32]. However, this approach does not seek to explain existing black-box models.

Counterfactual Explanations. For the tabular domain, various studies have pursued counterfactual explanations [6], [7], [9], [33], [34]. However, none of them directly and properly tackle the problem of generating plausible counterfactuals. [35] propose a technique to select counterfactual samples from the training set and show the applicability on synthetic datasets. However, their focus is not on generating completely new counterfactuals that don't occur within the training set. In the image domain, several works attempt to generate counterfactuals using diffusion models [13]–[16]. This is another line of works focusing on contrastive explanations [36], [37], however, these do not leverage diffusion modeling, like ours. However, while these are based within the image domain, the utility of diffusions models for counterfactual explanation in the tabular domain has remained unexplored. In the language domain, there has been a significant number of works for counterfactual generation [10], [11], [38]–[41]. However, these have primarily relied on auto-regressive LLMs and not diffusion models. Although [17] pursues diffusion-based language modeling, it does not pursue the task of counterfactual explanation and also does not deal with the tabular domain. Additionally, there has also been interest in the domain of search and retrieval for generating counterfactual explanations [42].

V. EXPERIMENTS

Datasets. In experiments, we evaluate the quality of generated counterfactuals on three datasets:

1) **Adult Income Dataset** [43]. This dataset contains educational, demographic, and occupancy information of

individuals. We use the following features: hours per week, education level, occupation, work class, race, age, marital status, and sex. These are selected following the pre-processing approach of [44].

2) **UCI Bank Dataset** [45]. This dataset contains the marketing campaigns of a banking institution.
3) **Housing Price Dataset** [46]. This dataset contains information regarding the demography (income, population, house occupancy) in the districts of California, the location of the districts (latitude, longitude), and general information regarding the house in the districts (number of rooms, number of bedrooms, age of the house).

Black-Box Model. For each dataset, we train a classifier to act as the black-box model that a counterfactual explainer would seek to explain. The architecture is a simple 2-layer MLP that takes the concatenated embeddings of columns of a row as input and tries to predict a class label. For each dataset, the classification task that the black-box model is trained to perform is as follows: *1) Adult Income Dataset:* Given a row as input, the black-box model predicts whether the income exceeds 50K per year or not. *2) UCI Bank Dataset:* Given a row describing attributes of a client, the black-box model predicts if the client will subscribe to a term deposit or not. *3) Housing Price Dataset:* Given a row as input, the black-box model predicts whether the house price is greater than $200K or not.

Baselines. We compare our model with two baseline counterfactual explainers for structured datasets: *1)* DiCE, the current state-of-the-art, and *2)* Wachter *et al.* [6]. These work by encoding the given row to a vector of per-column one-hot embeddings. To generate counterfactuals, they apply Stochastic Gradient Descent (SGD) to minimize a loss having terms focusing on validity, proximity, and diversity (in DiCE). These baselines provide the most comprehensive evaluation of the proposed model since, like ours, it also leverages gradient-based dynamics to generate counterfactuals. Although it might appear that the number of compared baselines is small, we highlight that this line of research, although important, is still in its infancy and the two baselines we compare with are the most relevant with respect to our contribution.

A. Metrics

We consider the following metrics for evaluating the generated counterfactuals.

1) **Validity Score.** We compute the validity score of the generated counterfactuals in $\mathbf{X}'$ by checking if they result in the desired label with respect to the black-box model.

$$\text{Validity Score} = \frac{1}{B}\sum_{b=1}^{B}\mathbb{I}(y' == f(\mathbf{x}'_b))$$

where $\mathbb{I}(\cdot)$ is an indicator function that takes a value 1 if its input is true else 0.

2) **Proximity Score.** We compute proximity score as the mean of distances between the generated counterfactuals in $\mathbf{X}'$ and the original input $\mathbf{x}$. This is computed as:

$$\text{Proximity Score} = \frac{1}{B}\sum_{b=1}^{B}\text{distance}(\mathbf{x}'_b, \mathbf{x})$$

where distance$(\cdot,\cdot)$ is a distance function between two instances that measures the fraction of N columns or values that do not match.

3) **Diversity Score.** We compute the diversity score of the generated counterfactuals in $\mathbf{X}'$ as the mean of the distances between each pair of samples.

$$\text{Diversity Score} = \frac{2}{B(B-1)}\sum_{i=1}^{B-1}\sum_{j=i+1}^{B}\text{distance}(\mathbf{x}'_i, \mathbf{x}'_j)$$

where distance$(\cdot,\cdot)$ is a distance function between two instances that measures the fraction of N columns or values that do not match.

4) **Plausibility.** The goal is to evaluate how likely is the generated counterfactual under the true data distribution. We learn a model of the desired distribution by learning an auto-regressive model p_ϕ over the tokens or values in the instances. This auto-regressive model is described in further detail in the supplementary material. To compute the plausibility score, we compute the negative log-likelihood of each generated counterfactual $\mathbf{x}'_b \in \mathbf{X}'$ using p_ϕ.

$$\begin{aligned}\text{Plausibility} &= -\frac{1}{B}\sum_{b=1}^{B}\log p_\phi(\mathbf{x}'_b) \\ &= -\frac{1}{B}\sum_{b=1}^{B}\sum_{n=1}^{N}\log p_\phi(\mathbf{x}'_{b,n}|\mathbf{x}'_{b,1},\ldots,\mathbf{x}'_{b,n-1})\end{aligned}$$

where a lower negative log-likelihood is desired for a more plausible counterfactual.

B. Benefits of SCD in Counterfactual Generation

In Table I, we compare our model SCD and our baseline, DiCE. It is remarkable that our model produces counterfactuals that are significantly more plausible than those generated by DiCE. In fact, the negative log-likelihood of our samples are 21.21, 42.37, and 42.91 while DiCE yields significantly worse results attaining 121.0, 166.7, and 109.5 on the 3 datasets, respectively. Our higher plausibility is also evidenced by our generated counterfactual samples in Table II. We can see that our model coherent values for the columns *Marital Status* and *Relationship* while the baseline DiCE produces contradictory values e.g., *Divorced* and *Husband* within the same row. This highlights the advantage of using a diffusion model that learns complex relationships to constrain the generated counterfactuals to be plausible. We show additional qualitative counterfactual samples generated by SCD in Table III.

Furthermore, our results show significant improvements in the diversity and proximity scores over the baseline, achieving

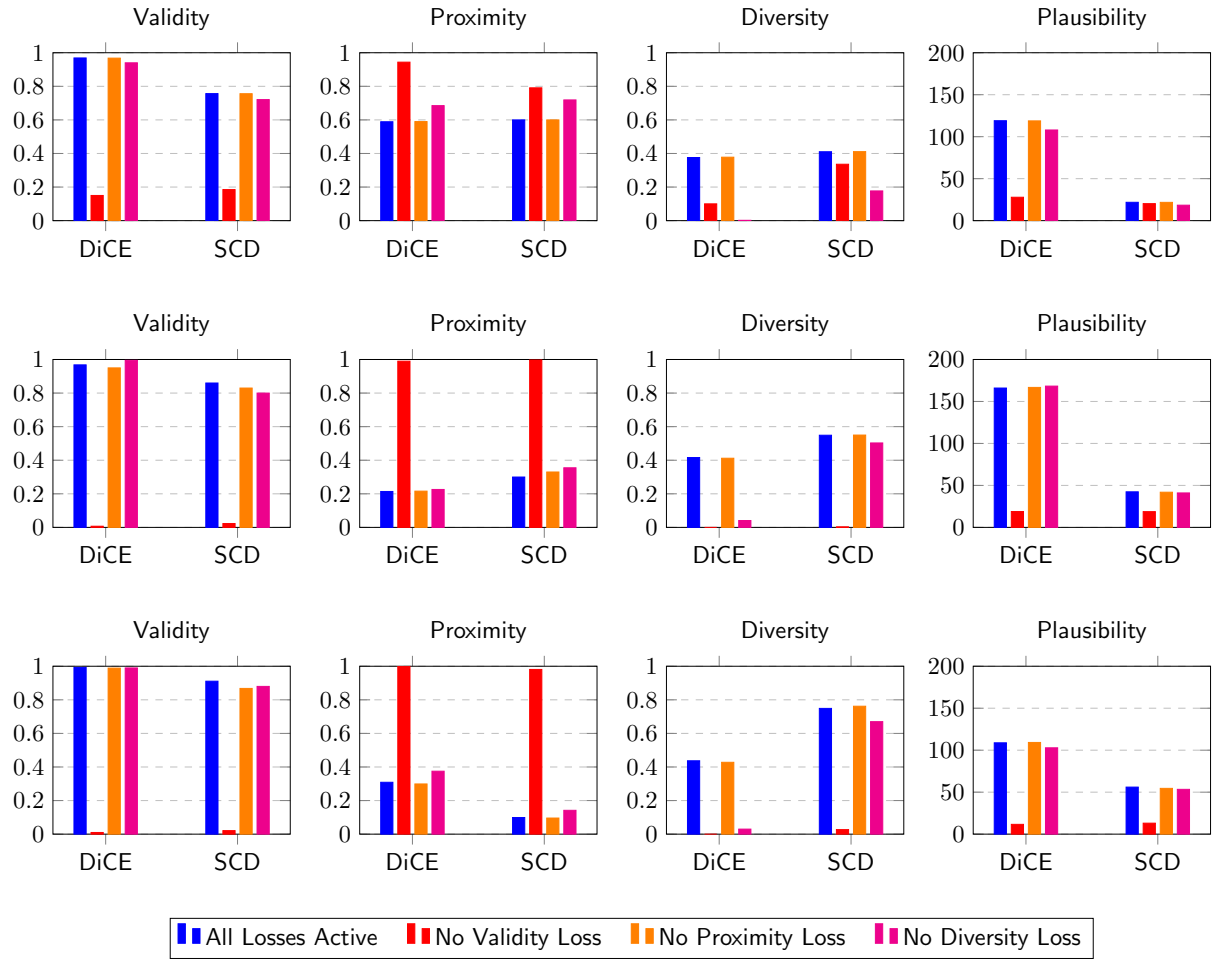

Fig. 2. **Ablation of Losses.** We perform a comparison of models on ablations of the guiding loss: *1)* all losses are active, *2)* no validity loss, *3)* no proximity loss, and *4)* no diversity loss. This comparison is done on Adult Income Dataset (*top*), UCI Bank Dataset (*middle*) and Housing Price Dataset (*bottom*). We note that when diversity loss is dropped, the performance of the baseline DiCE suffers while our model SCD maintains good diversity, proximity, and validity. In general, our model SCD maintains a very high plausibility in all scenarios relative to DiCE.

approximately 0.10-0.17 higher diversity and 0.04-0.10 higher proximity scores relative to DiCE. Our validity score, i.e., the fraction of generated counterfactuals that attain the desired label, is about 0.1 lower than the baseline. While this is a slight decline, it is not a significant concern since it is straightforward to remove the counterfactuals that do not attain the desired label via post-processing. Furthermore, some worsening of the validity score may be expected since SCD constrains the samples to be plausible while DiCE does not.

C. Analysis of Model Characteristics

Question 1. How does dropping various loss terms affect performance?

In our guiding loss, we used 3 terms: validity, proximity, and diversity. While our default version retains all three terms, we would like to assess what would happen if each of the three terms were individually dropped. In Fig. 2, we report these results.

1) *Dropping the Validity Term.* When we drop the validity term, we note that the validity score drops close to 0. That is, no generated samples are actually able to achieve the counterfactual label. This observation is shared for both our model as well as the baseline. This can be expected since the validity term in the loss is the only way to inform the generation process about the disparity between desired and the predicted label. Furthermore, all generated samples collapse to the original input, as suggested by a high proximity score and a very low diversity score.
2) *Dropping the Proximity Term.* When we drop the proximity term, we note that the scores are not significantly affected. We think this is because, in both SCD and the baseline, the process of generating the counterfactual starts with the original input, and the update steps are not able to deviate significantly from the original input.
3) *Dropping the Diversity Term.* When we drop the diversity term, we note, remarkably, that the diversity

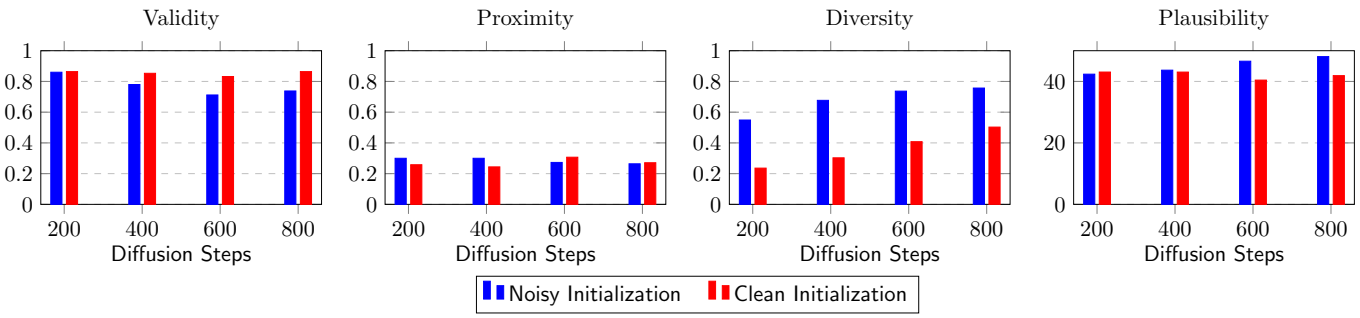

Fig. 3. **Changing Diffusion Steps.** We report our metrics with respect to 1) changing diffusion steps, and 2) starting the guided diffusion with (shown in blue) and without (shown in red) adding an initial noise input. We note that SCD remains robust to varying diffusion steps. Furthermore, we note a remarkable drop in diversity when the initial noise is not added at the start of the guided diffusion process.

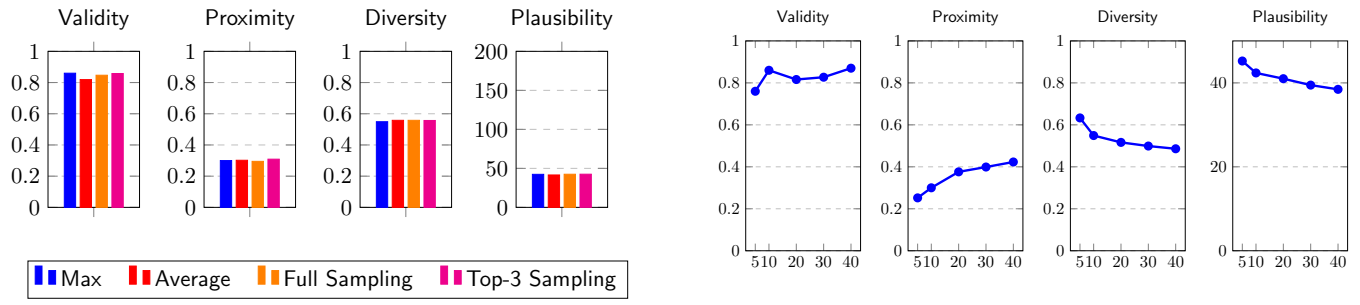

Fig. 4. **Left: Effect of sampling strategies.** We vary the sampling strategies in our guided diffusion process and show the effect on our metrics. We observe a slightly higher validity score for `Max`. For other metrics, the scores remain robust. **Right: Effect of Choice of B.** We report our metric with respect to B, the number of counterfactuals generated.

of samples of DiCE drops to 0. In comparison, our counterfactuals maintain high diversity even after removing the diversity loss term. This shows a unique characteristic of our model that by leveraging stochastic denoising, the samples of our model become naturally diverse. On the other hand, the existing models lack such stochasticity, requiring an explicit diversity loss term and careful tuning of its coefficient.

Question 2. How does varying the number of guided diffusion steps affect performance?

We perform this analysis and report results in Fig. 3. We note that our model is remarkably robust to the number of guided diffusion steps in terms of validity, proximity, and plausibility. In the diversity score, however, we see a slight upward trend with the increasing number of diffusion steps. We think this is because the diffusion steps are stochastic. Thus, accumulating randomness from a greater number of diffusion steps appears to promote higher sample diversity.

Question 3. How does adding noise at the start of guided diffusion affect performance?

In Fig. 3, we also compare our model with and without adding noise at the start of the guided diffusion. We note that adding the noise is clearly beneficial since not adding the noise worsens the diversity score. This observation is consistent across different numbers of diffusion steps during counterfactual generation.

Question 4. How does varying sampling strategy for guided diffusion affect performance?

We test various sampling strategies during guided diffusion and whether it affects the performance or not. We test 4 sampling strategies for the denoising diffusion step. The first strategy is to choose the highest probability embedding per column (denoted as `Max`). The second strategy is to use a probability-weighted average of embeddings (denoted as `Average`). The third strategy is to sample an embedding under the predicted distribution (denote as `Full Sampling`). Lastly, our fourth strategy is to take the top-3 highest probability embeddings and randomly sample among these. Across all 4 strategies, in Fig. 4, we find the performances to be similar, indicating that our model is robust to this choice.

Question 5. How does the number of generated counterfactuals affect performance?

We vary the number of generated counterfactuals in parallel (denoted as B) and report performance in Figure 4. Note that

TABLE III

COUNTERFACTUAL SAMPLES IN ADULT INCOME DATASET. GIVEN THE INPUT ROW WITH THE ORIGINAL LABEL "$\leq 50K$", WE ASK OUR METHOD SCD TO GENERATE COUNTERFACTUAL INSTANCES THAT FLIP THE LABEL TO "$> 50K$" WITH RESPECT TO A BLACK-BOX INCOME PREDICTOR. WE NOTE THAT SCD GENERATES PLAUSIBLE SAMPLES.

Method	Age	Workclass	Education	Ed. No.	Marital Status	Occupation	Relationship	Race	Gender	Hr/W	Country
Input	38	Private	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	40	United-States
Ours	38	Private	9th	5	Never-married	Handlers-cleaners	Other-relative	White	Male	40	United-States
	38	Private	HS-grad	9	Divorced	Other-service	Not-in-family	Black	Female	40	United-States
	44	Private	HS-grad	9	Divorced	Handlers-cleaners	Unmarried	White	Female	40	United-States
Input	49	Private	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female	16	Jamaica
Ours	31	Private	5th-6th	3	Married-spouse-absent	Other-service	Not-in-family	Other	Female	35	Jamaica
	61	Private	9th	5	Never-married	Machine-op-inspct	Other-relative	Black	Female	16	Trinadad&Tobago
	49	Private	9th	5	Separated	Other-service	Not-in-family	Black	Female	48	Jamaica
Input	23	Private	Bachelors	13	Never-married	Adm-clerical	Own-child	White	Female	30	United-States
Ours	23	Private	Bachelors	13	Never-married	Adm-clerical	Own-child	Black	Female	30	United-States
	25	Private	Bachelors	13	Never-married	Adm-clerical	Own-child	White	Female	30	United-States
	23	Private	Bachelors	13	Never-married	Adm-clerical	Own-child	White	Female	30	South
Input	39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	40	US
Ours	31	Self-emp-inc	Bachelors	13	Married-civ-spouse	Adm-clerical	Wife	White	Female	40	US
	34	Self-emp-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	55	US
	39	Federal-gov	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	White	Male	40	US
Input	49	Private	HS-grad	9	Married-spouse-absent	Craft-repair	Husband	White	Male	40	United-States
Ours	49	Private	HS-grad	9	Married-spouse-absent	Craft-repair	Own-child	White	Male	40	Canada
	27	Private	HS-grad	9	Married-civ-spouse	Other-service	Other-relative	Amer-Indian-Eskimo	Female	48	United-States
	49	Private	HS-grad	9	Separated	Craft-repair	Not-in-family	White	Male	55	United-States
Input	19	Private	HS-grad	9	Married-AF-spouse	Adm-clerical	Wife	White	Female	25	United-States
Ours	19	Private	9th	5	Never-married	Adm-clerical	Own-child	White	Female	25	United-States
	18	Private	HS-grad	9	Never-married	Adm-clerical	Own-child	Black	Female	20	United-States
	18	Private	HS-grad	9	Never-married	Adm-clerical	Own-child	White	Female	25	Canada

we did not re-tune or change any hyperparameters other than B. We note that our performance remains robust with this change across all metrics.

VI. CONCLUSION

In this paper, we introduced a novel counterfactual explainer called *Structured Counterfactual Diffuser* (SCD) for structured data aimed at producing highly plausible counterfactuals. Our technique leverages a diffusion model to learn complex relationships among various attributes of structured data. Via guided diffusion, our model not only exhibits high plausibility compared to the existing state-of-the-art but also shows significant improvement in proximity and diversity, while also maintaining high validity. In our analysis, we thoroughly analyze various important aspects of our proposed model, revealing useful insights. We find that our method removes the need for an explicit diversity loss by utilizing stochastic denoising that naturally produces diverse samples.

REFERENCES

[1] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv: Machine Learning*, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:11319376

[2] OpenAI, "Gpt-4 technical report," *ArXiv*, vol. abs/2303.08774, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:257532815

[3] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," *ArXiv*, vol. abs/2204.06125, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:248097655

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, "Model-agnostic interpretability of machine learning," *arXiv preprint arXiv:1606.05386*, 2016.

[5] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[6] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual explanations without opening the black box: Automated decisions and the gdpr," *Harv. JL & Tech.*, vol. 31, p. 841, 2017.

[7] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 2020, pp. 607–617.

[8] A.-H. Karimi, G. Barthe, B. Balle, and I. Valera, "Model-agnostic counterfactual explanations for consequential decisions," *ArXiv*, vol. abs/1905.11190, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:166227893

[9] W. Yang, J. Li, C. Xiong, and S. C. Hoi, "Mace: An efficient model-agnostic framework for counterfactual explanation," *arXiv preprint arXiv:2205.15540*, 2022.

[10] A. Ross, A. Marasović, and M. E. Peters, "Explaining nlp models via minimal contrastive editing (mice)," *arXiv preprint arXiv:2012.13985*, 2020.

[11] N. Madaan, I. Padhi, N. Panwar, and D. Saha, "Generate your counterfactuals: Towards controlled counterfactual generation for text," in *Proceedings of the AAAI Conference on Artificial Intelligence*, no. 15, 2021, pp. 13 516–13 524.

[12] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[13] M. Augustin, V. Boreiko, F. Croce, and M. Hein, "Diffusion visual counterfactual explanations," *Advances in Neural Information Processing Systems*, vol. 35, pp. 364–377, 2022.

[14] G. Jeanneret, L. Simon, and F. Jurie, "Diffusion models for counterfactual explanations," in *Asian Conference on Computer Vision*, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:247779169

[15] P. Sanchez and S. A. Tsaftaris, "Diffusion causal models for counterfactual estimation," in *CLEaR*, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:247011291

[16] "Diffusion-based visual counterfactual explanations – towards systematic quantitative evaluation," 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260866076

[17] X. Li, J. Thickstun, I. Gulrajani, P. S. Liang, and T. B. Hashimoto, "Diffusion-lm improves controllable text generation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4328–4343, 2022.

[18] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.

[19] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[20] X. Han, B. C. Wallace, and Y. Tsvetkov, "Explaining black box predictions and unveiling data artifacts through influence functions," *arXiv preprint arXiv:2005.06676*, 2020.

[21] L. A. Hendricks, R. Hu, T. Darrell, and Z. Akata, "Grounding visual explanations," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 264–279.

[22] Y. Goyal, Z. Wu, J. Ernst, D. Batra, D. Parikh, and S. Lee, "Counterfactual visual explanations," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2376–2384.

[23] A. Van Looveren and J. Klaise, "Interpretable counterfactual explanations guided by prototypes," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021, pp. 650–665.

[24] J. Li, W. Monroe, and D. Jurafsky, "Understanding neural networks through representation erasure," *arXiv preprint arXiv:1612.08220*, 2016.

[25] S. Feng, E. Wallace, A. Grissom II, M. Iyyer, P. Rodriguez, and J. Boyd-Graber, "Pathologies of neural models make interpretations difficult," *arXiv preprint arXiv:1804.07781*, 2018.

[26] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh, "Beyond accuracy: Behavioral testing of NLP models with CheckList," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020, pp. 4902–4912. [Online]. Available: https://www.aclweb.org/anthology/2020.acl-main.442

[27] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," *arXiv preprint arXiv:1712.06751*, 2017.

[28] M. T. Ribeiro, S. Singh, and C. Guestrin, "Semantically equivalent adversarial rules for debugging nlp models," in *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: long papers)*, 2018, pp. 856–865.

[29] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer, "Adversarial example generation with syntactically controlled paraphrase networks," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1875–1885. [Online]. Available: https://aclanthology.org/N18-1170

[30] R. Jia and P. Liang, "Adversarial examples for evaluating reading comprehension systems," *arXiv preprint arXiv:1707.07328*, 2017.

[31] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations." in *AAAI*, vol. 18, 2018, pp. 1527–1535.

[32] I. Higgins, L. Matthey, A. Pal, C. P. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:46798026

[33] A.-H. Karimi, G. Barthe, B. Balle, and I. Valera, "Model-agnostic counterfactual explanations for consequential decisions," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 895–905.

[34] R. Guidotti, A. Monreale, F. Giannotti, D. Pedreschi, S. Ruggieri, and F. Turini, "Factual and counterfactual explanations for black box decision making," *IEEE Intelligent Systems*, vol. 34, pp. 14–23, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:210931542

[35] R. Poyiadzi, K. Sokol, R. Santos-Rodriguez, T. De Bie, and P. Flach, "Feasible and actionable counterfactual explanations," 2020.

[36] A. Dhurandhar, T. Pedapati, A. Balakrishnan, P.-Y. Chen, K. Shanmugam, and R. Puri, "Model agnostic contrastive explanations for structured data," *ArXiv*, vol. abs/1906.00117, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:173990728

[37] A. Jacovi, S. Swayamdipta, S. Ravfogel, Y. Elazar, Y. Choi, and Y. Goldberg, "Contrastive explanations for model interpretability," in *Conference on Empirical Methods in Natural Language Processing*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:232092617

[38] T. Wu, M. T. Ribeiro, J. Heer, and D. S. Weld, "Polyjuice: Automated, general-purpose counterfactual generation," *arXiv preprint arXiv:2101.00288*, 2021.

[39] N. Madaan, D. Saha, and S. Bedathur, "Counterfactual sentence generation with plug-and-play perturbation," in *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE, 2023, pp. 306–315.

[40] V. Boreiko, M. Augustin, F. Croce, P. Berens, and M. Hein, "Sparse visual counterfactual explanations in image space," *ArXiv*, vol. abs/2205.07972, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:248834482

[41] P. Howard, G. Singer, V. Lal, Y. Choi, and S. Swayamdipta, "Neurocounterfactuals: Beyond minimal-edit counterfactuals for richer data augmentation," *ArXiv*, vol. abs/2210.12365, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:253098636

[42] Z. Xu, H. Lamba, Q. Ai, J. Tetreault, and A. Jaimes, "Counterfactual editing for search result explanation," *ArXiv*, vol. abs/2301.10389, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:256231426

[43] A. Frank, "Uci machine learning repository," *http://archive. ics. uci. edu/ml*, 2010.

[44] H. Zhu, "Predicting earning potential using the adult dataset," *Retrieved December*, vol. 5, p. 2016, 2016.

[45] R. P. Moro, S. and P. Cortez, "Bank Marketing," UCI Machine Learning Repository, 2012, DOI: https://doi.org/10.24432/C5K306.

[46] R. K. Pace and R. Barry, "Sparse spatial autoregressions," *Statistics & Probability Letters*, vol. 33, no. 3, pp. 291–297, 1997.

APPENDIX

In this section, we provide additional implementation details. We also provide the exact hyperparameters we used in our experiments in Table IV.

A. Diffusion Model Pre-Training

Building Row Embeddings. We transform a table row into a vector. For numeric columns or values, we first discretize them via binning into equal sized bins. Then, for the discretized numerical columns as well as the categorical columns, their discrete value is represented as an integer. For each column, we maintain a learned embedding dictionary. Corresponding to the integer value of each column, we retrieve the corresponding embedding from the learned embedding dictionary. The embeddings of all the columns are concatenated to create a vector representation of the entire row. The embeddings in the dictionary are learned during the training of the diffusion model and frozen afterwards.

B. Details of Guided Diffusion for Counterfactual Generation

Loss Coefficients. For the guiding objective function, $\lambda_{\text{validity}}$ is set to 1.0, $\lambda_{\text{proximity}}$ is set to 0.01 and $\lambda_{\text{diversity}}$ is set to 0.0001. Note that our method works with the same coefficients for all datasets, providing evidence of our method's robustness.

Classifier. The classifier is implemented as a simple 2-layer MLP with hidden dimension 768. It takes the row embeddings as input and predicts the class label depending on the task associated with each dataset. For each dataset, their corresponding MLP classifier are trained using a cross-entropy loss.

C. Plausibility Metric

In this section, we describe how we compute the plausibility metric. To measure plausibility, we compute the negative log-likelihood of the generated counterfactuals. The log-likelihood is computed with respect to the estimated data distribution modeled via an autoregressive RNN and another autoregressive Transformer model. Note that these architectures share no inductive biases or parameters with the models we evaluate, and thus, can be considered as objective measures.

RNN Model. The RNN model is trained on the tabular data by asking to recurrently predict the values within each row sequentially from left to right. The RNN is trained via teacher-forcing and cross-entropy loss for each value. The hidden dimension of the RNN model is 768.

Transformer Model. The transformer model is trained on the tabular data by asking it to predict (under causal masking) the values within each row. This essentially makes it an autoregressive model of the row. The transformer is trained to predict each row value via a cross-entropy loss conditioned on the values on the left. The hidden dimensions of the transformer model is 768. It is a transformer with 4 layers and 4 heads.

D. Additional Results

Here, we provide some additional experiment results and analyses.

Evaluation of Valid-Only Counterfactuals. In this section, we computed our metrics i.e., proximity, diversity and plausibility with only valid counterfactuals. We show these results in Table V. We find that the performance trend is similar to the trend noted without filtering away the non-valid counterfactuals, with our model SCD outperforming the baselines.

Incorporating Plausibility without Diffusion. In this section, we ask whether other traditional distribution modeling approaches e.g., VAEs, can also provide benefits in improving plausibility of the current state of the art or not? If yes, how does it compare with the use of diffusion modeling.

To test this, we created a model that we call DiCE-VAE. In DiCE-VAE, we train a VAE model to capture the data distribution in the row-embedding space. In the gradient search objective (similar to that of DICE), we simply add another term: negative ELBO or the Evidence Lower-Bound as estimated by the trained VAE model. We hypothesize that this additional loss term will prevent the search from exiting the plausible regions of the search space. The new guiding loss $\mathcal{L}$ can be formally described as:

$$\begin{aligned}\mathcal{L}(\mathbf{Z}', \mathbf{x}, f, y') = \lambda_{\text{validity}}\mathcal{L}_{\text{validity}}(\mathbf{Z}', f, y') \\ + \lambda_{\text{proximity}}\mathcal{L}_{\text{proximity}}(\mathbf{Z}, \mathbf{Z}') \\ + \lambda_{\text{diversity}}\mathcal{L}_{\text{diversity}}(\mathbf{Z}') \\ + \lambda_{\text{plausibility}}\mathcal{L}_{\text{plausibility}}(\mathbf{Z}')\end{aligned}$$

In experiments, we find that this indeed improves the plausibility in comparison to the baseline DICE, almost halving the negative log-likelihood of the generated counterfactuals from DiCE-VAE. We show the results in Table VI. We note that plausibility improves to 54.34 in DiCE-VAE as compared to Wachter and DiCE where it is 108.7 and 121.0 in Adult Income dataset. A similar trend is seen across three datasets as shown in Table VI. However, comparing with SCD using diffusion model performs even better than DiCE-VAE, thus justifying the use of diffusion model over the traditional distribution modeling approaches e.g., VAEs.

E. Ethics Statement

Building language models with steering ability can help in reducing bias, toxicity, etc. Our proposed system SCD does not support or amplify any biases and can not be exploited to generate such content. Infact, it helps in generating counterfactuals that indeed aid in making the models more explainable and bias-free. Hence, this work poses no threat of discrimination, or bias.

Model	Hyperparameters	Dataset		
		Adult Income	**UCI Bank**	**House Price**
Diffusion LM Pre-training	Batch Size	120	120	120
	# Epochs	500	500	500
	Max Text Length	11	16	9
	# Diffusion Steps	2000	2000	2000
	Learning Rate	1e-4	1e-4	1e-4
	# Learning Rate Warmup Steps	30000	30000	30000
	# Learning Rate Half Life	25000	25000	25000
	Gradient Clipping	0.05	0.05	0.05
Guided Diffusion	# Classes	2	2	2
	Weight Coefficient of Proximity Loss	0.01	0.01	0.01
	Weight Coefficient of Validity Loss	1.0	1.0	1.0
	Weight Coefficient of Diversity Loss	0.001	0.001	0.001
	Guider Learning Rate	1.5	1.5	1.5
	Vocabulary Size	2000	5000	5000
Plausibility Metric (GRU Model)	Batch Size	120	120	120
	# Epochs	500	500	500
	Vocabulary Size	2000	5000	5000
DiCE	# Classes	2	2	2
	Weight Coefficient of Proximity Loss	0.1	0.1	0.1
	Weight Coefficient of Validity Loss	1.0	1.0	1.0
	Weight Coefficient of Diversity Loss	0.0325	0.0325	0.0325
	Guider Learning Rate	2.5	2.5	2.5
	Vocabulary Size	2000	5000	5000
Wachter	# Classes	2	2	2
	Weight Coefficient of Proximity Loss	0.1	0.1	0.1
	Weight Coefficient of Validity Loss	1.0	1.0	1.0
	Guider Learning Rate	2.5	2.5	2.5
	Vocabulary Size	2000	5000	5000

TABLE IV
HYPERPARAMETERS OF OUR MODEL USED IN OUR EXPERIMENTS.

TABLE V
COMPARISON OF PLAUSIBILITY, PROXIMITY, DIVERSITY SCORES OF SCD, DICE AND WACHTER ON VARIOUS DATASETS WITH ONLY VALID COUNTERFACTUALS. FOR PROXIMITY AND DIVERSITY SCORES, HIGHER IS BETTER. FOR THE PLAUSIBILITY SCORE, LOWER IS BETTER SINCE IT CAPTURES THE NEGATIVE LOG-LIKELIHOOD OF THE GENERATED SAMPLES.

Dataset	**Plausibility** (↓)			**Proximity** (↑)		
	Wachter *et al.*	**DiCE**	**SCD**	**Wachter *et al.***	**DiCE**	**SCD**
Adult Income	110.57	120.10	**21.99**	**0.677**	0.581	0.583
UCI Bank	168.57	168.99	**74.64**	0.223	0.210	**0.323**
Housing Price	104.88	109.63	**74.57**	**0.365**	0.300	0.314
Dataset	**Diversity** (↑)					
	Wachter *et al.*	**DiCE**	**SCD**			
Adult Income	0.00	0.396	**0.414**			
UCI Bank	0.187	0.538	**0.549**			
Housing Price	0.0	**0.639**	0.534			

TABLE VI

COMPARISON OF PLAUSIBILITY, PROXIMITY, DIVERSITY, AND VALIDITY SCORES OF SCD, DICE-VAE, DICE AND WACHTER ON VARIOUS DATASETS. FOR VALIDITY, PROXIMITY, AND DIVERSITY SCORES, HIGHER IS BETTER. FOR THE PLAUSIBILITY SCORE, LOWER IS BETTER SINCE IT CAPTURES THE NEGATIVE LOG-LIKELIHOOD OF THE GENERATED SAMPLES.

Dataset	**Plausibility** (↓)				**Proximity** (↑)			
	Wachter *et al.*	**DiCE**	**DiCE-VAE**	**SCD**	**Wachter *et al.***	**DiCE**	**DiCE-VAE**	**SCD**
Adult Income	108.7	121.0	54.34	**21.21**	0.685	0.5764	0.623	**0.6173**
Housing Price	102.8	109.5	73.71	**42.91**	0.375	0.3055	0.337	**0.3417**
Dataset	**Diversity** (↑)				**Validity** (↑)			
	Wachter *et al.*	**DiCE**	**DiCE-VAE**	**SCD**	**Wachter *et al.***	**DiCE**	**DiCE-VAE**	**SCD**
Adult Income	0.002	0.3837	0.305	**0.4008**	0.9400	**0.9776**	0.847	0.7511
Housing Price	0.03	0.4289	0.607	**0.5986**	0.9999	**0.9908**	0.855	0.8526

TABLE VII

COMPARISON OF PLAUSIBILITY SCORES OF SCD, DICE-VAE, DICE AND WACHTER ON VARIOUS DATASETS WITH GRU MODEL AND A TRANSFORMER MODEL. FOR THE PLAUSIBILITY SCORE, LOWER IS BETTER SINCE IT CAPTURES THE NEGATIVE LOG-LIKELIHOOD OF THE GENERATED SAMPLES.

Dataset	**Plausibility with GRU Model** (↓)			**Plausibility with Transformer Model** (↓)		
	Wachter *et al.*	**DiCE**	**SCD**	**Wachter *et al.***	**DiCE**	**SCD**
Adult Income	108.7	121.0	**21.21**	85.18	94.70	**28.07**
UCI Bank	168.3	166.7	**42.37**	191.06	190.81	**107.69**
Housing Price	102.8	109.5	**42.91**	91.83	92.23	**49.88**