A Few Good Counterfactuals: Generating Interpretable, Plausible and Diverse Counterfactual Explanations

Barry Smyth[1,2](✉) and Mark T. Keane[1,2,3]

[1] School of Computer Science, University College Dublin, Dublin, Ireland
{barry.smyth,mark.keane}@ucd.ie

[2] Insight SFI Research Centre for Data Analytics, University College Dublin, Dublin, Ireland

[3] VistaMilk SFI Research Centre, University College Dublin, Dublin, Ireland

Abstract. Counterfactual explanations are an important solution to the Explainable AI (XAI) problem, but good, "native" counterfactuals can be hard to come by. Hence, the popular methods generate synthetic counterfactuals using "blind" perturbation, by manipulating feature values to elicit a class change. However, this strategy has other problems, notably a tendency to generate invalid data points that are *out-of-distribution* or that involve feature-values that do not naturally occur in a given domain. Instance-guided and case-based methods address these problems by grounding counterfactual generation in the dataset or case base, producing synthetic counterfactuals from naturally-occurring features, and guaranteeing the reuse of valid feature values. Several instance-guided methods have been proposed, but they too have their shortcomings. Some only approximate grounding in the dataset, or do not readily generalise to multi-class settings, or are limited in their ability to generate alternative counterfactuals. This paper extends recent case-based approaches by presenting a novel, general-purpose, case-based solution for counterfactual generation to address these shortcomings. We report a series of experiments to systematically explore parametric variations on common datasets, to establish the conditions for optimal performance, beyond the state-of-the-art in instance-guided methods for counterfactual XAI.

1 Introduction

Imagine your research paper has been reviewed by an AI that provides post-hoc explanations for its decisions. It might explain a rejection using a counterfactual statement: "*if the paper was written more clearly and the evaluation used more datasets, then it would have been accepted*". This proposes an alternative outcome (acceptance) if two aspects of the paper had been different (clearer writing *and* a stronger evaluation). This explanation is just one of many possible counterfactuals for explaining the rejection. For example, another might emphasise different aspects of the paper (perhaps related work and technical detail, for example): "*if the paper had included a better treatment of related work and*

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
M. T. Keane and N. Wiratunga (Eds.): ICCBR 2022, LNAI 13405, pp. 18–32, 2022.
https://doi.org/10.1007/978-3-031-14923-8_2

provided a clearer technical account of the main algorithm, then it would have been accepted". While we hope that neither of these options will be applied to the present work, they show how useful counterfactuals can be in providing a causal focus for how an alternative outcome *could* be achieved [3,29], rather than merely explaining why a particular outcome *was* achieved [10,22,23]. Accordingly, in recent years, there has been an explosion of research on how counterfactuals can be used in Explainable AI (XAI [1,20,36]) and algorithmic recourse [19].

From a machine learning perspective, every dataset will have some existing counterfactuals (what we call *native counterfactuals*), typically the *nearest unlike neighbours* (NUNs) of a target query to be explained [6,10,28]. However, not every NUN makes for a *good* counterfactual; for instance, it is generally agreed that *sparse* NUNs make better counterfactuals, because good native counterfactuals have few feature-differences between the query-instance and its explanatory counter-instance. However, [21] showed that 95% of datasets examined had <1% of natives with ≤2 differences (a common sparsity threshold). Hence, the most popular solutions to counterfactual generation have opted to generate synthetic counterfactuals, by perturbing query-instances using a loss function that balances proximity to the query against proximity to the decision boundary for the counterfactual class, using a scaled L1-norm distance-metric [37]. However, these proximity-driven, optimisation approaches have a tendency to generate invalid data-points, synthetic counterfactual-instances that may be out-of-distribution and/or involve feature-values that do not naturally occur [8,9,26,37].

For these reasons, many researchers have argued that counterfactual generation needs to be, somehow, grounded in the dataset of known instances [8,21,26,32]. Such *instance-guided methods* attempt to generate synthetic counterfactuals to handle a wide-range of target queries while being faithful to characteristics of the dataset [21,32]. However, current methods are *incomplete*, because they fail to identify good valid counterfactuals, and *inefficient*, because the counterfactuals they can generate are not guaranteed to be the best that could be generated, as determined by quality metrics (see below). The present paper advances a novel case-based method, using an elegant algorithm, that generates synthetic counterfactuals by directly adapting instances (native counterfactuals) in the dataset, using actual feature-values, to provide diverse, high-quality counterfactual explanations for a wide variety of target queries. In the remainder of this paper, we first set the scene for the current work by defining the properties of good counterfactual explanations and situate it relative to prior work. Then, we go on to present the details of the current novel algorithm and report several extensive experiments to test it against the current state-of-the-art.

2 Related Work

As AI systems become more widespread, the need for fairness [35], transparency [25], and explainability is increasingly important [1,17,24,31]; indeed, some governmental regulations (such as GDPR) now call for mandatory explanations for AI-based decisions [16]. At the same time, machine learning approaches that have proven to be so effective in real-world tasks (e.g. deep neural networks), appear to be among the most difficult to explain [18]. One approach to this problem is to cast such black-box models as white-box ones and then to use the latter to explain the former; for example, using post-hoc feature-based (e.g., as in LIME [34]) or example-based explanations (e.g., as in twin-systems [15,22,23]). Counterfacutal explanations are another post-hoc explanation strategy, one that is arguably better than example-based explanations [29], as they inform the user about the features that need to change in order to alter an automated decision (hence, their use in algorithmic recourse [19]). Furthermore, psychologically, people readily understand counterfactuals [2,3] and, importantly, they are implicated in causal understanding [2,14]. Finally, from a legal perspective, [37] have argued that counterfactual explanations are GDPR compliant.

In this section we consider the task of counterfactual generation. We begin by asking what makes a *good* counterfactual – one that is likely to be useful in practice – and we then review recent efforts to generate counterfactuals, paying particular attention to the difference between so-called *"blind" perturbation* approaches [5,26,27,30,37] and more recent *instance-based* approaches [21,26,32]. Because instance-based methods generate counterfactuals using feature values that naturally exist (rather than perturbed values that may not exist naturally), they enjoy certain plausibility benefits; although, state-of-the-art instance-based methods have a number of shortcomings too. We identify and discuss these deficits to motivate the new approach presented in this work.

2.1 What Are Good Counterfactual Explanations?

Intuitively, good counterfactual explanations should be easily understood by users – they should involve fewer, more plausible feature differences – and they should be available for most queries that arise. For example, a counterfactual that says *"if the paper was written more clearly, then the paper would have been accepted"* might be considered better than a more complex one saying *"if the paper was written more clearly, the evaluation more extensive and the review of the literature more comprehensive then the paper would have been accepted"*. There is a general consensus in the XAI literature that good counterfactuals should be:

- *Similar:* maximally similar to the target query, to be understandable to users.
- *Sparse:* differ in as few features as possible from the target, to be easily interpreted.
- *Plausible:* modify features/values that make sense to users (e.g., preferably from known instances).
- *Available:* for a majority of targets, to give a high degree of explanation coverage.

– *Diverse:* use a variety of features to offer counterfactuals that highlight different perspectives, either when multiple alternatives are required, or when it is useful to choose a single explanation from a set of alternatives that involve various feature differences.

The counterfactual literature has many methods that try to meet these properties, but with varying degrees of success.

2.2 Perturbation-Based Approaches

A recent review of the XAI literature has identified >100 distinct methods for computing counterfactuals [20]. Many are designed to meet the various properties of good counterfactuals by perturbing the feature-values of existing instances. For instance, Wachter et al.'s [37] seminal work generates a new counterfactual p', for a target problem p, by perturbing the features of p until a class change occurs, and in a manner that minimises the distance between p and p', $d(p, p')$.

While the approach of [37] can generate a counterfactual p' that is very similar to p, its "blind" perturbation approach can generate counterfactuals that lack sparsity [27] and diversity [30]. It can also generate counterfactuals with (potentially invalid) out-of-distribution feature values; Laugel et al. [26] showed that for some datasets this could occur in 30% of generated instances. Hence, Dandle et al. [5] have proposed modifications to the loss function to minimise the number of different features between p and p' (*diffs(p,p')*). And, Mothila et al. [30] have extended the optimisation function to deal with diversity, so that for a given p, the set of counterfactuals produced minimises the distance and feature differences within the set, while maximising the range of features changed across the set. However, the out-of-distribution problem remains an issue, even for these more advanced perturbation-based solutions.

2.3 Instance-Based Approaches

In response to these out-of-distribution problems, other researchers have argued that counterfactual explanations need to be more grounded in the feature space of the dataset. This has given rise to a family of *instance-guided techniques* that exploit known instances more directly [21,26,32,33]. FACE (Feasible and Actionable Counterfactual Explantions) is one such method that selects candidate counterfactuals that are situated in high-density regions of the dataset, where there is also a feasible path between the query and the generated counterfactual [32]. However, FACE really just approximates to the use of the dataset, as the density analysis merely informs the choice of one generated counterfactual over another.

Keane and Smyth (henceforth, KS20) [21] adopted a more direct instance-guided method, using known, good (native) counterfactuals in the dataset (and their feature values) to generate novel synthetic counterfactuals. KS20 define a *good* counterfactual to be one with ≤ 2 feature differences with respect to a target query, p (based on psychological considerations, see e.g. [11,12]). If a p' exists with

$class(p) \neq class(p')$, and if p and p' differ by no more than 2 features, then p' is a good counterfactual for p. However, often no such p' exists, hence KS20 generates a novel counterfactual by locating a *nearest like neighbour* (NLN) q ($class(p) = class(q)$) such that there exists another instance q' with $class(q) \neq class(q')$, and where q and q' differ by no more than 2 features. The pair $q - q'$ is a *counterfactual-pair* (an *explanation case* in KS20) and we refer to q' as its *counterfactual instance* and to $class(q')$ as its *counterfactual class*. KS20 generates a synthetic counterfactual, p' for p, by using the $q - q'$ pair as a template. The differences between q and q' are used to identify the feature values in p that need to be changed to produce p'; the values of the other (*matching*) features are transferred directly from p to p'. KS20 describe two variations (*direct* and *indirect*) to determine which features values to use in p' for these so-called *difference features*. In the *direct* approach they come from q' itself. In the indirect approach they come from the (like) neighbours of q'.

Thus, even though good counterfactual-pairs are rare in practice, any that do exist can be adapted in different ways, to construct many new good counterfactuals. The approach is not *guaranteed* to produce a valid good counterfactual for p, because the p' may not end up with a (predicted) class that is different from p, but KS20 showed that it regularly generated valid counterfactuals that were very similar to target queries, while remaining sparse (i.e., ≤ 2 feature differences). And because these counterfactuals were always built from existing feature-values they claimed plausibility benefits compared to perturbation methods. As this method works directly from known instances and their feature-values, by design the generated counterfactuals are within distribution (at least with respect to the values of individual features). In tests, KS20 showed that the generated counterfactuals were more similar to target queries than the native counterfactual-pairs in the dataset, thereby further supporting the within-distribution claim. However, KS20's reliance on a single nearest counterfactual-pair as the basis for synthetic counterfactual generation ultimately limits their method's performance in a number of important respects, especially in multi-class settings.

2.4 Instance-Based Shortcomings

Firstly, relying on a single counterfactual-pair means that KS20's generated counterfactuals are, by definition, based on feature values that come from a fixed set of difference features. For a given p, KS20 can only generate one type of counterfactual, because the nearest counterfactual-pair specifies one set of difference features. This limits counterfactual diversity. In a multi-class setting it may be desirable to consider counterfactuals from counterfactual-pairs that are associated with several available counterfactual classes, to generate more diverse counterfactuals, which use a variety of difference features.

If a good counterfactual cannot be generated, directly or indirectly, from the nearest counterfactual-pair, then KS20 fails, thereby limiting the availability of good counterfactuals. This problem may be especially acute in multi-class domains, where it may be feasible to generate good counterfactuals from a variety of different counterfactual classes. However, if only a single counterfactual-pair can be used then all but one of these counterfactual classes will be ignored: in other words KS20 cannot produce a valid counterfactual unless it can be generated from the nearest counterfactual-pair, even though a valid counterfactual may be available by reusing features from a different counterfactual-pair.

Even when KS20 successfully uses a nearest counterfactual-pair $(q - q')$ to construct a good counterfactual, it may not be the best available. It may be possible to generate a counterfactual that is *more* similar to p by starting with a *less* similar counterfactual-pair. In other words, there may exist another counterfactual pair, $q_2 - q'_2$, such that it is possible to generate a good counterfactual from q'_2 that is even more similar to p than the one generated from the nearest pair, $q - q'$. Even though this 'better' counterfactual can be generated, it is not available to KS20, because of its reliance of the single, most similar counterfactual-pair.

Similarly, KS20 may suffer from a *plausibility deficit* too. Even if an alternative counterfactual from q'_2 offers no similarity advantage, it may be preferable if it is based on a contrasting set of difference features that are more plausible or more "actionable". All good counterfactuals are not creating equally. Some may involve feature differences that are not within the control of the user and don't serve as an actionable explanation. Even if an alternative counterfactual could be produced, from a different counterfactual-pair, using more actionable difference features, KS20 will be blind to it.

In summary then, while KS20 enjoys the plausibility benefits of instance-based approaches, and has been shown to perform well in practice, it's reliance on a single counterfactual-pair can lead to sub-optimal performance in terms of availability, similarity, plausibility, and diversity. This criticism motivates a new approach that is tested here in a systematic set of experiments to parametrically explore its performance, using several key evaluation metrics. As KS20 is currently the state-of-the-art in instance-based counterfactual methods, it is used as the baseline in these tests. The present method simplifies the KS20 algorithm in an elegant way. Stated simply, KS20 proposed a *1NN* approach to counterfactual generation, as a *single* nearest native-counterfactual pair is used as a *template* for the counterfactual. The new method considers an intuitive kNN extension, where $k > 1$ nearest-neighbour counterfactual pairs are reused, each providing a different set of counterfactual candidates (see Algorithm 1) using potentially contrasting difference features. As we shall see, this modification at once unifies the two variations presented in KS20 (direct and indirect) for generating difference-feature values, while at the same time providing a more general-purpose counterfactual generation approach that is well suited to binary *and* multi-class domains.

3 Good Counterfactuals in Multi-class Domains

Most counterfactual methods assume that an underlying decision model (M; e.g. a deep learner) is making predictions to be explained using a generated counterfactual; that is, M is also used to determine predicted classes for the generated counterfactuals. Here, the present method aims to generate *good counterfactuals* ($\leq$2 feature differences) to explain the prediction of a target query, p. Given a set of training cases/instances, I, the approach relies on the reuse of an existing good (native) counterfactual-pair, represented as a so-called *explanation case* (XC) as in KS20. An individual explanation case, xc_d, contains a target query instance, x, and a nearby counterfactual instance x' – that is, x' is a *unlike neighbour* (*UN*) of x', meaning $class(x) \neq class(x')$ – with no more than d feature differences between x and x' as in Eqs. 1–5.

$$UN(x, x') \iff class(x) \neq class(x') \tag{1}$$

$$matches(x, x') = \{f \epsilon\ x \mid x.f \approx x'.f\} \tag{2}$$

$$diffs(x, x') = \{f \epsilon\ x \mid x.f \not\approx x'.f\} \tag{3}$$

$$xc_d(x, x') \iff UN(x, x') \ \wedge\ |diffs(x, x')| \leq d \tag{4}$$

$$XC_d = \left\{xc_d(x, x')\ \forall\ x, x' \epsilon\ I\right\} \tag{5}$$

Each explanation case, xc_d, is associated with a set of *match* features, whose values are equivalent – within some tolerance – in x and x', and a set of $\leq d$ *difference* features, with differing values. Here we assume $d = 2$ and an xc[1] acts as a template for generating new counterfactuals, by identifying features that *can* be changed (*difference features*) and those that cannot (*match features*).

3.1 Reusing the *kNN* Explanation Cases

To generate a good counterfactual for some target problem/query, p, the method first identifies the $k \geq 1$ nearest xcs with $\leq d$ differences, based on the similarity between p and each $xc.x$; see line 2 and lines 5–8 in Algorithm 1. It then constructs new counterfactuals, *cfs*, from the feature values of p and $xc.x'$ (the good counterfactual for $xc.x$), for each xc; line 3 in Algorithm 1. Importantly, this method is not limited by the specific values of the difference features in $xc.x'$, because it also considers the feature values available from other nearby instances *with the same class* as $xc.x'$ (line 11 in Algorithm 1). Thus, each generated counterfactual, *cf*, is made up of the *match feature* values (*ms*) from p and the *difference feature* values (*ds*) from $xc.x'$ *or* its like-neighbours, as shown in lines 15–19 in Algorithm 1.

3.2 Validating Candidate Counterfactuals

Each generated *cf* is associated with a predicted class, $M(cf)$, based on the underlying classification model, M, and this predicted class must be checked to validate the counterfactual. In a multi-class ($n > 2$) setting there are at least two ways to validate a candidate counterfactual. One can look for *any class change*, so *cf* is considered valid if and only if its predicted class ($M(cf)$) *differs* from p's class; this is used by KS20.

[1] For now, we drop the d without loss of generality.

A *stronger* test is to confirm that *cf* has the *same class* as the counterfactual instance used to produce it; i.e. $M(cf) = class(xc.x')$. This is stronger in a multi-class setting, because it accepts only a *single, specific class change*, compared with the $n-1$ valid classes of the weaker method. The weaker method also feels less appropriate because starting from $class(xc.x')$, but ending with a different class, seems questionable. Thus, the stronger approach (lines 20–21 in Algorithm 1) constrains the *cf* to remain within the vicinity of the original explanation class used to produce it, thereby ensuring greater plausibility.

```
Given  : p, target problem;
         I, training instances;
         d, the number of features differences allowed for a good cf;
         $XC_d$, explanation cases for d;
         k, number of XCs to be reused;
         M, underlying (classification) model.
Output:  cfs, valid, good counterfactuals for p.
1  def gen-kNN-CFs($p, I, d, XC_d, k, M$):
2      $xcs \leftarrow getXCs(p,\ XC_d,\ k)$
3      $cfs \leftarrow \{\ genCFs(p, xc, I, M) \mid xc \in xcs\ \}$
4      return $cfs$
5  def getXCs($p, XC, k$):
6      $XC' \leftarrow \{\ xc \in XC \mid class(xc.x) = class(p)\}$
7      $xcs \leftarrow sort\big(XC', key = sim(xc.x, p)\big)$
8      return $xcs[: k]$
9  def genCFs($p, xc, I, d, M$):
10     $nun \leftarrow \{\ xc.x'\ \}$
11     $nuns \leftarrow nun \cup \{\ i \in I \mid class(i) = class(xc.x')\ \}$
12     $cfs \leftarrow \{\ genCF(p, n, d) \mid n \in nuns\ \}$
13     $cfs \leftarrow \{\ cf \mid cf \in cfs \wedge validateCF(cf, xc, M)\ \}$
14     return $sort\big(cfs,\ key = sim(cf, p)\big)$
15 def genCF($p, nun, d$):
16     $ms \leftarrow \{f \mid f \in matches(p, nun)\}$
17     $ds \leftarrow \{f \mid f \in diffs(p, nun)\}$
18     $cf \leftarrow ms \cup ds$
19     return $cf$ if $(cf \neq p) \wedge (|ds| \leq d)$
20 def validateCF($cf, xc, M$):
21     return $M(cf) = class(xc.x')$
```

Algorithm 1: Generating multiple, good (for a given d) counterfactuals by reusing the k nearest explanation cases to p.

3.3 Discussion

This new method unifies an important class of instance-based approaches to counterfactual generation. It subsumes and extends KS20 to generate a set of up to $k \times m$ counterfactual candidates for a given p: there are up to m distinct candidates for each of the m unique combinations of difference feature values among the k explanation cases used. The new method also promises better coverage, plausibility and diversity. On *coverage*, generating more counterfactual candidates improves the chances of producing valid counterfactuals and, therefore, should increase the fraction of target problems that can be explained. On *plausibility*, the approach has the potential to generate counterfactuals that are even more similar to the target problem than those associated with a single nearest explanation case. Finally, on *diversity*, since different explanation cases may rely on different combinations of match/difference features, arising from the reuse of different explanation cases, then the resulting counterfactuals should draw from a more diverse set of difference-features.

4 Evaluation

We evaluate the quality of counterfactuals produced by the *knn* approach using 10 common ML datasets, with varying numbers of classes, features, and training instances, in comparison to two (KS20) baselines, using three key evaluation metrics. We generate counterfactuals with, at most, two feature differences ($d = 2$), using KS20's definition of a *good* counterfactual. Some have argued against the strictness of this 2-difference constraint, often pointing to image and time-series data, but, for such data, the count would be based on higher-level latent features (rather than pixels or time-points). Here, for comparison, the experiments also report $d = 3$ results, to accept slightly more complex good counterfactuals.

4.1 Methodology

A form of 10-fold cross-validation is used to evaluate the newly generated counterfactuals, by selecting 10% of the training instances at random to use as target problems. Then, the XC (explanation case) case-base is built from a subset of the XCs that are available from the remaining instances; we use at most 2x as many XCs as there are test instances/target problems. Finally, any remaining instances, which are not part of any selected XCs, are used to train the underlying classifier; in this case we use a gradient boosted classifier [13][2], which was found to be capable of generating sufficiently accurate classification performance across the test datasets, although, in principle, any sufficiently accurate ML model could be used.

We use Algorithm 1, to generate good counterfactuals, by varying k, the number of nearest-neighbour explanation cases, and d, the maximum number of difference features permitted in a good counterfactual. Two variants of the KS20 technique are used as baselines: (i) the *direct 1NN* variant, which generates a counterfactual from a single XC only (a limited, special-case of our *kNN*

[2] SciKitLearn, with deviance loss, a learning rate of 0.1, and 100 boosting stages.

approach that does not look beyond the counterfactual instance, $xc.x'$, of the nearest explanation case), and (ii) the *indirect* variant (*1NN**) which also considers nearby like-neighbours of $xc.x'$ as a source of extra difference features. This *1NN** variant is actually equivalent to our kNN approach with $k = 1$; note, KS20 found *1NN** to be superior to *1NN*.

Generated counterfactuals are evaluated using 3 different metrics, averaging across the test cases and folds:

– *Test Coverage:* the fraction of test queries/target problems that can be associated with a good counterfactual, to assess explanatory coverage
– *Relative Distance:* the ratio of the distance between the *closest* counterfactual (*cf*) produced and its target problem p, and the distance between the target problem $xc.x$ and the (*original*) counterfactual from the XC used to generate *cf*; thus, a relative distance <1 means the new *cf* is *closer* to p than $xc.x$ was to $xc.x'$. This is our proxy measure for reflecting plausibility[3].
– *Feature Diversity:* the fraction of unique difference features that appear in the counterfactuals produced. Note, *1NN* has the same diversity as *1NN**, since a single XC is reused and thus the same difference features appear.

4.2 Results

Figures 1 and 2 show the results for $1 \leq k \leq 100$ with $d = 2, 3$. Performance per dataset is shown as a separate line graph with statistical significance encoded as follows. If the difference between two *successive* values of k is significant ($p < 0.05$), then the corresponding points are connected by a solid line, otherwise they are connected by a dashed line; for coverage we use a *z-test* and for relative distance and diversity we use a *t-test*. Further, if a marker is filled, it means that the difference between its value and the KS20 baseline is significant ($p < 0.05$). Notice that the x-axis is non-linear, to provide greater detail for $k \leq 10$.

In Fig. 1(a) we see how the ability to produce good counterfactuals increases with k, up to a point, depending on the number of available XCs for each dataset. In all datasets, for $k > 1$, coverage is significantly greater than the baseline, and coverage increases to more than 80% of target problems for a large enough k. On average, the current kNN approach is able to increase coverage by almost a factor of 2, compared to the KS20 *1NN** method, as indicated by the *relative improvement* values for coverage in Fig. 1(d); the approximate values for k shown indicate when this maximum coverage is achieved.

Likewise in Fig. 1(b) we see that these improvements in coverage also offer statistically significant *reductions* (improvements) in relative distance, for increasing k, this time compared with *1NN* because it offers better relative distance values than *1NN** on average. Thus, by considering additional explanation cases, even those that are further away from the target problem, we can generate good

[3] As this is an instance-based technique the out-of-distribution metrics sometimes used in evaluating perturbation-based techniques are not germane.

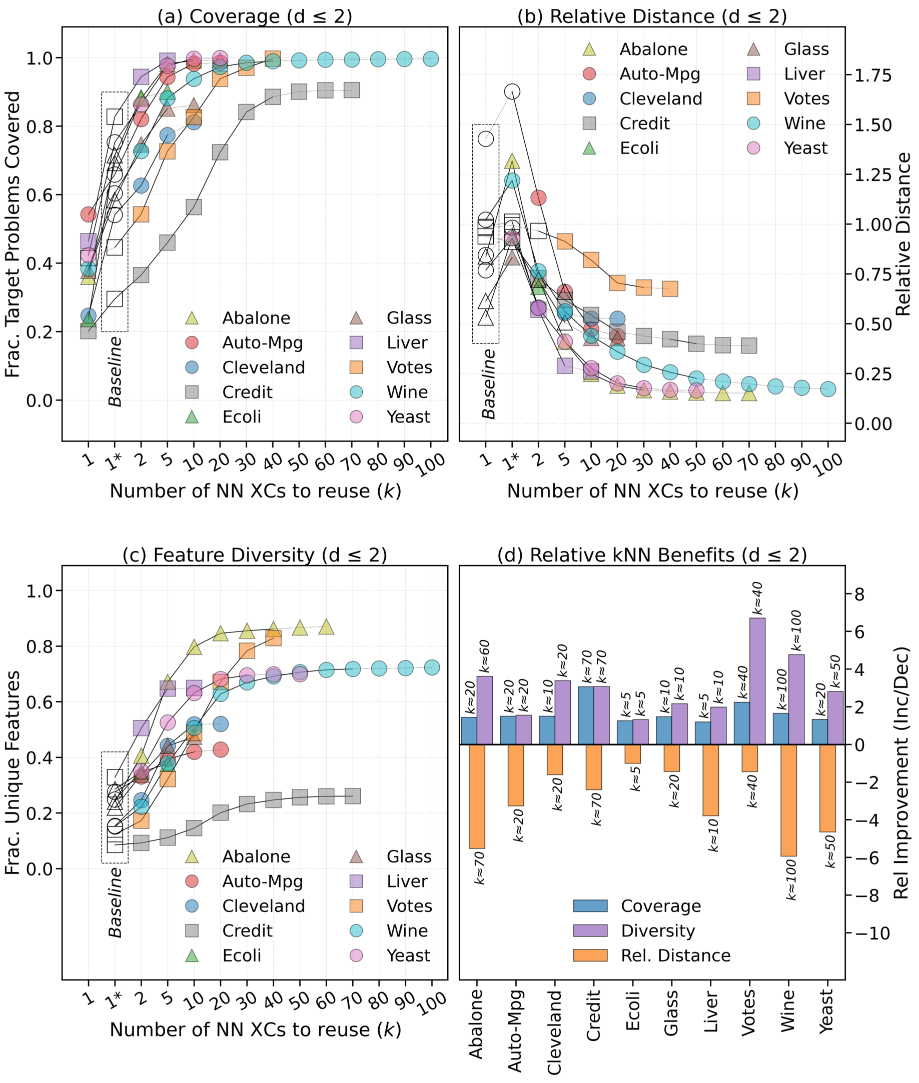

Fig. 1. Counterfactual evaluation results for $d \leq 2$ feature differences: (a) counterfactual coverage, (b) mean relative distance, and (c) counterfactual diversity along with the relative improvements (d) compared to the *1NN*/*1NN** baseline as appropriate.

counterfactuals that are closer to the target. Incidentally, the increase in relative distance for *1NN**, compared with *1NN*, is due to the significant increase in coverage offered by *1NN**, which means more valid counterfactuals participate in the relative distance calculations. Once again, in Fig. 1(d) we show a relative improvement (decrease) in these distances (compared with *1NN*): on average there is a 3x decrease in relative distance. This is usually achieved for a larger

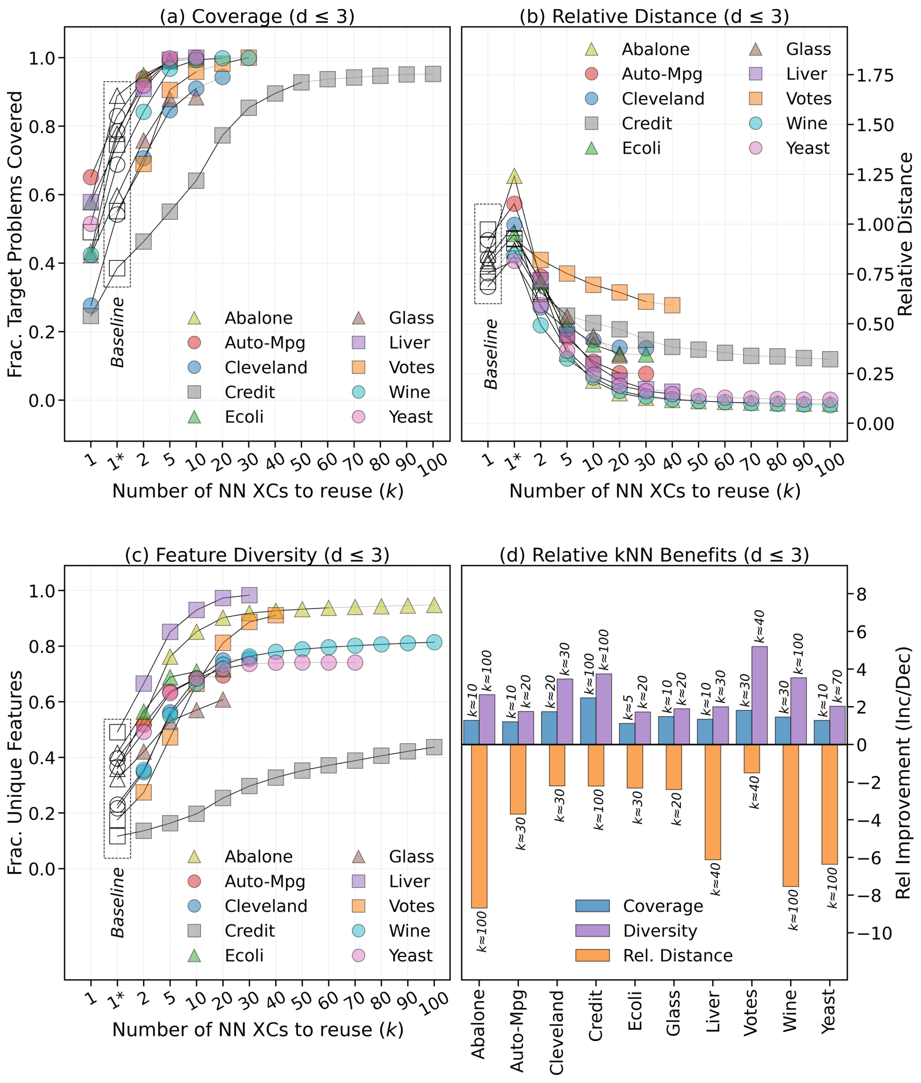

Fig. 2. Counterfactual evaluation results for $d \leq 3$ feature differences: (a) counterfactual coverage, (b) mean relative distance, and (c) counterfactual diversity along with the relative improvements (d) compared to the *1NN/1NN** baseline as appropriate.

value of k than the best coverage, which emphasises the benefits of continuing the search even after an initial good counterfactual has been located.

Finally, the results for feature diversity are shown in Fig. 1(c), once more with significant improvements for increasing values of k, although not every dataset produces counterfactuals with high levels of diversity. For example, in Fig. 1(c) the counterfactuals produced for *Credit* only include up to 25–30% of the available features as difference features. On the other hand, the counterfactuals produced for *Abalone* include over 80% of features as their difference features, while datasets such as *Auto-MPG, Cleveland, and Glass* achieve moderate levels of diversity with 45% to 50% feature participation. Nevertheless, these are considerable improvements (3x on average) compared to the diversity of the baseline (*1NN**) approach, as per Fig. 1(d).

The best value for k varies by dataset, but in practice values of k in the range $10 \leq k \leq 20$ perform well, in terms of coverage, relative distance, and diversity, for all datasets. The $d = 3$ results in Fig. 2, though not discussed in detail, also show similar results and trends, which suggests that the improvements found are not limited to $d = 2$ good counterfactuals; the best results for kNN demonstrate significant improvements over both KS20-baselines.

5 Conclusions

Counterfactuals play an important role in Explainable AI because they can be more causally informative than factual forms of explanation. However, useful native counterfactuals – those similar to a target problem but which differ in only a few (e.g. 1–2) features – can be rare in practice, leading some to propose techniques for generating synthetic counterfactuals [5,30,37]. However, such synthetic counterfactuals often rely on features which may not occur naturally, limiting their explanatory-utility. In response, others have advanced instance-based techniques to generate counterfactuals from naturally occurring feature values. The main contribution of this work is a unifying approach for instance-based counterfactual generation. A second contribution stems from its systematic evaluation of instance-based techniques to demonstrate the optimal parameters for current and previous methods, across a wide range of benchmark datasets.

There are limitations that invite future research. We focused on classification tasks, but the approach should be equally applicable to prediction tasks. The current evaluation focuses on a *like-for-like* comparison with instance-based counterfactual generation methods, but does not include a direct comparison with perturbation-based methods, mostly because the latter cannot guarantee counterfactuals with naturally occurring feature values. Nevertheless, a direct comparison between instance-based and other perturbation-based approaches is warranted and planned. It will also be worthwhile to consider additional metrics to evaluate counterfactuals such as those considered by [4,7].

Though we have provided an offline analysis of counterfactual quality, we have not yet evaluated the counterfactuals produced *in situ*, as part of a real live-user explanation setting. This will also be an important part of future research, as the utility of any counterfactual generation technique will depend critically on the nature of the counterfactuals produced and their informativeness as explanations to "real" end-users. The current tests identify optimal versions of instance-based methods that need to be considered in such future user studies.

Acknowledgements. Supported by Science Foundation Ireland via the Insight SFI Research Centre for Data Analytics (12/RC/2289) and with the Department of Agriculture, Food and Marine via the VistaMilk SFI Research Centre (16/RC/3835).

References

1. Adadi, A., Berrada, M.: Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). IEEE Access **6**, 52138–52160 (2018)
2. Byrne, R.M.: The Rational Imagination: How People Create Alternatives to Reality. MIT Press, Cambridge (2007)
3. Byrne, R.M.: Counterfactuals in Explainable Artificial Intelligence (XAI). In: IJCAI-19, pp. 6276–6282 (2019)
4. Chou, Y.L., Moreira, C., Bruza, P., Ouyang, C., Jorge, J.: Counterfactuals and causability in explainable artificial intelligence: theory, algorithms, and applications. Inf. Fusion **81**, 59–83 (2022)
5. Dandl, S., Molnar, C., Binder, M., Bischl, B.: Multi-objective counterfactual explanations. arXiv preprint arXiv:2004.11165 (2020)
6. Dasarathy, B.V.: Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design. IEEE Trans. Syst. Man Cybern. **24**(3), 511–517 (1994)
7. Del Ser, J., Barredo-Arrieta, A., Díaz-Rodríguez, N., Herrera, F., Holzinger, A.: Exploring the trade-off between plausibility, change intensity and adversarial power in counterfactual explanations using multi-objective optimization. arXiv preprint arXiv:2205.10232 (2022)
8. Delaney, E., Greene, D., Keane, M.T.: Instance-based counterfactual explanations for time series classification. In: Sánchez-Ruiz, A.A., Floyd, M.W. (eds.) ICCBR 2021. LNCS (LNAI), vol. 12877, pp. 32–47. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86957-1_3
9. Delaney, E., Greene, D., Keane, M.T.: Uncertainty estimation and out-of-distribution detection for counterfactual explanations. In: ICML21 Workshop on Algorithmic Recourse. arXiv-2107 (2021)
10. Doyle, D., Cunningham, P., Bridge, D., Rahman, Y.: Explanation oriented retrieval. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 157–168. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28631-8_13
11. Förster, M., Hühn, P., Klier, M., Kluge, K.: Capturing users' reality: a novel approach to generate coherent counterfactual explanations. In: Proceedings of the 54th Hawaii International Conference on System Sciences, p. 1274 (2021)
12. Förster, M., Klier, M., Kluge, K., Sigler, I.: Fostering human agency: a process for the design of user-centric XAI systems. In: Proceedings of the International Conference on Information Systems (ICIS) (2020)
13. Friedman, J.H.: Stochastic gradient boosting. Comput. Stat. Data Anal. **38**(4), 367–378 (2002)
14. Gerstenberg, T., Goodman, N.D., Lagnado, D.A., Tenenbaum, J.B.: A counterfactual simulation model of causal judgments for physical events. Psychol. Rev. (2021)
15. Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M., Kagal, L.: Explaining explanations. In: Proceedings of the IEEE 5th International Conference on Data Science and Advanced Analytics, pp. 80–89. IEEE (2018)
16. Goodman, B., Flaxman, S.: European union regulations on algorithmic decision-making and a "right to explanation". AI Mag. **38**(3), 50–57 (2017)
17. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Comput. Surv. **51**(5), 1–42 (2018)

18. Gunning, D.: Explainable Artificial Intelligence (XAI). DARPA, Web **2**(2) (2017)
19. Karimi, A.H., von Kügelgen, J., Schölkopf, B., Valera, I.: Algorithmic recourse under imperfect causal knowledge. In: NIPS 33 (2020)
20. Keane, M.T., Kenny, E.M., Delaney, E., Smyth, B.: If only we had better counterfactual explanations. In: Proceedings of the 30th International Joint Conference on Artificial Intelligence, IJCAI-21, pp. 4466–4474 (2021). https://doi.org/10.24963/ijcai.2021/609
21. Keane, M.T., Smyth, B.: Good counterfactuals and where to find them: a case-based technique for generating counterfactuals for explainable AI (XAI). In: Watson, I., Weber, R. (eds.) ICCBR 2020. LNCS (LNAI), vol. 12311, pp. 163–178. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58342-2_11
22. Kenny, E.M., Keane, M.T.: Twin-systems to explain artificial neural networks using case-based reasoning. In: IJCAI-19, pp. 2708–2715 (2019)
23. Kenny, E.M., Keane, M.T.: Explaining deep learning using examples: optimal feature weighting methods for twin systems using post-hoc, explanation-by-example in XAI. Knowl.-Based Syst. **233**, 107530 (2021)
24. Kusner, M.J., Loftus, J.R.: The long road to fairer algorithms. Nature (2020)
25. Larsson, S., Heintz, F.: Transparency in artificial intelligence. Internet Policy Rev. **9**(2) (2020)
26. Laugel, T., Lesot, M.J., Marsala, C., Renard, X., Detyniecki, M.: The dangers of post-hoc interpretability. In: IJCAI-19, pp. 2801–2807. AAAI Press (2019)
27. McGrath, R., et al.: Interpretable credit application predictions with counterfactual explanations. In: NIPS Workshop on Challenges and Opportunities for AI in Financial Services (2018)
28. McKenna, E., Smyth, B.: Competence-guided case-base editing techniques. In: Blanzieri, E., Portinale, L. (eds.) EWCBR 2000. LNCS, vol. 1898, pp. 186–197. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44527-7_17
29. Miller, T.: Explanation in artificial intelligence: insights from the social sciences. Artif. Intell. **267**, 1–38 (2019)
30. Mothilal, R.K., Sharma, A., Tan, C.: Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the Conference on Fairness, Accountability, and Transparency, pp. 607–617 (2020)
31. Muhammad, K.I., Lawlor, A., Smyth, B.: A live-user study of opinionated explanations for recommender systems. In: IUI, pp. 256–260 (2016)
32. Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., De Bie, T., Flach, P.: Face: feasible and actionable counterfactual explanations. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, pp. 344–350 (2020)
33. Ramon, Y., Martens, D., Provost, F., Evgeniou, T.: A comparison of instance-level counterfactual explanation algorithms for behavioral and textual data: SEDC, LIME-C and SHAP-C. Adv. Data Anal. Classif. **14**(4), 801–819 (2020). https://doi.org/10.1007/s11634-020-00418-3
34. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should i trust you? In: Proceedings of the ACM SIGKDD, pp. 1135–1144 (2016)
35. Russell, C., Kusner, M.J., Loftus, J., Silva, R.: When worlds collide: integrating different counterfactual assumptions in fairness. In: NIPS, pp. 6414–6423 (2017)
36. Verma, S., Dickerson, J., Hines, K.: Counterfactual explanations for machine learning: a review. arXiv:2010.10596 (2020)
37. Wachter, S., Mittelstadt, B., Russell, C.: Counterfactual explanations without opening the black box. Harv. J. Law Tech. **31**, 841 (2017)