ContEx: Contextualized Explanations via SimplEx

Sam Houliston
DAMPT
University of Cambridge
sh2198@cam.ac.uk

Abstract

Post-hoc interpretability approaches have been proven to be powerful tools to generate explanations for the predictions made by a trained blackbox model. In addition, the needs of end-users must be carefully considered when providing useful and contextually-relevant explanations. In this work, we address this by extending the SimplEx algorithm by Crabbé et al. [2021] which provides local post-hoc explanations of a *black-box* prediction by extracting similar examples from a corpus and by revealing the relative importance of their features. Our method ContEx improves on SimplEx by providing both a contextually-relevant corpus selection scheme and the study of feasible counterfactual examples. The corpus selection scheme remains faithful to the underlying data distribution and respect conditions specified by the user. We generate counterfactual examples with the same principles via the FACE algorithm from [Poyiadzi et al., 2020], and then apply SimplEx's projected jacobians analysis to explain a sequence of examples from the test point to the counterfactual. Such a sequence enables users to draw a quantitatively interpretable story of the shortest realizable path from one prediction to another type.

1 Introduction and Related Work

We begin with a brief overview and motivation for the study of model explainability, followed by a review of the related works in generating counterfactual explanations, and then a discussion of SimplEx, its strengths and its limitations.

1.1 Explainable Artificial Intelligence

The evergrowing success and application of machine learning is well justified: models achieve record accuracy and performance in a range of application domains from healthcare, to automation and finance, thanks to more involved architectures, powerful computing resources and richer datasets. For people to work and live in healthy symbiosis with these powerful machine learning models they must trust and understand them. Better understanding the decision process of these systems is also a pathway to their improvement. The field of eXplainable AI (XAI) answers this call and considers the design of inherently interpretable models or the development of post-hoc explainability methods that help contextualise predictions of a model post training.

While there isn't an absolute dichotomy in performance (in the broad sense) between black-box models and *white-box* models which are inherently interpretable by design, restricting to a class of inherently interpretable models often comes at the cost of lower prediction accuracy [Rai, 2019, Rudin and Radin, 2019]. Post-hoc techniques are desireable as address the interpretability of black-box models by complementing their predictions with various kinds of explanations without sacrificing their accuracy. These techniques can provide two types of explanations: local and

Preprint. Under review.

global explanations. Local explanations address individual predictions whereas global explanations seek to provide an understanding of the overall decision process, by for example grouping local explanations, establishing a link between latent representations and human-defined concept classes, or model distillation to deduce. In this work we consider local post-hoc explanations and build on the SimplEx method which employs *feature importance explanations* and *example-based explanations*. For related works in these areas, please refer to the original SimplEx paper [Crabbé et al., 2021].

Counterfactual (CF) explanations indicate how an instance should (minimally) change to alter its prediction, however, they are hypothetical by nature and often subject to the Rashomon effect: many different changes in the features can result in the same change in prediction. Laugel et al. [2019] investigate the relevance of generated CF explanations- whether can be justified, i.e. continuously connected to some ground-truth data. They designed a test to highlight the risk of generating undesirable CF explanations and apply this test to several datasets and classifiers to show that the risk is high. Mothilal et al. [2020] designed a method that focuses on generating feasible and diverse CF explanations by taking into account user-defined feasibility conditions. They also propose multiple metrics (proximity, validity and diversity) to evaluate their explanations. Rawal and Lakkaraju [2020] bring in causality to their counterfactual analysis by proposing a model agnostic framework called Actionable Recourse Summaries (ArES) which takes into account recourse sets - user defined constraints on the features - when generating counterfactual explanations. Recourse is broadly defined as the ability (of a person) to change the prediction of a machine learning model by altering actionable input variables. The work of [Poyiadzi et al., 2020] is of great interest and the basis of our extension of SimplEx: they designed an algorithm to find existing counterfactuals (to a test data point) within a dataset that are coherent with the underlying data distribution and supported by the "feasible paths" of change, whose conditions can be tailored by feature constraints specific to the problem at hand. Their method involves first finding a set of data points that respect given conditions and that are sufficiently close to the test point. Then a path across this set is formed from the test point towards a differently-classified point: the counterfactual. Applying SymplEx's feature analysis to this counterfactual and the intermediary examples may reveal much about which features are most responsible for the changes in prediction/latent representation.

1.2 SimplEx

We now highlight and discuss some of the merits and demerits of SimplEx.

Advantages	Limitations
Marries feature-importance and example-based explanations in post-hoc fashion.	Only considers similar examples in latent space (i.e. representation and decision-wise). This can be supplemented by counterfactuals or examples similar in input space.
Corpus of examples freely designated by user.	No scheme to select the optimal corpus for a given model and dataset.
Innovative approximation in latent space by corpus decomposition, is useful to understand which samples the model classifies similarly.	From a user standpoint it may be more meaningful to see how similar samples are mapped *differently* in latent space by the model.
Integrated jacobians improves upon integrated gradients method to quantify the contribution of different features to the latent representation.	Optimal choice of baseline to be proven. (Averaging variables in the context of medicine is delicate, with the presence of bimodal variables such as height, weight, etc..).

Table 1: Pros and cons of SimplEx

1.3 Contribution

A Meaningful Corpus Selection Scheme

SimplEx seeks to best approximate the test point's latent representation as a linear combination of the latent representation of corpus samples. This is useful from an explainability standpoint, as it uncovers which corpus samples (from the input space) are most-similarly mapped in latent representation by the model (and thus most-similarly classified, the last layer being a linear combination of the latent representation weights). This can lead to SimplEx providing examples that are very different to the test sample in feature space but that are classified similarly. While this may be useful in understanding the model's mechanics (especially when paired with feature importance analysis via integrated jacobians), from the user perspective, it would be more meaningful to focus on a corpus decomposition of "achievable" samples which are similar feature-wise to the test sample.

To address this, we propose to construct a relevant corpus around the test sample by following a similar procedure to the graph-generating algorithm in the FACE method by Poyiadzi et al. [2020]. The new corpus formed is formed from samples in a high density region which includes the test sample, taking into account user-defined conditions on the data (such as "must be female" or "underweight" in a medical context). The corpus is representative of the underlying data distribution. In addition to improved explainability, this modification ensures the corpus decomposition to be more reliable as classifiers tend to be less trustworthy in sparsely populated regions of the data space, especially close to a decision boundary [Poyiadzi et al., 2020]. With this new corpus, SimplEx can generate examples that are both relevant in the input space, and similar in latent space, and perhaps lead to more granular feature attribution analysis (given the corpus is already close to the test sample).

Explaining (a sequence to) Counterfactuals with SimplEx

A counterfactual example alone is useful; a counterfactual example equipped with feature-importance scores is even more useful. We propose to augment SimplEx's analysis by including counterfactual examples. By following through with the method from [Poyiadzi et al., 2020] a path can be generated, within the corpus, from the test point to a feasible counterfactual example. In addition to providing feature importance scores to the counterfactual (with respect to a baseline), we propose to provide scores for each example in the path from the test sample to the counterfactual, provided the sequence isn't too long. This involves calculating integrated jacobians for each example using the previous node as a reference.

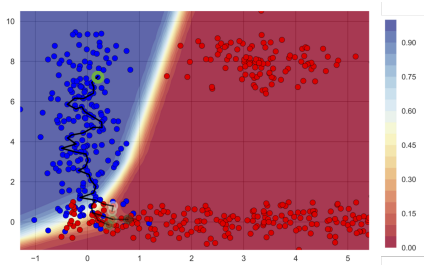


Figure 1: Example of five shortest feasible paths from test point (green) to counterfactual data point (highlighted and red) in feature space.
Image edited from Poyiadzi et al. [2020].

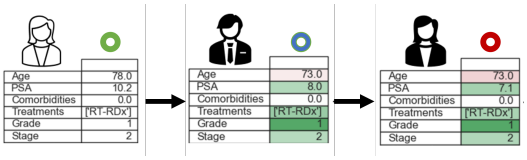


Figure 2: An example output of ContEx displaying intermediary point (blue) between the test (green) and counterfactual (red), with their feature importance analysis.
Image edited from Crabbé et al. [2021].

2 Method

2.1 Problem Formalism

We adopt the same problem setting as in the original SimplEx paper. Let $\mathcal{X} \in \mathcal{R}^{d_\mathcal{X}}$ be an input (or feature) space and $\mathcal{Y} \in \mathcal{R}^{d_\mathcal{Y}}$ dY be an output (or label) space, where $d_\mathcal{Y}$ and $d_\mathcal{X}$ are respectively the dimension of the input and the output space. Our task is to explain individual predictions of a given

black-box model $\mathbf{f} : \mathcal{X} \to \mathcal{Y}$. We make the following assumption on the family of black-boxes that we wish to interpret.

Assumption 2.1 *Black-box Restriction*

We restrict to black-boxes $\mathbf{f} : \mathcal{X} \to \mathcal{Y}$ *that can be decomposed as* $\mathbf{f} = \mathbf{l} \circ \mathbf{g}$*, where* $\mathbf{g} : \mathcal{X} \to \mathcal{H}$ *maps an input* $\mathbf{x} \in \mathcal{X}$ *to a latent vector* $\mathbf{h} = \mathbf{g}(\mathbf{x}) \in \mathcal{H}$ *and* $\mathbf{l} : \mathcal{H} \to \mathcal{Y}$ *linearly maps a latent vector* $\mathbf{h} \in \mathcal{H}$ *to an output* $\mathbf{y} = \mathbf{l}(\mathbf{h}) = \mathbf{A}\mathbf{h} \in \mathcal{Y}$*. In the following we call* $\mathcal{H} \in \mathcal{R}^{d_{\mathcal{H}}}$*. Typically, this space has higher dimension than the output space* $d_{\mathcal{H}} \geq d_{\mathcal{Y}}$

2.2 Corpus Selection Scheme

The corpus can be formed with the graph and its weighted edges generated by the FACE Counterfactual generator which we first describe (see Algorithm 1). To construct a graph, the FACE algorithm starts with the entire data set. It attributes a weight to every pair of points, commensurate to their density (approximated by a kernel density estimator), their proximity, and whether they satisfy the user-defined feasibility function (lines 1-7). Once all weights attributed, the set of candidate counterfactual targets $\mathbf{I_{CT}}$ is formed by all points with sufficient density, and sufficient probability of originating from a different/target class(lines 8-11).

Algorithm 1: FACE Counterfactual Generator

input : Data ($X \in \mathbb{R}^d$), density estimator ($\hat{p} : \mathcal{X} \to [0,1]$), probabilistic predictor ($clf : \mathcal{X} \to [0,1]$), distance function ($d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$), distance threshold ($\epsilon > 0$), weight function ($w : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$), and conditions function ($c : \mathcal{X} \times \mathcal{X} \to \{True, False\}$).

output: Graph (V, E, W) and candidate targets (I_{CT}).

/* Construct a graph. */

1 **for** *every pair* (x_i, x_j) *in* X **do**

2 **if** $d(x_i, x_j) > \epsilon$ ***and*** $c(x_i, x_j)$ ***is*** *True* **then**

3 $i \nsim j$

4 $w_{ij} = 0$

5 **else**

6 $i \sim j$

  /* In this case we use Equation 2 (KDE). This should be adjusted for k-NN and ϵ-graph constructions by using Equation 3 and 4 respectively. */

7 $w_{ij} = w(\hat{p}(\frac{x_i + x_j}{2})) \cdot d(x_i, x_j)$

/* Get a set of candidate targets. */

8 $I_{CT} = \{\}$

9 **for** x_i *in* X **do**

10 **if** $clf(x_i) \geq t_p$ ***and*** $\hat{p}(x_i) \geq t_d$ **then**

11 $I_{CT} = I_{CT} \cup i$

In their implementation, the authors of FACE initialised the weight function as $w(z) = -\log z$ and the $l2$-norm as the distance function. The authors recommend to first apply the algorithm with a fairly "restrictive" configuration, to be subsequently being relaxed until a counterfactual is found. They observe the following pattern during this process: (a) no counterfactual is generated, (b) an appropriate counterfactual is generated, and (c) a bad counterfactual is generated. The authors also provide a computational complexity analysis of their algorithm and suggest alternative density estimation techniques such as k-NN and ϵ-graphs.

We wish to construct the corpus with points that are sufficiently close to the test point, sufficiently likely and that satisfy the user constraints. We suggest to alter the FACE algorithm by adding the following lines of code after line 7 of Algorithm 1. These involve checking whether the point $\mathbf{x_i}$ is sufficiently close to $\mathbf{x}_{\text{test}}$ (to a threshold ϵ_{test}), and add little computational cost.

$$\textbf{if } d(\mathbf{x_i}, \mathbf{x}_{\text{test}}) \leq \epsilon_{\text{test}} \quad : \tag{1}$$

$$\mathcal{C} = \mathcal{C} \cup \mathbf{x_i}. \tag{2}$$

Accuracy of the New Corpus Decomposition

SimplEx can be run on this new corpus. Given that the corpus examples are close to the test point in feature space, they should be similarly classified and thus should still provide an accurate corpus decomposition. This is to be empirically verified by comparing the corpus-residuals (as defined in the original paper) from the new (constrained and distribution concurrent) corpus and the entire data set.

2.3 Explaining Counterfactuals with SimplEx

We propose to use the FACE algorithm to identify a counterfactual example and its shortest path of samples to the test point (according to the weights assigned in Algorithm 1). Our method consists of picking informative mid-destinations in this path and calculating their projected jacobians, see Figure 3 for an illustration, and Figure 2 for an example final output. Algorithm 2 describes our counterfactual and explanation generation process: we first generate the counterfactual following FACE, and then use the generated path extract k examples to finally calculate their projected jacobians following SimplEx's method. The implementation details for the calculation of the projected jacobians can be found in the appendix of the SimplEx paper.

Note: we take the sets or arrays to start at 1 and not 0.

Algorithm 2 ContEx: Sequence to Counterfactual Explainer

Input: Data: $X \subset \mathcal{X}$, Set of candidate targets: $I_{CT} \subset X$, Test point: $\mathbf{x_{test}} \in \mathbf{X}$,
Graph output from Algorithm 1: $\mathcal{G} = (V, E, W)$,
Maximum number of explanations: k.
Result: Path to counterfactual: $\mathcal{P}$, Set of explained examples $E = \left\{ \left(x^c, (p_i^c)_{i=1}^{d_{\mathcal{X}}} \right) \right\}_{c \leq k}$.

`/* Shortest path from test point to a CF from` I_{CT} `*/`
$\mathcal{P} \leftarrow$ `Dijkstra(graph=`$\mathcal{G}$`, start=`$\mathbf{x}_{\text{test}}$`, target=`I_{CT}`)`
$p \leftarrow |\mathcal{P}|$

`/* Extract at most` k `examples from` $\mathcal{P}$ `and calculate jacobians */`
$i \leftarrow 2$
while $i \leq \min(p, k)$ **do**
  $E = E \cup (\mathcal{P}_i,$ `JacobianProjections(`$\mathcal{P}_i$`, reference=`$\mathcal{P}_{i-1}$`,` $\mathbf{g}$`))`
  if $p \leq k$ **then**
    $i \leftarrow i + 1$
  else
    `/* Take` k `evenly-spaced examples from` $\mathcal{P}$ `*/`
    $i \leftarrow i + \lceil p/k \rceil$
  end if
end while
Return: $\mathcal{P}$, E

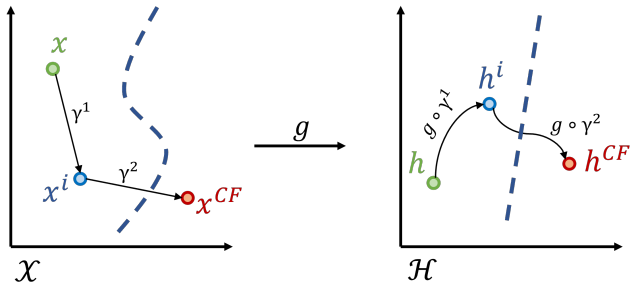


Figure 3: Illustration of a path from the test point (green) to the feasible counterfactual (red) through an intermediary point (blue). The dotted line represents the models' decision boundary.

Validation of Counterfactuals

To validate the counterfactual examples for our model and test point, we can apply relevant test metrics from previous works mentioned in the related works section of this paper.

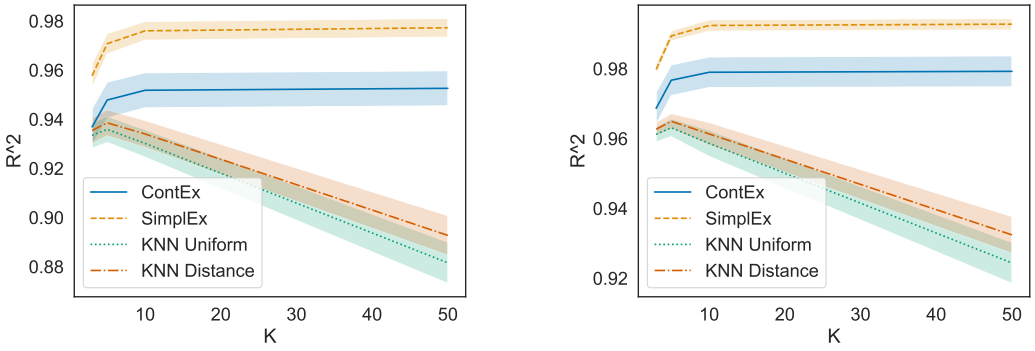
3 Experiments

We have implemented a working prototype of ContEx with its corpus selection scheme and feasible counterfactual analysis. First, we evaluate whether our automatic corpus selection scheme retains a good quality of latent space approximations. Second, we present an example output of a working prototype of ContEx explanations. The code for the experiments is available on the Github repository `https://github.com/samhouliston/Contex`. For the first experiment, we have created an explainer file named `contex.py`, and we have modified the experiment script `mnist.py`, and the figure plotting script `plot_results.py`. For the second experiment, we have edited the MNIST use case of the `use_case.py` script.

3.1 Precision of Corpus Decomposition

The purpose of this experiment is to confirm the proposed corpus selection scheme allows good approximations of the latent representations of test samples. To this end, we replicate the study of SimplEx's approximation quality by Crabbé et al. [2021] on the MNIST dataset. We briefly recall their experimental setup and metrics.

The MNIST dataset $\mathcal{D}$ is split into a training set $\mathcal{D}_{train}$ and test set $\mathcal{D}_{test}$. We train a black-box convolutional neural network $\mathbf{f}$ to classify the images. A corpus of 1000 images $\mathcal{C} \subset \mathcal{D}_{train}$ is randomly sampled from the training set, and a set of test examples $\mathcal{T} \subset \mathcal{D}_{test}$ is sampled from the test set. For each test sample $\mathbf{x} \in \mathcal{T}$, we calculate the latent representation $\mathbf{h} = \mathbf{g}(\mathbf{x})$ and let the chosen method use K corpus examples to build an approximation. We repeat the experiment for several values of K and with different latent approximation methods. For ContEx, we further refine the corpus to $\mathcal{C}_{cont} \subset \mathcal{C}$ by the selection scheme described in section 2.2. In our implementation we use the distance between neighbours as a rough proxy[1] for density and keep the 60th percentile of densest points ($|\mathcal{C}_{cont}| = 600$). In the context of MNIST image classification, we abstain from imposing user-defined conditions on the corpus examples. Then, the corpus latent decomposition is computed exactly as in SimplEx. We compare this approximation to that obtained from SimplEX on $\mathcal{C}$, to the KNN Uniform method and the KNN Distance method.



(a) $R^2_{\mathcal{H}}$ score for the latent approximation

(b) $R^2_{\hat{\mathcal{Y}}}$ score for the output approximation

Figure 4: Precision of corpus decomposition for MNIST (avg ± std).

Results The results for ContEx, SimplEx and the KNN baselines are presented in Figure 4. Our results concur with all of the author's observations. ContEx's performance -although high- is systematically lower than SimplEx but follows the same response to increasing values of K. This is expected, as in the formation of the distribution-faithful corpus, some examples are discarded : this explains the small difference in accuracy with SimplEx, as well as ContEx's higher standard deviations.

Our scheme (much like the FACE algorithm) requires choosing a few hyperparameter values, such as cutoff distance or density. These values are particularly sensitive to the data in question and would require some hyperparameter tuning for optimal application. In our experiments we defined these as proportions of our corpus and data, for example, "discard the 20th percentile of examples with

[1]The kernel density estimator method was ill-suited to process such high-dimensionality data.

highest average distance to neighbour" or "calculate the average distance from every point to its closest 10% of the corpus members".

3.2 Use case demonstration with MNIST Data Set

We present a use case of ContEx where a classification prediction of a handwritten digit is explained by a corpus decomposition, and a counterfactual example. For this demonstration, we adapted the `use_case.py` script provided by the SimplEx authors. We use a corpus size of 300 images in this experiment. Figure 5 displays the corpus decomposition of our test sample, as well as the feature importance scores. Figure 6 displays the 'closest realizable' counterfactual example obtained, with two feature-importance illustrations obtained via the projected jacobians analysis by taking different baselines. Figure 6 (b) is computed by taking the usual baseline consisting of a black image, while figure 6 (c) takes the test point as a reference.

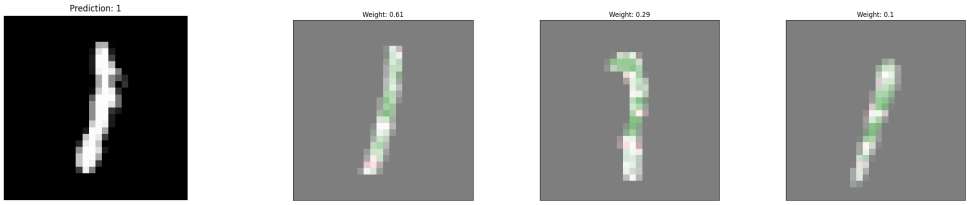
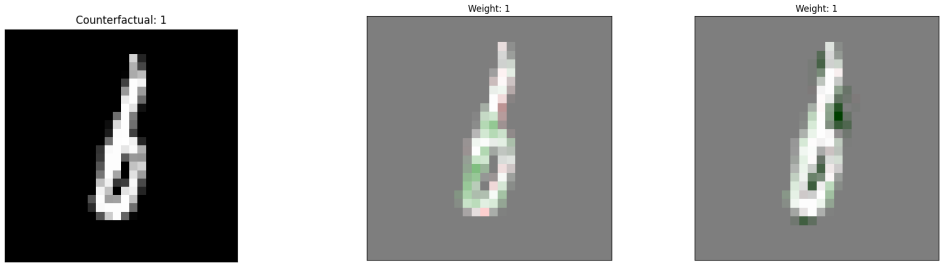


Figure 5: SimplEx decomposition, $C = 300$, $n_{\text{keep}} = 3$



(a) Counterfactual example (b) Standard baseline (c) Test point as baseline

Figure 6: Projected jacobians analysis of counterfactual, with different baselines

Indeed, the counterfactual example is similar to the test sample; it is however correctly classified by the model as "6". By analysis of the feature importance with respect to the test input as a baseline in figure 6 (c), we notice that the important pixels correspond to the regions uniquely relevant to the number six: (in typography lexicon) the concavity towards the right above the "eye", the inside of the eye, and the bottom "bowl" of the number. We can argue that the CNN model has used these features effectively to learn a meaningful representation and is able to differentiate between a 1 and a 6. We recognize such analysis is sensitive to the baseline, however our example but translation invariance to the reference image. Performing this with dis-centered test-images could help quantify how robust the model is to translation.

We acknowledge that, given the smaller corpus size (300) and our simplified proxy of the corpus selection scheme / graph generator, we have not generated "intermediate samples" that would appear in the path from the test point to the counterfactual (as depicted in figure 1 and 2). Our graph edge weights are high and this forces the shortest path to simply be the direct connection between the test point and the closest counterfactual (differently-classified corpus member). We believe a finer implementation that follows algorithm 1 more closely would lead to a more complete explanation. In

spite of this, we note that seeing the counterfactual with its feature-importance analysis adds much to the understanding of the network's prediction.

4 Conclusion and Future Work

By taking a closer look at SimplEx and its practicality from a user standpoint, we identified the need to provide explanations that are more contextually relevant. We address this via two extensions to the algorithm. First, we describe an automatic corpus selection scheme that focuses on samples similar to the test sample, that are faithful to the underlying data distribution, and that respect conditions specified by the user. Second, we proposed to augment SimplEx, which already marries feature-importance and example-based explanations, by including an analysis of "feasible" counterfactual examples -another pillar method of explainable AI. Our proposed extensions leverage the strengths of SimplEx and address its limitations to provide a complete, personalized and deployable tool that serves to contextualise predictions by black-box models.

ContEx was developed with medical applications in mind, and is especially suited -but not limited to- prediction problems involving tabular data. This method, like SimplEx, is adaptable to time series data. Further improvements to the counterfactual explanations would be to extend the FACE path algorithm to generate *diverse* counterfactual examples. Investigating the choice of baseline $\mathbf{x_0}$ in the calculation of integrated jacobians would provide methodological improvement. In the same spirit as XRAI algorithm by Google [Kapishnikov et al., 2019], a more complete analysis may consist of choosing multiple baselines. Alternatively, the choice of baseline could be justified from an information-theoretic point of view, taking into account the test-point's distribution.

The projected jacobians analysis pioneered by SimplEx's authors opens numerous promising avenues of investigation. In the context of semi/self-supervised learning, applying projected jacobians could help study the relative effectiveness of augmentations or pretext tasks in the training of a feature encoder. For contrastive learning, we could track the improvement (separation) of representations throughout training to establish which features are the most relevant, and also use this to compare the working of different contrastive loss functions. More ambitious work could aim to tackle the understanding and translation of latent-space Concept Activation Regions [Crabbé and van der Schaar, 2022] in terms of the input feature space.

References

Jonathan Crabbé, Zhaozhi Qian, Fergus Imrie, and Mihaela van der Schaar. Explaining latent representations with a corpus of examples. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. Face: Feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. ACM, feb 2020. doi: 10.1145/3375627.3375850. URL `https://doi.org/10.1145%2F3375627.3375850`.

Arun Rai. Explainable ai: from black box to glass box. In *Journal of the Academy of Marketing Sciences*, 2019.

C. Rudin and J. Radin. Why are we using black box models in ai when we don't need to? a lesson from an explainable ai competition. *Harvard Data Science Review*, 2019. URL `https://doi.org/10.1162/99608f92.5a8a3a3d`.

Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. The dangers of post-hoc interpretability: Unjustified counterfactual explanations. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2019. doi: 10.48550/ARXIV.1907.09294. URL `https://arxiv.org/abs/1907.09294`.

Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. ACM, jan 2020. doi: 10.1145/3351095.3372850. URL `https://doi.org/10.1145%2F3351095.3372850`.

Kaivalya Rawal and Himabindu Lakkaraju. Beyond individualized recourse: Interpretable and interactive summaries of actionable recourses. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. doi: 10.48550/ARXIV.2009.07165. URL https://arxiv.org/abs/2009.07165.

Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viégas, and Michael Terry. Xrai: Better attributions through regions. In *International Conference on Computer Vision (ICCV)*, 2019.

Jonathan Crabbé and Mihaela van der Schaar. Concept activation regions: A generalized framework for concept-based explanations. In *International Conference on Machine Learning (ICML)*, 2022. doi: 10.48550/ARXIV.2209.11222. URL https://arxiv.org/abs/2209.11222.