Diverse, Global and Amortised Counterfactual Explanations for Uncertainty Estimates

Dan Ley[1], **Umang Bhatt**[1,2], **Adrian Weller**[1,2]

[1]University of Cambridge, UK, [2]The Alan Turing Institute, UK
dwl36@cantab.ac.uk, {usb20, aw665}@cam.ac.uk

Abstract

To interpret uncertainty estimates from differentiable probabilistic models, recent work has proposed generating a single Counterfactual Latent Uncertainty Explanation (CLUE) for a given data point where the model is uncertain, identifying a single, on-manifold change to the input such that the model becomes more certain in its prediction. We broaden the exploration to examine δ-CLUE, the set of potential CLUEs within a δ ball of the original input in latent space. We study the diversity of such sets and find that many CLUEs are redundant; as such, we propose DIVerse CLUE (∇-CLUE), a set of CLUEs which each propose a distinct explanation as to how one can decrease the uncertainty associated with an input. We then further propose GLobal AMortised CLUE (GLAM-CLUE), a distinct and novel method which learns amortised mappings on specific groups of uncertain inputs, taking them and efficiently transforming them in a single function call into inputs for which a model will be certain. Our experiments show that δ-CLUE, ∇-CLUE, and GLAM-CLUE all address shortcomings of CLUE and provide beneficial explanations of uncertainty estimates to practitioners.

Introduction

For models that provide uncertainty estimates alongside their predictions, explaining the source of this uncertainty reveals important information. For instance, determining the features responsible for predictive uncertainty can help to identify in which regions the training data is sparse, which may in turn implicate under-represented sub-groups (by age, gender, race etc). In sensitive settings, domain experts can use uncertainty explanations to appropriately direct their attention to the specific features the model finds anomalous.

In prior work, Adebayo et al. (2020) touch on the unreliability of saliency maps for uncertain inputs, and Tsirtsis, De, and Gomez-Rodriguez (2021) observe that high uncertainty can result in vast possibilities for counterfactuals. Additionally, when models are uncertain, their predictions may be incorrect. We thus consider uncertainty explanations an important precedent for model explanations; only once uncertainty has been explained can state-of-the-art methods be deployed to explain the model's prediction. However, there has been little work in explaining predictive uncertainty.

Depeweg et al. (2017) introduce decomposition of uncertainty estimates, though recent work (Antorán et al. 2021) has demonstrated further leaps, proposing to find an explanation of a model's predictive uncertainty for a given input by searching in the latent space of an auxiliary deep generative model (DGM): they identify a single possible change to the input such that the model becomes more certain in its prediction. Termed CLUE (Counterfactual Latent Uncertainty Explanations), this method aims to generate counterfactual explanations (CEs) on-manifold that reduce the uncertainty of an uncertain input $\mathbf{x}_0$. These changes are distinct from adversarial examples, which find nearby points that change the label (Goodfellow, Shlens, and Szegedy 2015).

CLUE introduces a latent variable DGM with decoder $\mu_\theta(\mathbf{x}|\mathbf{z})$ and encoder $\mu_\phi(\mathbf{z}|\mathbf{x})$. $\mathcal{H}$ refers to any differentiable uncertainty estimate of a prediction $\mathbf{y}$. The pairwise distance metric takes the form $d(\mathbf{x}, \mathbf{x}_0) = \lambda_x d_x(\mathbf{x}, \mathbf{x}_0) + \lambda_y d_y(f(\mathbf{x}), f(\mathbf{x}_0))$, where $f(\mathbf{x}) = \mathbf{y}$ is the model's mapping from an input $\mathbf{x}$ to a label, thus encouraging similarity in input space and/or prediction space. CLUE minimises:

$$\mathcal{L}(\mathbf{z}) = \mathcal{H}\left(\mathbf{y}|\mu_\theta(\mathbf{x}|\mathbf{z})\right) + d\left(\mu_\theta(\mathbf{x}|\mathbf{z}), \mathbf{x}_0\right), \qquad (1)$$

to yield $\mathbf{x}_{\text{CLUE}} = \mu_\theta(\mathbf{x}|\mathbf{z}_{\text{CLUE}})$ where $\mathbf{z}_{\text{CLUE}} = \arg\min_{\mathbf{z}} \mathcal{L}(\mathbf{z})$. There are however limitations to CLUE, including the lack of a framework to deal with a diverse set of possible explanations and the lack of computational efficiency. Although finding multiple explanations was suggested, we find the proposed technique to be incomplete.

We start by discussing the multiplicity of CLUEs. Providing practitioners with many explanations for why their input was uncertain can be helpful if, for instance, they are not in control of the recourse suggestions proposed by the algorithm; advising someone to change their age is less actionable than advising them to change a mutable characteristic (Poyiadzi et al. 2020). Specifically, we develop a method to generate a set of possible CLUEs within a δ ball of the original point in the latent space of the DGM used: we term this δ-CLUE. We then introduce metrics to measure the diversity in sets of generated CLUEs such that we can optimise directly for it: we term this ∇-CLUE. After dealing with CLUE's multiplicity issue, we consider how to make computational improvements. As such, we propose a distinct method, GLAM-CLUE (GLobal AMortised CLUE), which serves as a summary of CLUE for practitioners to audit their

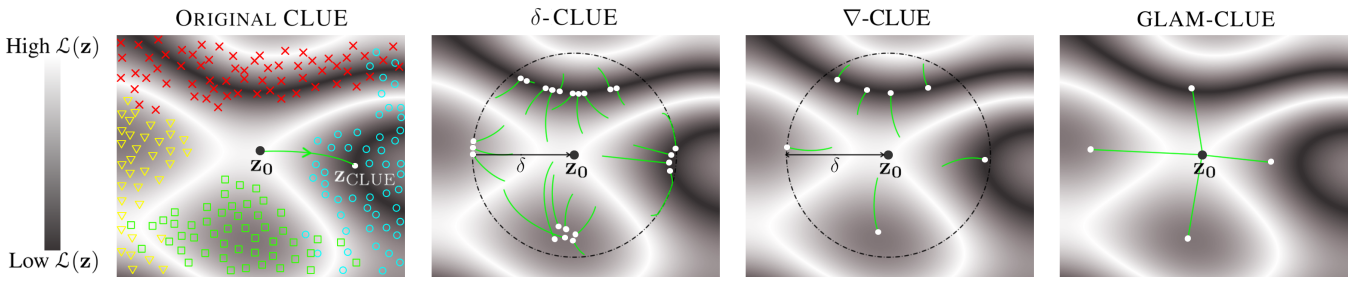


Figure 1: Conceptual colour map of objective function $\mathcal{L}(z)$ with $\mathbf{z}_0$ located in high cost region. White circles indicate explanations found. Left: Gradient descent to region of low cost (original CLUE). Training data shown in colour. Left Centre: Gradient descent constrained to δ-ball. Diverse starting points yield diverse local minima, albeit with many redundant solutions (δ-CLUE). Right Centre: Direct optimisation for diversity (∇-CLUE). Right: Efficient, unconstrained mappings without gradient descent (GLAM-CLUE), allowing computational speedups.

model's behavior on uncertain inputs. It does so by finding translations between certain and uncertain groups in a computationally efficient manner. Such efficiency is, amongst other factors, a function of the dataset, the model, and the number of CEs required; there thus exist applications where either ∇-CLUE or GLAM-CLUE is most appropriate.

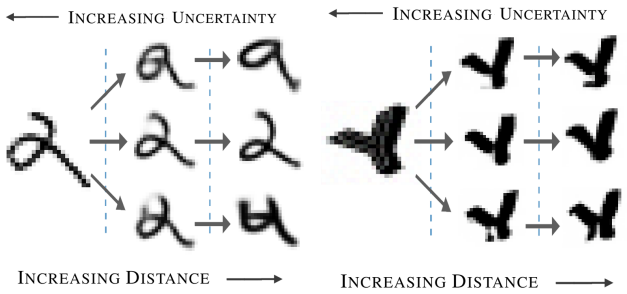
Multiplicity in Counterfactuals

Constraining CLUEs: δ-CLUE

We propose δ-CLUE (Ley, Bhatt, and Weller 2021), which generates a set of solutions that are all within a specified distance δ of $\mathbf{z}_0 = \mu_\phi(\mathbf{z}|\mathbf{x_0})$ in latent space: $\mathbf{z}_0$ is the latent representation of the uncertain input $\mathbf{x}_0$ being explained. We achieve multiplicity by initialising the search randomly in different areas of latent space. While CLUE suggests this, its random generation method and lack of constraint are prone to a) finding minima in a limited region of the space or b) straying far from this region without control over the proximity of CEs (Appendix B). Figure 1 contrasts the original and proposed objectives (left and left centre respectively).

The original CLUE objective uses VAEs (Kingma and Welling 2013) and BNNs (MacKay 1992) as the DGMs and classifiers respectively. The predictive uncertainty of the BNN is given by the entropy of the posterior over the class labels; we use the same measure. The hyperparameters (λ_x, λ_y) control the trade-off between producing low uncertainty CLUEs and CLUEs which are close to the original inputs. To encourage sparse explanations, we take $d_x(\mathbf{x}, \mathbf{x}_0) = \|\mathbf{x} - \mathbf{x}_0\|_1$. We find this to suffice for our datasets, though other metrics such as FID scores (Heusel et al. 2018) could be used in more complex vision tasks for both evaluation (as in Singla et al. (2020)) and optimisation of CEs (see Appendix B). In our proposed δ-CLUE method, the loss function matches Eq 1, with the additional δ requirement as $\mathbf{x}_{\delta-\text{CLUE}} = \mu_\theta(\mathbf{x}|\mathbf{z}_{\delta-\text{CLUE}})$ where $\mathbf{z}_{\delta-\text{CLUE}} = \arg\min_{\mathbf{z}:\ \rho(\mathbf{z},\mathbf{z}_0)\leq\delta} \mathcal{L}(\mathbf{z})$ and $\mathbf{z_0} = \mu_\phi(\mathbf{z}|\mathbf{x_0})$. We choose $\rho(\mathbf{z}, \mathbf{z}_0) = \|\mathbf{z} - \mathbf{z}_0\|_2$ (the ℓ_2 norm) in this paper, as shown in Figure 1. We first set $\lambda_x = \lambda_y = 0$ to explore solely the uncertainty landscape, given that the size of the δ-ball removes the strict need for the distance component in $\mathcal{L}(z)$ and grants control over the locality of solutions, before trialling $\lambda_x = 0.03$. We apply the δ constraint at each stage of the optimisation (Figure 1, left centre), as in Projected Gradient Descent (Boyd, Boyd, and Vandenberghe 2004).

For each uncertain input, we exploit the non-convexity of CLUE's objective to generate diverse δ-CLUEs by initialising in different regions of latent space (Figure 1). While previous work has considered sampling the latent space around an input (Pawelczyk, Broelemann, and Kasneci 2020a), we find that subsequent gradient descent yields improvements. Example results are in Figure 2. δ-CLUE is a special case of Algorithm 1, or explicitly Algorithm 3 (Appendix B).



(a) Uncertain $2 \rightarrow 9, 2, 4$.

(b) Uncertain Y $\rightarrow$ Y, V, X.

Figure 2: Visualisation of the trade-off between uncertainty $\mathcal{H}$ and distance d. Left: MNIST. Right: Symbols.

Diversity Metrics for Counterfactual Explanations

Once we have generated a set of viable CLUEs, we desire to measure the diversity within the set; as such, we require candidate convex similarity functions between points, which could be applied either pairwise or over all counterfactuals. We consider these between counterfactual labels (prediction space) or between counterfactuals themselves (input or latent space). A given diversity function D can be applied to a set of $k > 0$ counterfactuals in an appropriate space i.e. $D(\mathbf{x}_1, ..., \mathbf{x}_k)$, $D(\mathbf{z}_1, ..., \mathbf{z}_k)$ or $D(\mathbf{y}_1, ..., \mathbf{y}_k)$ where $\mathbf{x}_i \in \mathbb{R}^{d'}$, $\mathbf{z}_i \in \mathbb{R}^{m'}$ and $\mathbf{y}_i \in \mathbb{R}^{c'}$ (we define the hard prediction $y_i = \max_j(\mathbf{y}_i)_j$). Table 1 summarises the metrics.

Leveraging Determinantal Point Processes: We build on Mothilal, Sharma, and Tan (2020) to leverage determinantal point processes, referred to as DPPs (Kulesza 2012),

DIVERSITY METRIC	FUNCTION (D)
DETERMINANTAL POINT PROCESSES	$\det(\mathbf{K})$ where $\mathbf{K}_{i,j} = \frac{1}{1+d(\mathbf{x}_i, \mathbf{x}_j)}$
AVERAGE PAIRWISE DISTANCE	$\frac{1}{\binom{k}{2}} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} d(\mathbf{x}_i, \mathbf{x}_j)$
COVERAGE	$\frac{1}{d'} \sum_{i=1}^{d'} (\max_j(\mathbf{x}_j - \mathbf{x}_0)_i + \max_j(\mathbf{x}_0 - \mathbf{x}_j)_i)$
PREDICTION COVERAGE	$\frac{1}{c'} \sum_{i=1}^{c'} \max_j[(\mathbf{y}_j)_i]$
DISTINCT LABELS	$\frac{1}{c'} \sum_{j=1}^{c'} \mathbf{1}_{[\exists i \,:\, y_i = j]}$
ENTROPY OF LABELS	$-\frac{1}{\log c'} \sum_{j=1}^{c'} p_j(k) \log p_j(k)$

Table 1: Diversity metrics, D. Where necessary, we define $D = 0$ for $k = 1$ and take d to be some arbitrary distance metric.

as $\det(\mathbf{K})$ in Table 1. DPPs implicitly normalise to $0 \leq D \leq 1$. This metric is effective overall and achieves diversity by diverting attention away from the most popular (or salient) points to a diverse group of points instead. However, matrix determinants are computationally expensive for large k.

Diversity as Average Pairwise Distance: We can calculate diversity as the average distance between all distinct pairs of counterfactuals (as in Bhatt et al. (2021)). While we can adjust for the number of pairs (accomplishing invariance to k), this metric does not satisfy $0 \leq D \leq 1$, scaling instead with the pairwise distances characterised by the dataset.

Coverage as a Diversity Metric: Previous work in interpretability has leveraged the notion of coverage as a measure of the quality of sets of CEs. Ribeiro, Singh, and Guestrin (2016) define coverage to be the sum of distinct features contained in a set, weighted by feature importance: this could be applied to CEs to suggest a way of optimally choosing a subset from a full set of CEs. Plumb et al. (2020) introduce coverage as a measure of the quality of global CEs. Herein, we interpret coverage as a measure of diversity, using it directly for optimisation and evaluation of CEs. The metric, as given in Table 1, rewards changes in both positive and negative directions separately (though penalises a lack of changes in positive/negative directions). See Appendix C.

Prediction Coverage: Since rewarding negative changes in $\mathbf{y}$-space is redundant (maximising the prediction of one label implicitly minimises the others), we adjust the coverage metric in $\mathbf{y}$-space to be the maximum prediction for a particular label found in a set of CEs, averaged over all predictions. This satisfies $\frac{1}{c'} \leq D \leq 1$, where we require at least $k = c'$ CEs to achieve $D = 1$, equivalent to finding at least one fully confident prediction for each label.

Targeting Diversity of Class Labels: While recent work focuses on producing diverse explanations for binary classification problems (Russell 2019) and others summarise current methods therein (Pawelczyk, Broelemann, and Kasneci 2020b), these metrics perform well in applications rich in class labels, and conversely are likely ineffective in binary tasks. Posterior probabilities are defined as $\mathbf{y} \in \mathbb{R}^{c'}$ and $y_i = \arg\max_i \mathbf{y}_i$. We define the probability of class j as $p_j(k) = \frac{\sum_{i=1}^{k} \mathbf{1}_{[y_i = j]}}{k} = \frac{\text{number of counterfactuals in class } j}{\text{number of counterfactuals}}$. Using this, we suggest diversity through the **Number of Distinct Labels** found, as well as the **Entropy of the Label Distribution**. The former metric loses its effect once all labels are found, whereas the latter does not. The former satisfies $0 \leq D \leq 1$, and given that the maximum entropy of a c' dimensional distribution is $\log(c')$, so too does the latter.

Algorithm 1: ∇-CLUE (**simultaneous**)

Inputs: δ, k, $\mathcal{S}$, r, $\mathbf{x}_0$, d, ρ, $\mathcal{H}$, μ_θ, μ_ϕ, D, λ_D

Initialise $\varnothing$ of CLUEs: $X_{\text{CLUE}} = \{\}$;
Set δ-ball centre of $\mathbf{z}_0 = \mu_\phi(\mathbf{z}|\mathbf{x}_0)$;
for $1 \leq i \leq k$ **do**
  Set initial value of $\mathbf{z}_i = \mathcal{S}(\mathbf{z}_0, r, i, k)$;
end for
while *loss $\mathcal{L}$ has not converged* **do**
  for $1 \leq i \leq k$ **do**
    Decode: $\mathbf{x}_i = \mu_\theta(\mathbf{x}|\mathbf{z}_i)$;
    Use predictor to obtain $\mathcal{H}(\mathbf{y}|\mathbf{x}_i)$;
    $\mathcal{L}(\mathbf{z}_i) = \mathcal{H}(\mathbf{y}|\mathbf{x}_i) + d(\mathbf{x}_i, \mathbf{x}_0)$;
  end for
  $\mathcal{L}(\mathbf{z}_1, ..., \mathbf{z}_k) = -\lambda_D D(\mathbf{z}_1, ..., \mathbf{z}_k) + \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(\mathbf{z}_i)$;
  Update $\mathbf{z}_1, ..., \mathbf{z}_k$ with $\nabla_{\mathbf{z}_1, ..., \mathbf{z}_k} \mathcal{L}(\mathbf{z}_1, ..., \mathbf{z}_k)$;
  for $1 \leq i \leq k$ **do**
    Constrain $\mathbf{z}_i$ to δ ball using $\rho(\mathbf{z}_i, \mathbf{z}_0)$;
  end for
end while
for $1 \leq i \leq k$ **do**
  Decode explanation: $\mathbf{x}_i = \mu_\theta(\mathbf{x}|\mathbf{z}_i)$;
  if $\mathcal{H}(\mathbf{y}|\mathbf{x}_i) < \mathcal{H}_{\text{threshold}}$ **then**
    $X_{\text{CLUE}} \leftarrow X_{\text{CLUE}} \cup \mathbf{x}_i$;
  end if
end for

Outputs: X_{CLUE}, a set of $n \leq k$ diverse CLUEs

Optimizing for Diversity: ∇-CLUE

The diversity metrics defined in Table 1 find utility in the optimisation of a set of k counterfactuals. We optimise for diversity in the CLUEs we generate through an explicit diversity term in our objective for the CLUEs found. We call this DIVerse CLUE or ∇-CLUE. We posit that whilst some aforementioned metrics may perform poorly during optimisation, we retain them for evaluation.

Once the diversity metric is selected, the optimisation of k counterfactuals can be performed **simultaneously** (Algorithm 1) in latent space (Mothilal, Sharma, and Tan 2020), or **sequentially** (Appendix D), where the approach is analogous to a greedy algorithm of the former approach. The notation $X_{\text{CLUE}} = \{\mathbf{x}_1, ..., \mathbf{x}_k\}$ is adopted to represent a set of k counterfactuals (similarly Z_{CLUE} and Y_{CLUE}).

We denote an initialisation scheme $\mathcal{S}$ of radius r to generate starting points for the gradient descent. Note that the removal of the δ constraint or the initialisation may be

achieved at $\delta = \infty$ and $r = 0$ respectively (although the latter yields the same counterfactual k times as a result of symmetry). Thus, **the ∇-CLUE algorithm is equivalent to δ-CLUE when $\lambda_D = 0$**, which is itself equivalent to the original CLUE algorithm when $\delta = \infty$, $r = 0$ and $k = 1$. Example results are in Figure 3.

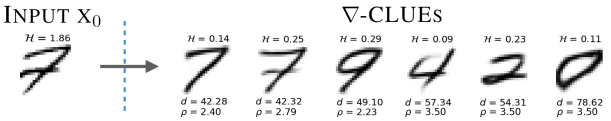


Figure 3: We produce a **diverse set** of candidate explanations that show how to reduce predictive uncertainty while remaining close to x_0 in both input and latent space ($\mathcal{H}$ is uncertainty, d is input distance, ρ is latent distance). We see that the left image might most easily be resolved into a confident 7 or 9. Results are taken from a larger set of ∇-CLUEs and are not exemplary of setting $k = 5$.

Simultaneous Diversity Optimisation (Algorithm 1): By optimising simultaneously over k counterfactuals in latent space, issues with how the diversity metric D might scale with k can be avoided. We have the simultaneous optimisation problem of minimising $\mathcal{L}(\mathbf{z}_1, ..., \mathbf{z}_k) = -\lambda_D D(\mathbf{z}_1, ..., \mathbf{z}_k) + \frac{1}{k}\sum_{i=1}^{k} \mathcal{L}(\mathbf{z}_i)$ where $\mathcal{L}(\mathbf{z}_i) = \mathcal{H}(\mathbf{y}|\mu_\theta(\mathbf{x}|\mathbf{z}_i)) + d(\mu_\theta(\mathbf{x}|\mathbf{z}_i), \mathbf{x}_0)$, to yield $X_{\text{CLUE}} = \mu_\theta(X|Z_{\text{CLUE}})$ where $Z_{\text{CLUE}} = \arg\min_{\mathbf{z}_1,...,\mathbf{z}_k} = \mathcal{L}(\mathbf{z}_1, ..., \mathbf{z}_k)$. Note that we apply the diversity function in latent space; it could equally be applied in input space.

Sequential Diversity Optimisation (Appendix D): Given a set of counterfactuals Z_{CLUE} (initially the empty set $\varnothing$), we can apply ∇-CLUE sequentially, appending each new counterfactual to the set. At each iteration, we minimise $\mathcal{L}(\mathbf{z}) = \lambda_D D(Z_{\text{CLUE}} \cup \mathbf{z}) + \mathcal{H}(\mathbf{y}|\mu_\theta(\mathbf{x}|\mathbf{z})) + d(\mu_\theta(\mathbf{x}|\mathbf{z}), \mathbf{x}_0)$ to yield $\mathbf{z}_{\text{CLUE}}$ which we append to the set.

Global and Amortised Counterfactuals

CLUE primarily focuses on local explanations of uncertainty estimates, as Antorán et al. (2021) propose a method for finding a single, small change to an uncertain input that takes it from uncertain to certain with respect to a classifier. Such local explanations can be computationally expensive to apply to large sets of inputs. Large sets of counterfactuals are also difficult to interpret. We thus face challenges when using them to summarise global uncertainty behaviour, which is important in identifying areas in which the model does not perform as expected or the training data is sparse.

We desire a computationally efficient method that requires a finite portion of the dataset (or a finite set of CEs) from which global properties of uncertainty can be learnt and applied to unseen test data with high reliability. We propose GLAM-CLUE (GLobal AMortised CLUE), which achieves such reliability with considerable speedups.

Proposed Method: GLAM-CLUE

GLAM-CLUE takes groups of high/low certainty points and learns mappings of arbitrary complexity between them in latent space (**training step**). Mappers are then applied to generate CEs from uncertain inputs (**inference step**). It can be seen as a global equivalent to CLUE. Initially, inputs are taken from the training data to learn such mappings, but we demonstrate that we can make improvements by instead using CLUEs generated from uncertain points in the training data. Algorithm 2 defines a mapper of arbitrary complexity from uncertain groups to certain groups in latent space: $\mathbf{z}_{\text{certain}} = G(\mathbf{z}_{\text{uncertain}})$. These mappers have parameters $\boldsymbol{\theta}$.

Algorithm 2: GLAM-CLUE (Training Step)

Inputs: Inputs $X_{\text{uncertain}}$, X_{certain}, groups $Y_{\text{uncertain}}$, Y_{certain}, DGM encoder μ_ϕ, loss $\mathcal{L}$, trainable parameters $\boldsymbol{\theta}$

```
for all groups (i → j) in (Y_uncertain, Y_certain) do
    Select X_i from X_uncertain, Y_uncertain;
    Select X_j from X_certain, Y_certain;
    Encode: Z_i = μ_φ(Z|X_i);
    while loss ℒ has not converged do
        Update θ_{i→j} with ∇_{θ_{i→j}} ℒ(θ_{i→j}|Z_i, X_j);
    end while
end for
```

Outputs: A collection of mapping parameters $\boldsymbol{\theta}_{i\to j}$ for given mappers $G_{i\to j}$ that take uncertain inputs from group i and produce nearby certain outputs in group j

To strive for global explanations, we restrict each mapper in our experiments to be a single latent translation from an uncertain class i to a certain class j: $\mathbf{z}_j = G_{i\to j}(\mathbf{z}_i) = \mathbf{z}_i + \boldsymbol{\theta}_{i\to j}$. When run on test data, mappers should reduce the uncertainty of points while keeping them close to the original. To train the parameters of the translation $\boldsymbol{\theta}$, we use the loss function detailed in Equation 2, similar to Van Looveren and Klaise (2021), who inspect the k nearest data points (our min operation implies $k = 1$). We infer from Figure 7, right, that regularisation in latent space implies regularisation in input space. We learn separate mappers for each pair of groups defined by the practitioner (Figure 6); Algorithm 2 loops over these groups, partitioning the data accordingly, and returning distinct parameters $\boldsymbol{\theta}_{i\to j}$ for each case.

$$\mathcal{L}(\boldsymbol{\theta}|Z_{\text{uncertain}}, X_{\text{certain}}) =$$

$$\lambda_\theta \|\boldsymbol{\theta}\|_1 + \frac{1}{|Z_{\text{uncertain}}|} \sum_{\mathbf{z}\in Z_{\text{uncertain}}} \min_{\mathbf{x}\in X_{\text{certain}}} \|\mu_\theta(\mathbf{z}+\boldsymbol{\theta}) - \mathbf{x}\|_2^2 \quad (2)$$

Few works in the counterfactual literature address uncertainty explanations; we avoid the comparison with state-of-the-art counterfactual methods for the reasons discussed in the introduction. However, there exist multiple standard baselines against which we can test performance. Firstly, we can perform Difference Between Means (DBM) of uncertain data to certain data in either input or latent space. This can be added to uncertain test data and reconstructed in the case of input space, or decoded in the case of latent space. Another baseline is the Nearest Neighbours (NN) in high certainty training data, in either input or latent space. Figure 5 visualises these baselines in latent space. Our experiments demonstrate that GLAM-CLUE outperforms these baselines significantly, and performs on par with CLUE. Pawelczyk et al. (2021) create a benchmarking tool which shows that CLUE performs on par with the current state-of-the-art. By extension, so too does our scheme, but 200 times faster.

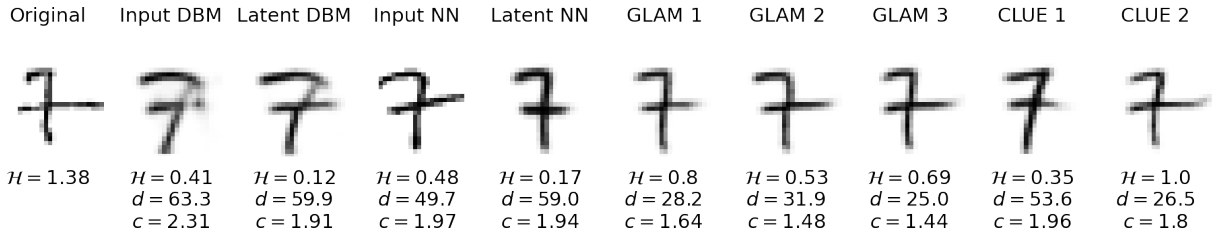


Figure 4: Comparison of the explanations generated for an uncertain input (far left) by the baselines, GLAM-CLUE, and CLUE. $\mathcal{H}$ is uncertainty, d is input distance, $c = \mathcal{H} + \lambda_x d$ is cost. Low uncertainties in some baseline schemes are invalidated by unrealistic distances. GLAM 1/2/3 are described in the Experiments/GLAM-CLUE section. CLUE 1/2 are generated from $\lambda_x = 0$ and $\lambda_x = 0.03$ respectively.

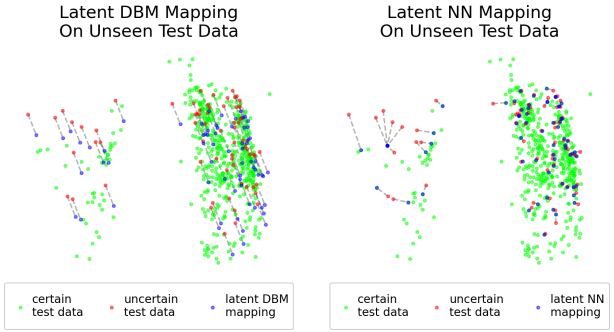


Figure 5: Visualisation of the DBM and NN baselines for MNIST's digit 4 in a 2D latent space. Left: Uncertain points in the test data with their respective latent DBM mappings. Right: Uncertain points in the test data with their respective NN mappings. High certainty training data is shown in green throughout.

When the class of uncertain test data is unknown, mappings could be applied over each combination of classes, picking the best performing CEs. When the number of classes is large, a scheme to select a limited number of these (e.g. the top n predictions from the classifier) could be used.Generic mappings from uncertainty to certainty would not require this selection but on the whole would be harder to train (simple translations are likely invalid for the far right case of Figure 6). We posit that more complex models such as neural networks could improve the performance of mappings at the risk of losing the global sense of the explanation.

Grouping Uncertainty

Most counterfactual explanation techniques center around determining ways to change the class label of a prediction; for example, Transitive Global Translations (TGTs) consider each possible combination of classes and the mappings between them (Plumb et al. 2020). We choose here to partition the data into classes, but also into certain and uncertain groups according to the classifier used. By using these partitions, we learn mappings from uncertain points to certain points, either within specific classes or in the general case. While TGTs constrain a mapping G from group i to j to be symmetric ($G_{i\to j} = G^{-1}_{j\to i}$) and transitive ($G_{i\to k} = G_{j\to k} \circ G_{i\to j}$), we see no direct need for the symmetry constraint. There exists an infinitely large domain of uncertain points, unlike the bounded domain for certain points, implying a many-to-one mapping. We also forgo the transitivity constraint: defining direct mappings from uncertain points to specific certain points is sufficient.

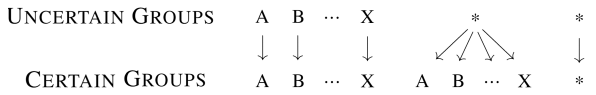


Figure 6: Example mappings from uncertainty to certainty in groups A to X, without necessarily satisfying symmetry or transitivity. Asterisks represent members belonging to any group.

Our method is general to all schemes (and more) in Figure 6. Our experiments consider these groups to be class labels, testing against the far left scheme which considers mapping from uncertain points to certain points within a given class. Future work may consider modes within classes, as well as the more general far right scheme of learning mappings from arbitrary uncertain inputs to their certain analogues. The original CLUE method is analogous to the far right scheme, which is agnostic to the particular classes it maps to and from (although struggles with diverse mappings).

Experiments

We perform experiments on 3 datasets to validate our methods: UCI Credit classification (Dua and Graff 2017), MNIST image classification (LeCun 1998) and Synbols image classification (Lacoste et al. 2020). On Credit and MNIST, we train VAEs as our DGMs (Kingma and Welling 2013) and BNNs for classification (MacKay 1992). For Synbols, we train Hierarchical VAEs (Zhao, Song, and Ermon 2017) and a resnet deep ensemble, owing to higher dataset complexity (rotations, sizes and obscurity of shapes). We demonstrate that our constraints allow practitioners to better control the uncertainty-distance trade-off of CEs (δ-CLUE) and the diversity of CEs (∇-CLUE). We then show that we can efficiently generate explanations that apply globally to groups of inputs with our amortised scheme (GLAM-CLUE).

δ-CLUE

We learn from the δ-CLUE experiments that the δ value controls the trade-off between the uncertainty of the CLUEs generated and their distance from the original point (Figure 2). Importantly, by tuning λ_x in the distance term d of Equation 1, we achieve lower distances with only small uncertainty increases (Figure 7, right). We observe further in

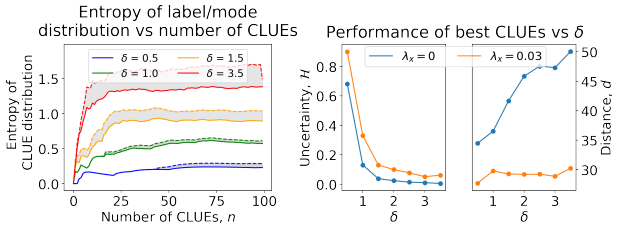


Figure 7: Left: Diversity analysis in MNIST. Entropy of the distribution of class labels (solid) and modes (dashed) found as number of CLUEs increases. Labels vary from 0 to 9 whilst there exist multiple modes within each label. Observe the entropy saturating as we converge to all minima within the δ ball. Right: Performance of δ-CLUEs (uncertainty, $\mathcal{H}$, input space distance, d). Batch size: 8 most uncertain MNIST digits. Learning rate: 0.1. Iterations: 30.

Figure 7, left that diversity increases with δ, although a large number of CLUEs can be required before such levels become saturated (left). Modes are defined as groups of points within specific classes. Full analysis in Appendix B.

Takeaway: δ-CLUE produces a high performing set of diverse explanations. However, we require many iterations to achieve such diversity (∇-CLUE addresses this).

∇-CLUE

We perform an ablative study, increasing the diversity weight λ_D and optimising the DPP diversity metric in **z**-space, measuring the effect that this has on each other metric. We use the simultaneous ∇-CLUE scheme in Algorithm 1 for a fixed number of $k = 10$ CLUEs and parameters: $\delta = r = 4$ for MNIST; $\delta = r = 1$ for UCI Credit. The optimal δ value(s) can be determined through experimentation (Figure 7, right), although Appendix B discusses alternative methods such as inspecting nearest neighbours in the data.

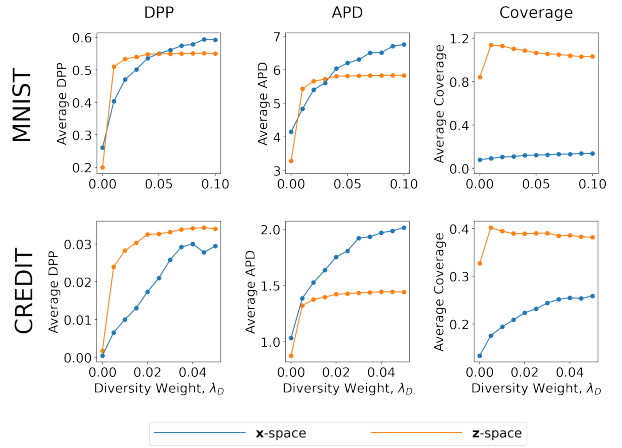


Figure 8: Effect of λ_D on diversity. Row 1: MNIST. Row 2: UCI Credit. Columns 1 to 3: DPP, APD and Coverage diversity metrics applied to the set of $k = 10$ ∇-CLUEs. $\lambda_D = 0$ is δ-CLUE. Batch size: 8 most uncertain inputs. Learning rate: 0.1. Iterations: 30.

Takeaway: When optimising for one diversity metric, increasing λ_D monotonically improves diversity by almost every other metric. Average uncertainty suffers only a small amount relative to the gains we achieve in diversity and ∇-CLUE requires fewer counterfactuals to achieve the same level of diversity as δ-CLUE.

GLAM-CLUE

Gradient descent at the inference step (generation of CEs) is computationally expensive. Uncertainty estimates, distance metrics, and diversity metrics (notably DPPs, which operate on $k \times k$ matrices) all require evaluation over many iterations, to yield only a single counterfactual to a local uncertain input. While local explanations have utility in certain settings, GLAM-CLUE computes CEs for all uncertain test points in a single, amortised function call, permitting considerable speedups. We demonstrate that the performance of these counterfactual beats mean performance of all the baselines discussed, achieving lower variance also.

We train 3 mappers: GLAM 1 learns from all certain and uncertain 4s in the MNIST training data; GLAM 2/3 learn from all uncertain 4s in the training data and their corresponding certain CLUEs, for $\lambda_x = 0$ and $\lambda_x = 0.03$ respectively. Figure 9 shows improvements when using GLAM 2 and 3, demonstrating that CLUEs capture properties of uncertainty more reliably than the training data, at the expense of extra computation time to generate the CLUEs used.

We observe that while the baseline schemes achieve low uncertainties, they do so at the expense of moving much further away from the input (Figure 4), implying infeasible actionability. An advantage to GLAM-CLUE is that the uncertainty-distance trade-off can be tuned with λ_θ in Equation 2: larger λ_θ restricts translations in latent space, thus lowering distances in input space but raising uncertainties. For a given λ_x, GLAM-CLUE's fast learning rate allows for the optimal λ_θ to be determined quickly. Furthermore, 98% of uncertain 4 to certain 4 GLAM-CLUE mappings resulted in a classification of 4 (87% for CLUE which simply minimises uncertainty and is not class specific).

Takeaway: Amortisation of counterfactuals works. A simple global translation for class specific points is shown to produce counterfactuals of comparable quality to CLUE. Notably, performance of GLAM-CLUE is improved when training on CLUEs rather than training data, optimal when we generate CLUEs using $\lambda_x = 0.03$, as used in evaluation.

Computational Speedup

At the inference step, GLAM-CLUE performs significantly faster than CLUE in terms of **average CPU time**, detailed in Table 2. For uncertain 4s in the MNIST test set, CLUE required on average 220 seconds to converge; GLAM-CLUE took around 1 second to compute. The bottleneck in these

Input DBM	Latent DBM	Input NN
0.0306	0.0262	0.0236
Latent NN	**GLAM-CLUE**	**CLUE**
0.0245	0.0238	4.68

Table 2: Avg. time in seconds for 1 MNIST CE (inference step). We achieve similar speedups (186 times faster) for UCI Credit.

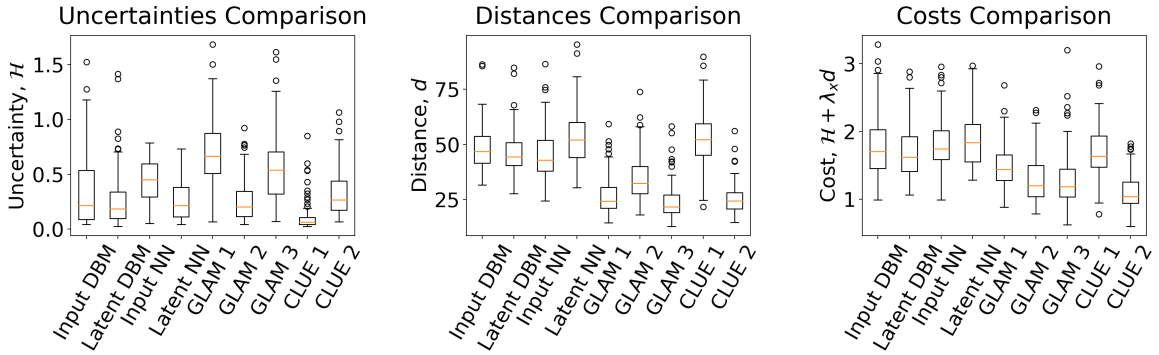


Figure 9: GLAM-CLUE schemes vs baselines when mapping uncertain 4s to certain 4s in MNIST. Left: Distributions of uncertainties, $\mathcal{H}$ (original uncertainties exceed 1.5). Centre: Distributions of input distances, d. Right: Distributions of total costs, $\mathcal{H} + \lambda_x d$, with $\lambda_x = 0.03$ as used by Antorán et al. (2021). Similar results for all classes (Appendix E). CLUE 1/CLUE 2 are generated from $\lambda_x = 0$ and $\lambda_x = 0.03$ respectively. Batch size: 6000 (all 4s in training set. Learning rate: 0.1. Iterations: 30. Multiple random seed runs yield negligible differences.

processes is the uncertainty evaluation of the BNN, and as such these timings are not necessarily representative of all models. A drawback to GLAM-CLUE is that the optimisation required on average 17.6 seconds to train. Should CLUEs be included during training (i.e. GLAM 2 and 3), extra time is required to obtain these. Moving beyond basic mappers to more advanced models, we expect performance to improve at the cost of an increased training step time.

Takeaway: GLAM-CLUE produces explanations around 200 times faster than CLUE. This speedup, alongside the baselines, means that we have the option to take the best performing counterfactual out of GLAM-CLUE and the baselines, without requiring significant computation.

Related and Future Work

The majority of this paper is dedicated to increasing the practical utility of the uncertainty explanations proposed as CLUE in Antorán et al. (2021), and we mitigate CLUE's multiplicity and efficiency issues. Very few works address explaining the uncertainty of probabilistic models. Booth et al. (2020) take a user-specified level of uncertainty for a sample in an auxiliary discriminative model and generate the corresponding sampling using deep generative models (DGM). Joshi et al. (2018) propose xGEMs that use a DGM to find CEs (as we do) though not for uncertainty. Mothilal, Sharma, and Tan (2020) and Russell (2019) use linear programs to find a diverse set of CEs, though also not for uncertainty. Neither paper considers computational advances nor ventures to consider global CEs, as we do. Plumb et al. (2020) define a mapper that transforms points from one low-dimensional group to another. Mahajan, Tan, and Sharma (2020) and Yang et al. (2021) redesign DGMs to generate CEs quickly, similar to GLAM-CLUE. In spirit of such works, we propose amortising CLUE to find a transformation that leads the model to treat the transformed uncertain points from Group A as certain points from Group B. This method could extend beyond CLUE to other classes of CEs.

Future explorations include higher dimensional datasets such as CIFAR10 (Krizhevsky 2012) and CelebA (Liu et al. 2015) that would fully test CLUE and the extensions proposed in this paper, potentially requiring the use of FID scores (Heusel et al. 2018) to replace the simple distance metric in both evaluation (Singla et al. 2020) and optimisation. DGM alternatives such as GANs (Goodfellow et al. 2014) could be explored therein. Further, since Antorán et al. (2021) demonstrate success on human subjects in the use of DGMs for counterfactuals, our reasoning is that we can hope to retain this efficacy with our extensions of CLUE, though ideally additional human experiments would further validate our methods. Multiple runs at various random seeds would also shed light on the sensitivity of the ∇-CLUE algorithm.

Conclusion

Explanations from machine learning systems are receiving increasing attention from practitioners and industry (Bhatt et al. 2020). As these systems are deployed in high stakes settings, well-calibrated uncertainty estimates are in high demand (Spiegelhalter 2017). For a method to interpret uncertainty estimates from differentiable probabilistic models, Antorán et al. (2021) propose generating a Counterfactual Latent Uncertainty Explanation (CLUE) for a given data point on which the model is uncertain. In this work, we examine how to make CLUEs more useful in practice. We devise δ-CLUE, a method to generate a set of potential CLUEs within a δ ball of the original input in latent space, before proposing DIVerse CLUE (∇-CLUE), a method to find a set of CLUEs in which each proposes a distinct explanation for how to decrease the uncertainty associated with an input (to tackle the redundancy within δ-CLUE). However, these methods prove to be potentially computationally inefficient for large amounts of data. To that end, we propose GLobal AMortised CLUE (GLAM-CLUE), which learns an amortised mapping that applies to specific groups of uncertain inputs. GLAM-CLUE efficiently transforms an uncertain input in a single function call into an input that a model will be certain about. We validate our methods with experiments, which show that δ-CLUE, ∇-CLUE, and GLAM-CLUE address shortcomings of CLUE. We hope our proposed methods prove beneficial to practitioners who seek to provide explanations of uncertainty estimates to stakeholders.

Acknowledgments

UB acknowledges support from DeepMind and the Leverhulme Trust via the Leverhulme Centre for the Future of Intelligence (CFI) and from the Mozilla Foundation. AW acknowledges support from a Turing AI Fellowship under grant EP/V025379/1, The Alan Turing Institute under EPSRC grant EP/N510129/1 and TU/B/000074, and the Leverhulme Trust via CFI. The authors thank Javier Antorán for his helpful comments and pointers.

References

Adebayo, J.; Muelly, M.; Liccardi, I.; and Kim, B. 2020. Debugging Tests for Model Explanations. In *Advances in Neural Information Processing Systems*.

Antorán, J.; Bhatt, U.; Adel, T.; Weller, A.; and Hernández-Lobato, J. M. 2021. Getting a CLUE: A Method for Explaining Uncertainty Estimates. In *International Conference on Learning Representations*.

Bhatt, U.; Chien, I.; Zafar, M. B.; and Weller, A. 2021. DIVINE: Diverse Influential Training Points for Data Visualization and Model Refinement. arXiv:2107.05978.

Bhatt, U.; Xiang, A.; Sharma, S.; Weller, A.; Taly, A.; Jia, Y.; Ghosh, J.; Puri, R.; Moura, J. M.; and Eckersley, P. 2020. Explainable Machine Learning in Deployment. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 648–657.

Booth, S.; Zhou, Y.; Shah, A.; and Shah, J. 2020. BayesTrEx: Model Transparency by Example. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*.

Boyd, S.; Boyd, S. P.; and Vandenberghe, L. 2004. *Convex Optimization*. Cambridge university press.

Depeweg, S.; Hernández-Lobato, J. M.; Doshi-Velez, F.; and Udluft, S. 2017. Uncertainty Decomposition in Bayesian Neural Networks with Latent Variables. arXiv:1706.08495.

Dosovitskiy, A.; and Djolonga, J. 2020. You Only Train Once: Loss-Conditional Training of Deep Networks. In *International Conference on Learning Representations*.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative Adversarial Networks. In *Advances in Neural Information Processing Systems*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations*.

Grover, D.; and Toghi, B. 2019. MNIST dataset classification utilizing k-NN classifier with modified sliding-window metric. In *Science and Information Conference*, 583–591. Springer.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2018. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Advances in Neural Information Processing Systems*.

Joshi, S.; Koyejo, O.; Kim, B.; and Ghosh, J. 2018. xGEMs: Generating Examplars to Explain Black-Box Models. *arXiv preprint arXiv:1806.08867*.

Kingma, D. P.; and Welling, M. 2013. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*.

Krizhevsky, A. 2012. Learning Multiple Layers of Features from Tiny Images. *University of Toronto*.

Kulesza, A. 2012. Determinantal Point Processes for Machine Learning. *Foundations and Trends® in Machine Learning*, 5(2-3): 123–286.

Lacoste, A.; Rodríguez, P.; Branchaud-Charron, F.; Atighehchian, P.; Caccia, M.; H. Laradji, I.; Drouin, A.; Craddock, M.; Charlin, L.; and Vázquez, D. 2020. Synbols: Probing Learning Algorithms with Synthetic Datasets. In *Advances in Neural Information Processing Systems*.

LeCun, Y. 1998. The MNIST Database of Handwritten Digits. *http://yann. lecun. com/exdb/mnist/*.

Ley, D.; Bhatt, U.; and Weller, A. 2021. δ-CLUE: Diverse Sets of Explanations for Uncertainty Estimates. In *ICLR Workshop on Security and Safety in Machine Learning Systems*.

Liu, Z.; Luo, P.; Wang, X.; and Tang, X. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

MacKay, D. J. 1992. A Practical Bayesian Framework for Backpropagation Networks. *Neural computation*, 4(3): 448–472.

Mahajan, D.; Tan, C.; and Sharma, A. 2020. Preserving Causal Constraints in Counterfactual Explanations for Machine Learning Classifiers. In *NeurIPS Workshop on CausalML: Machine Learning and Causal Inference for Improved Decision Making*.

Mothilal, R. K.; Sharma, A.; and Tan, C. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*.

Pawelczyk, M.; Bielawski, S.; van den Heuvel, J.; Richter, T.; and Kasneci, G. 2021. CARLA: A Python Library to Benchmark Algorithmic Recourse and Counterfactual Explanation Algorithms. In *Advances in Neural Information Processing Systems (Benchmark & Data Set Track)*.

Pawelczyk, M.; Broelemann, K.; and Kasneci, G. 2020a. Learning Model-Agnostic Counterfactual Explanations for Tabular Data. In *Proceedings of The Web Conference 2020*, 3126–3132.

Pawelczyk, M.; Broelemann, K.; and Kasneci, G. 2020b. On Counterfactual Explanations under Predictive Multiplicity. In *Conference on Uncertainty in Artificial Intelligence*, 809–818. PMLR.

Plumb, G.; Terhorst, J.; Sankararaman, S.; and Talwalkar, A. 2020. Explaining Groups of Points in Low-Dimensional Representations. In *International Conference on Machine Learning*, 7762–7771. PMLR.

Poyiadzi, R.; Sokol, K.; Santos-Rodriguez, R.; De Bie, T.; and Flach, P. 2020. FACE: Feasible and Actionable Counterfactual Explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 344–350.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. Why Should I Trust You?: Explaining the Predictions of any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. ACM.

Russell, C. 2019. Efficient Search for Diverse Coherent Explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 20–28.

Singla, S.; Pollack, B.; Chen, J.; and Batmanghelich, K. 2020. Explanation by Progressive Exaggeration. In *International Conference on Learning Representations*.

Spiegelhalter, D. 2017. Risk and Uncertainty Communication. *Annual Review of Statistics and Its Application*, 4: 31–60.

Tsirtsis, S.; De, A.; and Gomez-Rodriguez, M. 2021. Counterfactual Explanations in Sequential Decision Making Under Uncertainty. In *Advances in Neural Information Processing Systems*.

Van Looveren, A.; and Klaise, J. 2021. Interpretable Counterfactual Explanations Guided by Prototypes. In *Machine Learning and Knowledge Discovery in Databases. Research Track*, 650–665.

Weinberger, K. Q.; and Saul, L. K. 2009. Distance metric learning for large margin nearest neighbor classification. *Journal of machine learning research*, 10(2).

Yang, F.; Alva, S. S.; Chen, J.; and Hu, X. 2021. Model-Based Counterfactual Synthesizer for Interpretation. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*.

Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 586–595.

Zhao, S.; Song, J.; and Ermon, S. 2017. Learning Hierarchical Features from Deep Generative Models. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 4091–4099. PMLR.

Appendix

This appendix is formatted as follows.

1. We discuss **datasets and models** in Appendix A.
2. We provide a full analysis of the **δ-CLUE** experiments and design choices in Appendix B.
3. We discuss **diversity metrics for counterfactual explanations** further in Appendix C.
4. We analyse **∇-CLUE** in Appendix D.
5. We discuss **GLAM-CLUE** in Appendix E.

Where necessary, we provide a discussion for potential limitations of our work and future improvements that could be studied.

A Datasets and Models

One tabular dataset and two image datasets are employed in our experiments (all publicly available). Details are provided in Table 3.

The default of credit card clients dataset, which we refer to as "Credit" in this paper, can be obtained from https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients/. We augment input dimensions by performing a one-hot-encoding over necessary variables (i.e. gender, education). Note that this dataset is different from the also common German credit dataset.

The MNIST handwritten digit image dataset can be obtained from and is described in detail at http://yann.lecun.com/exdb/mnist/. For the aforementioned datasets, we thank Antorán et al. (2021) for making their private BNN and VAE models available for use in our work.

For the experiments on Synbols, we use the black and white dataset, as provided by ElementAI at https://github.com/ElementAI/synbols/. Additional models (resnet classifiers, hierarchical VAEs) and loading scripts are taken from https://github.com/ElementAI/synbols-benchmarks/.

B δ-CLUE

We perform constrained optimisation during gradient descent (Figure 1, centre). A later part of this Appendix (Constrained vs Unconstrained Search) provides justification for this decision. In our experiments, we search the latent space of a VAE to generate δ-CLUEs for the 8 most uncertain digits in the MNIST test set, according to our trained BNN.

We trial this over **a)** a range of several δ values from 0.5 to 3.5, **b)** two latent space loss functions: **Uncertainty** $\mathcal{L}_{\mathcal{H}} = \mathcal{H}$ and **Distance** $\mathcal{L}_{\mathcal{H}+d} = \mathcal{H} + d$ and **c)** two initialisation schemes as depicted in Figure 10. Initialisation scheme $\mathcal{S}_1$ picks a random direction at a uniform random radius within the delta ball, while the other scheme $\mathcal{S}_2$ is along paths determined by the nearest neighbours (**NN**) for each class in the training data. We label these experiment variants as: **Uncertainty Random**: [$\mathcal{L}_{\mathcal{H}}$, $\mathcal{S}_1$], **Uncertainty NN**: [$\mathcal{L}_{\mathcal{H}}$, $\mathcal{S}_2$], **Distance Random**: [$\mathcal{L}_{\mathcal{H}+d}$, $\mathcal{S}_1$] and **Distance NN**: [$\mathcal{L}_{\mathcal{H}+d}$, $\mathcal{S}_2$].

In Figure 11, the $\mathcal{L}_{\mathcal{H}}$ experiments (blue and orange) demonstrate how the best CLUEs found improve as the δ ball expands, at the cost of increased distance from the original input. The $\mathcal{L}_{\mathcal{H}+d}$ experiments (green and red) suggest that the $\mathcal{L}_{\mathcal{H}+d}$ objective can vastly improve performance when it comes to distance (right), at the expense of higher (but acceptable) uncertainty.

Takeaway 1: as δ increases, using either loss $\mathcal{L}_{\mathcal{H}}$ or $\mathcal{L}_{\mathcal{H}+d}$, we reduce the uncertainty of our CLUEs at the expense of greater distance d. Loss $\mathcal{L}_{\mathcal{H}+d}$ experiences larger performance gains in the distance curves (green and red, Figure 11, right).

We demonstrate that δ-CLUEs are successful in converging sufficiently to all local minima within the ball, given large enough n (Figure 15, left). Additionally, as the size of the δ ball increases, the random generation scheme $\mathcal{S}_1$ used in experiments **Uncertainty Random** and **Distance Random** converge to the highest numbers of diverse CLUEs

Name	Targets	Input Type	Input Dimension	No. Train	No. Test
Credit	Binary	Continuous & Categorical	24	27000	3000
MNIST	Categorical	Image (Greyscale)	28×28	60000	10000
Synbols	Categorical	Image (RGB)	$3 \times 32 \times 32$	60000	20000

Table 3: Summary of the datasets used in our experiments.

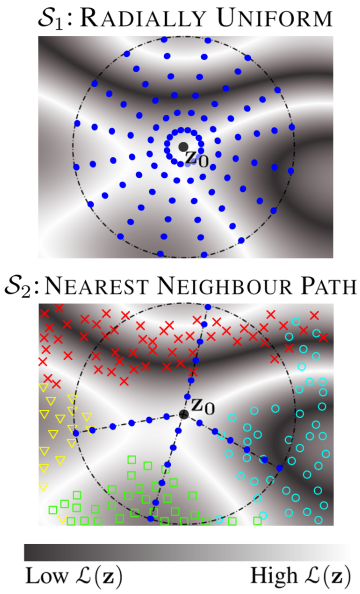


Figure 10: Two possible initialisation schemes $\mathcal{S}_i$ to yield diverse minima. One is random, the other deterministic. Details are provided in this Appendix.

Algorithm 3: δ-CLUE

Inputs: δ, k, $\mathcal{S}$, r, $\mathbf{x}_0$, d, ρ, $\mathcal{H}$, μ_θ, μ_ϕ

Initialise $\varnothing$ of CLUEs: $X_{\text{CLUE}} = \{\}$;
Set δ-ball centre of $\mathbf{z}_0 = \mu_\phi(\mathbf{z}|\mathbf{x}_0)$;
for $1 \leq i \leq k$ **do**
  Set initial value of $\mathbf{z}_i = \mathcal{S}(\mathbf{z}_0, r, i, k)$;
  while *loss $\mathcal{L}$ has not converged* **do**
    Decode: $\mathbf{x} = \mu_\theta(\mathbf{x}|\mathbf{z}_i)$;
    Use predictor to obtain $\mathcal{H}(\mathbf{y}|\mathbf{x})$;
    $\mathcal{L} = \mathcal{H}(\mathbf{y}|\mathbf{x}) + d(\mathbf{x}, \mathbf{x}_0)$;
    Update $\mathbf{z}_i$ with $\nabla_{\mathbf{z}}\mathcal{L}$;
    if $\rho(\mathbf{z}_i, \mathbf{z}_0) > \delta$ **then**
      Project $\mathbf{z}_i$ onto the surface of the δ-ball as $\mathbf{z}_i = \delta \times \frac{\mathbf{z}_i - \mathbf{z}_0}{\rho(\mathbf{z}_i, \mathbf{z}_0)}$;
    end if
  end while
  Decode explanation: $\mathbf{x}_{\delta-\text{CLUE}} = \mu_\theta(\mathbf{x}|\mathbf{z}_i)$;
  if $\mathcal{H}(\mathbf{y}|\mathbf{x}_{\delta_{\text{CLUE}}}) < \mathcal{H}_{\text{threshold}}$ **then**
    $X_{\text{CLUE}} \leftarrow X_{\text{CLUE}} \cup \mathbf{x}_{\delta_{\text{CLUE}}}$;
  end if
end for

Outputs: X_{CLUE}, a set of $n \leq k$ CLUEs

(Figure 15, right, blue and green). In both loss function landscapes ($\mathcal{L}_{\mathcal{H}}$ and $\mathcal{L}_{\mathcal{H}+d}$), we obtain similarly high levels of diversity as δ increases.

Takeaway 2: we can achieve a diverse plethora of high quality CLUEs when it comes to both class labels and modes of change within classes, permitting a full summary of uncertainty.

Given a diverse set of proposed δ-CLUEs (Figure 12), the performances of each class can be ranked by choosing an appropriate δ value and loss $\mathcal{L}$ for the mentioned trade-offs (see the last subsection of this Appendix). Here, the 2 achieves lower uncertainty for a given distance, whilst the 9 and 4 require higher distances to achieve the same uncertainty. Without a δ constraint, we can move far from the original input and obtain a CLUE from any class that is certain to the BNN.

Takeaway 3: we can produce a **label distribution** over the δ-CLUEs to better summarise the diverse changes that could be made to reduce uncertainty.

Gradient Descent vs Sampling

Figures 13 and 14 demonstrate the superiority of performing gradient descent over sampling in terms of the quality of counterfactuals found. Sampling does have the advantage of being computationally faster, and so more advanced search methods in future work could utilise sampling before performing gradient descent to achieve optimal performance with faster computation.

Distance Metrics

In this section, we take $d_x(\mathbf{x}, \mathbf{x}_0) = \|\mathbf{x} - \mathbf{x}_0\|_1$ to encourage sparse explanations. The original CLUE paper found that for regression, $d_y(f(\mathbf{x}), f(\mathbf{x}_0))$ is mean squared error, and for classification, cross-entropy is used, noting that the best choice for $d(\cdot, \cdot)$ will be task-specific.

In some applications, these simple metrics may be insufficient, and recent work by (Zhang et al. 2018) alludes to the shortcomings of even more complex distance metrics such as PSNR and SSIM. For MNIST digits (28x28 pixels), *Mahanalobis distance* has been shown to be effective (Weinberger and Saul 2009), as well as other methods that achieve translation invariance (Grover and Toghi 2019).

For instance, the experiment in Figure 16 details how simple distance norms (either in input space or latent space) lack robustness to translations of even 5 pixels.

Constrained vs Unconstrained Search

For small δ, minima within the δ ball are rare, and so it is necessary to use a constrained optimisation method in our

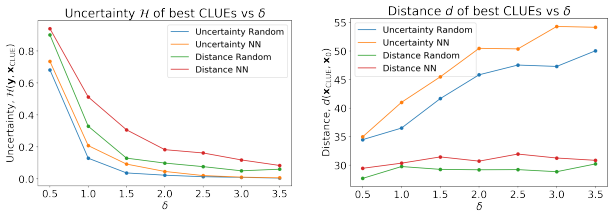


Figure 11: Left: Increasing the size of the δ ball yields lower uncertainty CLUEs. Right: The average distance of CLUEs from $\mathbf{x}_0$ increases with δ. Note that scheme $\mathcal{S}_1$ (blue and green) outperforms scheme $\mathcal{S}_2$ (orange and red) for this dataset.

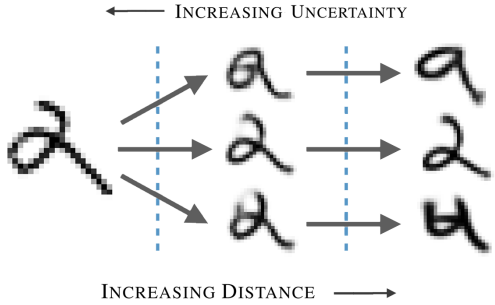


Figure 12: MNIST visualisation of the trade off between uncertainty $\mathcal{H}$ and distance d (example of 3 diverse labels discovered by δ-CLUE).

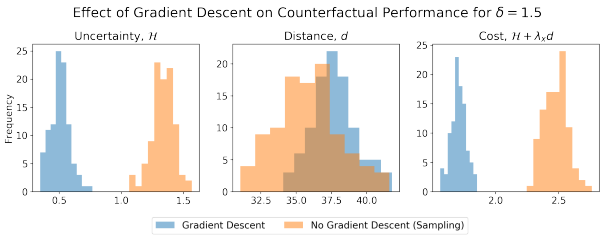


Figure 13: Effect of gradient descent on performance. Blue: gradient descent vs Orange: no gradient descent (sampling). For a particular δ value, we compute 100 random samples for each of the 8 most uncertain MNIST digits. We then perform constrained gradient descent over each of these values and compare the performance to the initial sample.

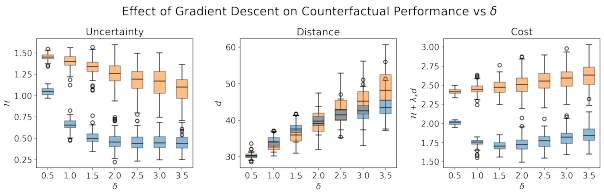


Figure 14: We repeat Figure 14, over a range of δ values, showing that random sampling followed by gradient descent is far superior to only random sampling.

experiments (Figure 17), to avoid solutions being rejected.

Thus, we observe in Figure 18, right, that for small δ, virtually all δ-CLUEs lie on the surface of the ball. The left figure indicates that average latent space distances $\rho(\mathbf{z}_{\text{CLUE}}, \mathbf{z}_0)$ lie close to the line $\delta = \delta$ (purple, dashed), with the distance weighted loss $\mathcal{L}_{\mathcal{H}+d} = \mathcal{H} + d$ producing more nearby δ-CLUEs, as expected. In either case, the effect of the constraint weakens for larger δ, as more minima exist within the ball instead of on it. Depending on user preference, the optimal δ value represents the trade off between uncertainty reduction and distance from the original input.

As stated in the main text, there may exist methods to determine δ pre-experimentation; the distribution of training data in latent space could potentially uncover relationships between uncertainty and distance, both for individual inputs and on average. For instance, we might search in latent space for the distance to nearest neighbours within each class to determine δ. In many cases, it could be useful to provide a summary of counterfactuals at various distances and uncertainties, making a range of δ values more appropriate.

Initialisation Schemes $\mathcal{S}_i$

This appendix details the initialisation schemes $\mathcal{S}_i$ that are used to generate start points for the algorithm. While some schemes may appear preferential in 2 dimensions, the manner at which these scale up to higher dimensions means that we could require an infeasible number of initialisations to cover the appropriate landscape, and so deterministic schemes such as a path towards nearest neighbours within each class ($\mathcal{S}_2$), or a gradient descent into predictions within each class ($\mathcal{S}_5$) might be desirable. We should also note that although the radius, r, of a specific scheme could vary, it is assumed throughout to match the δ value. The following mathematical analysis applies to an ℓ_2-norm $\rho(\mathbf{z}, \mathbf{z}_0) = \|\mathbf{z} - \mathbf{z}_0\|_2$:

$$\mathcal{S}_1 : \rho(\mathbf{z}, \mathbf{z}_0) \sim \mathcal{U}(0, r) \implies \mathbb{E}[\rho(\mathbf{z}, \mathbf{z_0})] = \frac{r}{2}$$

(pick a random radial direction)

$$\mathcal{S}_3 : \rho(\mathbf{z}, \mathbf{z}_0) \sim \mathcal{N}\left(0, \frac{r^2}{4}\right) \text{ s.t. } 0 \leq \rho(\mathbf{z}, \mathbf{z}_0) \leq r$$

(pick a random radial direction)

$$\mathcal{S}_4 : [\mathbf{z} - \mathbf{z}_0]_i \sim \mathcal{U}(-r, r) \text{ s.t. } \rho(\mathbf{z}, \mathbf{z}_0) \leq r$$

(perturb each dimension)

We propose two potential deterministic schemes, that may outperform a random scheme when a) the latent dimension is large, b) δ becomes very large, c) we impose a larger distance weight in the objective function or d) we change datasets. Here $\mathbf{z}_i$ represents the starting point for explanation i, k is the total number of explanations (both used in Algorithm 1), Y represents the total number of class labels y, and $j \in \mathbb{Z}^+$. This produces a total of $Y \times j_{\max} = Y \times \lfloor \frac{k}{Y} \rfloor = k$ explanations if $Y|n$.

$$\mathcal{S}_2 : \mathbf{z}_i = \mathbf{z}_0 + \delta \times \frac{j}{m} \times \frac{\mathbf{z}_y - \mathbf{z}_0}{\rho(\mathbf{z}_y, \mathbf{z}_0)} \; \forall y$$

$$\mathcal{S}_5 : \mathbf{z}_i = \mathbf{z}_0 + \mathbf{s}_{yj} \; \forall y$$

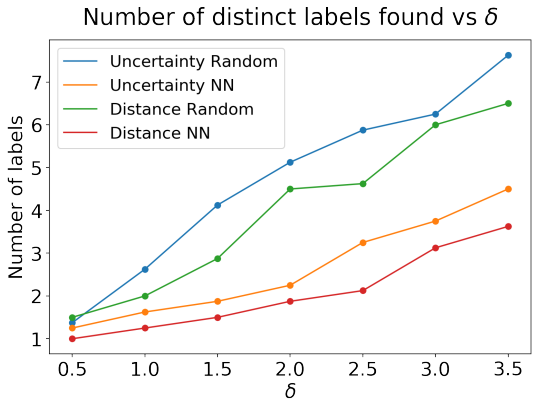


Figure 15: Average number of distinct labels found by sets of 100 CLUEs as δ increases. For small δ, typically only 1 class exists (low diversity). The random search $\mathcal{S}_1$ (blue and green) achieves the greatest diversity.

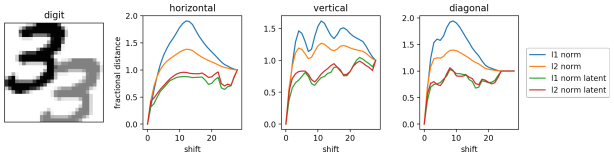


Figure 16: We apply horizontal, vertical and diagonal translations of an MNIST digit (in input space and latent space for ℓ_1 and ℓ_2 norms). As we increase the pixel shift, we compute the distance between the shifted and original digits, divided by the distance between an empty image and the original (to normalise over different metrics, resulting in convergence to 1.0). For reference, the shaded digit indicates the original digit shifted diagonally by 10 pixels.

$$\text{where } 1 \leq j \leq m \text{ and } m = \left\lfloor \frac{k}{Y} \right\rfloor$$

where, for the $\mathcal{S}_5$ scheme, $\mathbf{s}_{yj}$ is defined along a path from $\mathbf{z}_0$ to a radius δ, where at all points the direction of $\mathbf{s}$ is $\nabla_{\mathbf{z}} p(\text{class}(\mathbf{z}) = y)$, and $\frac{j}{m}$ is defined as the fraction travelled along that path.

A series of modifications to these schemes may improve their performance:

- Generating within small regions around each of the points along the path (in $\mathcal{S}_2$ and $\mathcal{S}_5$).
- Performing a series of further subsearches in latent space around each of the best δ-CLUEs under a particular scheme.
- Combining δ-CLUEs from multiple schemes to achieve greater diversity.
- Appendix D details maximising the *diversity* of initialisations, before performing gradient descent on the loss function, which could find utility in cases of low k.

Further MNIST δ-CLUE Analysis

For an uncertain input $\mathbf{x}_0$, we generate 100 δ-CLUEs and compute the minimum, average and maximum uncertainties/distances from this set, before averaging this over 8 different uncertain inputs. Repeating this over several δ values produces Figures 22 through 24.

Special consideration should be taken in selecting the best method to assess a set of 100 δ-CLUEs: the minimum/average uncertainty/distance δ-CLUEs could be selected, or some form of submodular selection algorithm could be deployed on the set. Figure 23 shows the variance in performance of δ-CLUEs; the worst δ-CLUEs converge to high uncertainties and high distances that are too undesirable (the selection of δ-CLUEs is then a non-trivial problem to solve, and in our analysis we simply select the best cost δ-CLUE for each CLUE, where cost is a combination of uncertainty and distance).

In Figure 25, the late convergence of class 2 (green) and the

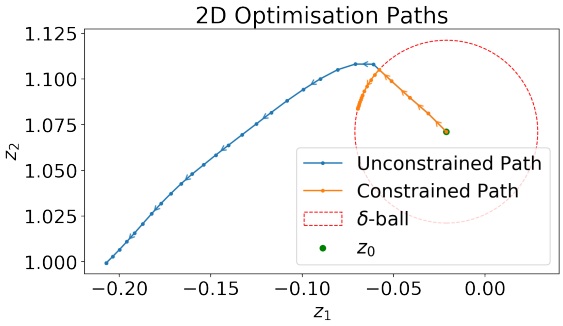


Figure 17: Constrained vs unconstrained gradient descents in a 2D VAE latent space $\mathcal{L}(\mathbf{z}) = \mathcal{H}$. We project values outside of the δ ball onto its surface at each step of the gradient descent.

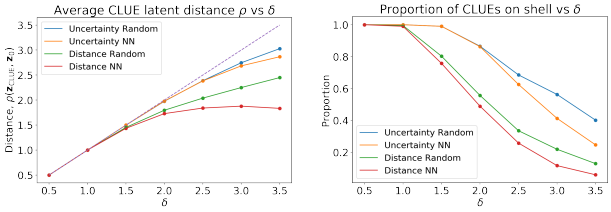


Figure 18: Justification for use of a constrained method. More solutions lie on the ball than inside it for a given δ. Left: How the average final distance in latent space varies with δ. Right: proportion of points that lie on the shell as δ increases. At small δ, almost all minima lie on the shell, whereas at larger δ more lie inside.

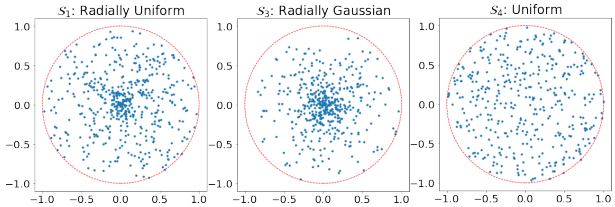


Figure 19: Random generation schemes $\mathcal{S}_1$, $\mathcal{S}_3$ and $\mathcal{S}_4$ depicted in 2D space. In Schemes $\mathcal{S}_3$/$\mathcal{S}_4$ we reject samples outside of the search radius ($\rho(\mathbf{z}, \mathbf{z}_0) > r$). Future schemes may generate within a sub-ball that is smaller than the ball with which we constrain, though this may only be effective in specific latent landscapes.

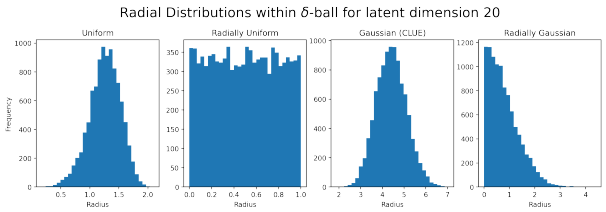


Figure 20: Comparison of initialisation methods. Left: perturbing each dimension uniformally ($\mathcal{S}_4$). Left centre: sampling the radius uniformally $\mathcal{U} \sim [0, 1]$ and picking a random direction uniformally ($\mathcal{S}_1$). Right centre: perturbing each dimension with Gaussian noise $\mathcal{N}(0, 1)$ (Antorán et al. 2021). Right: sampling the radius with a Gaussian $\mathcal{N}(0, 1)$ and picking a random direction uniformally ($\mathcal{S}_3$). Overall, our radially uniform method allows for easiest control over the region of latent space explored (note the scales of the other methods, which are functions of either the probabilistic sampling or the latent dimension, both of which introduce added complexities when performing searches.

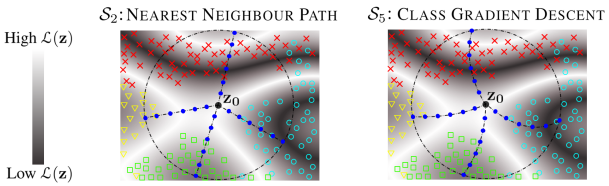


Figure 21: Left: Scheme $\mathcal{S}_2$, nearest neighbour path, searches for the nearest low uncertainty points in training data for each class, before initialising starting points fractionally on the path towards said neighbour. Right: Scheme $\mathcal{S}_5$ performs a gradient descent in the prediction space of the BNN, towards maximising the probability of each class. It too initialises starting points along said path.

lack of 1s, 3s and 6s suggests that $n > 100$ is required, although under computational constraints $n = 100$ yields good quality CLUEs for the prominent classes (7 and 9).

Figure 26 demonstrates how convergence of the δ-CLUE set is a function, not only of the class labels found, but also of the different mode changes that result within each class (alternative forms of each label). In the main text (Figure 15), we count manually the mode changes within each class; in future, clustering algorithms such as Gaussian Mixture Models could be deployed to automatically assess these. The concept of modes is important when a low number of classes exists, such as in binary classification tasks, where we may require multiple ways of answering the question: "what possible mode change could an end user make to modify their classification from a no to a yes?".

Computing a Label Distribution from δ-CLUEs

This final appendix addresses the task of computing a label distribution from a set of δ-CLUEs, as suggested by takeaway 3 of the main text. We use $\delta = 3.5$ and analyse one uncertain input $\mathbf{x}_0$ under the experiment **Distance Random** where $\mathcal{L}_{\mathcal{H}+d} = \mathcal{H} + d$ and $\mathcal{S}_1$ are used.

For (Figure 27, right), we take the minimum costs from (Figure 28, right) and take the inverse square.

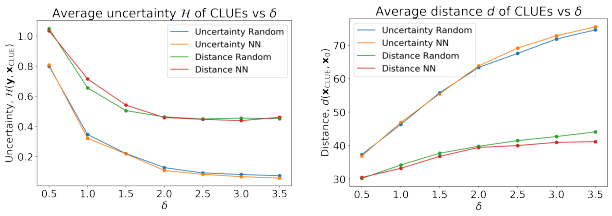


Figure 22: In Figure 11 of the main text, we plot the best (minimum) uncertainties/distances of the δ-CLUEs. Here, we reproduce the plot for average uncertainties/distances and observe that it follows similar trends, shifted vertically, with higher disparity between the $\mathcal{L}_{\mathcal{H}}$ and $\mathcal{L}_{\mathcal{H}+d}$ loss functions.

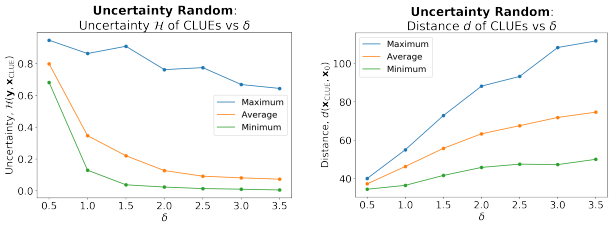


Figure 23: We reproduce Figure 11 for the **Uncertainty Random** experiment ($\mathcal{L}_{\mathcal{H}} = \mathcal{H}$ and $\mathcal{S}_1$), plotting the minimum, average and maximum values found in the set of 100 δ-CLUEs averaged over 8 uncertain inputs.

Effect of VAE latent dimension, m This section reproduces the **Uncertainty Random** experiments, where $\mathcal{L} = \mathcal{H}$, with the random initialisation scheme, $\mathcal{S}_1$, while we vary the latent dimension of the VAE used, m.

The first observation that uncertainty, $\mathcal{H}$ increases as m increases is illustrated in Figure 30. Since the reconstruction error increases as we decrease m, CLUEs become both blurrier and less uncertain, as the VAE fails to reconstruct a high quality image and resorts to more generic, smoothed shapes. Secondly, we see in Figure 31 that the distribution of CLUEs within the δ ball is pushed towards the surface of the ball as m is increased. The final observation is that, for a given δ, lower dimensional VAEs achieve greater diversity (Figure 32) as a result of their failure to reconstruct images (this ties into the second observation).

C Diversity Metrics for Counterfactual Explanations

This appendix details the properties of each diversity metric in conjunction with sets of counterfactuals from different δ values. We also provide further clarity on the coverage diversity metric.

Observe in Figure 33 that as δ increases, diversity increases virtually monotonically. The metrics operating in $\mathbf{y}$-space (prediction coverage, number of distinct labels and entropy of the label distribution) increase less smoothly with δ. Furthermore, the volatility of the DPP measure is highlighted; sets of counterfactuals that led to marginal increases for all other metrics caused a sharp increase at $\delta \approx 3.0$ for DPPs.

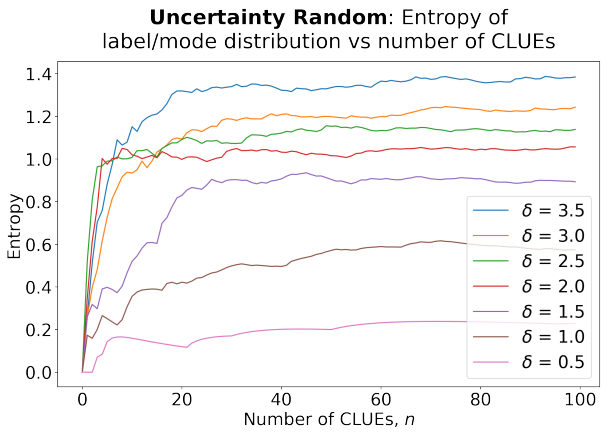


Figure 24: A more refined plot of Figure 15, left, to answer the question: "How many times must we run δ-CLUE in order to saturate the entropy of the label distribution of the δ-CLUEs found?".

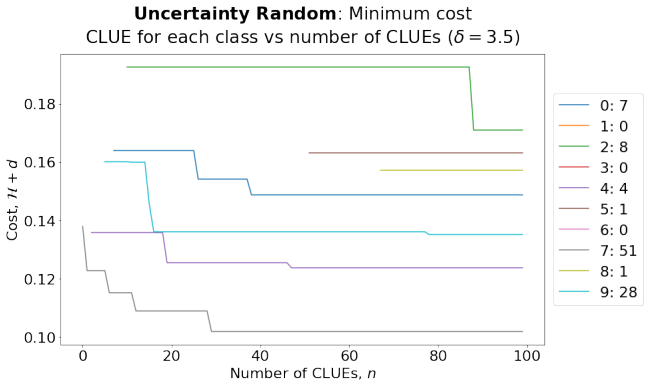


Figure 25: For a single uncertain input $\mathbf{x}_0$, we generate n δ-CLUEs and observe how the minimum cost (a combination of uncertainty and distance) of δ-CLUEs for each class converges. Legend shows class labels 0 to 9, and the final number of each discovered by δ-CLUE (summing to 100).

Figure 34 demonstrates further interesting properties of the metrics. Intuitively, we might expect the diversity in a set of k counterfactuals to increase with k, but this is not the case for many of the metrics. Coverage, prediction coverage and the number of distinct labels all behave in this way, as is expected i.e. the number of distinct labels cannot reduce as more items are introduced. However, DPPs, APD and entropy of the label distribution exhibit decreases as k increases. We expect the latter metric to increase and plateau, as in Figure 7, with some random noise around this convergence. However, if the diversity of a set does not rise fast enough with k (which it does not in our experiments), DPPs and APDs fundamentally become smaller, as they normalise in some way with respect to k. The effect of k on the diversity metrics has implications regarding the tuning of the hyperparameter λ_D, where the optimal value is now likely to change significantly with k. However, even for fixed k, this raises concerns for the sequential ∇-CLUE optimisation, to be discussed in Appendix D.

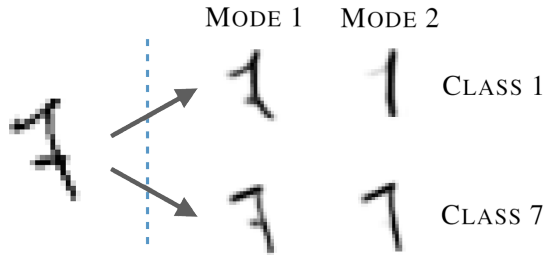


Figure 26: MNIST: 10 class labels exist (0 to 9), whereas an undefined number of modes within each class also exist. These modes are counted manually in this paper.

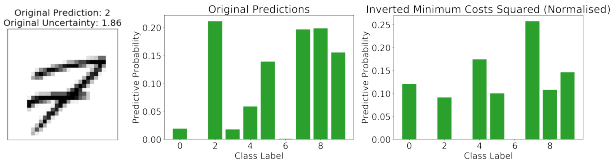


Figure 27: Left: An original uncertain input that is incorrectly classified. Centre: The original predictions from the BNN. Right: The new **label distribution** based off of the δ-CLUEs found.

Coverage as a Diversity Metric

Figure 35 visualises this metric within MNIST to aid understanding. We now determine the bound on D under the coverage metric. Take $S_+ = \sum_{i=1}^{d'} \max(x_i)$ and $S_- = \sum_{i=1}^{d'} \min(x_i)$ to represent the sum over all features of the maximum and minimum values each feature can take, and that $|\mathbf{x}_0| = \sum_{i=1}^{d'} (\mathbf{x}_0)_i$ (the sum over all features of the uncertain input $\mathbf{x}_0$), where d' is the dimensionality of the feature space. The minimum coverage of a counterfactual ($D = 0$) clearly occurs when the counterfactual is simply the original input. The maximum coverage can be calculated as:

$$D_{\max} = \frac{1}{d'} \left((S_+ - |\mathbf{x}_0|) - (|\mathbf{x}_0| - S_-) \right) = \frac{S_+ - S_-}{d'}$$

(independent of $\mathbf{x}_0$).

In the MNIST experiments performed, we have $d' = 28 \times 28 = 784$, with the maximum and minimum values of each pixel to be 1 and 0 respectively, thus giving $S_+ = 784$ and $S_- = 0$. This does indeed result in $D_{\max} = 1$. If S_+ and S_- are known, we can guarantee this normalisation by dividing the coverage by $D_{\max}$. In other applications, where the features can scale infinitely, normalising D is not possible.

In theory, 2 counterfactuals are sufficient to achieve the maximum coverage (one counterfactual of all features at their maximum values, and one counterfactual of all features at their minimum values e.g. one fully black and one fully white image in MNIST). While coverage must can never decrease as k increases, the exact nature of this relationship is dependent on the dataset and the counterfactual generation method (e.g. Figure 34). This is analytically indeterminate and thus cannot be regularised.

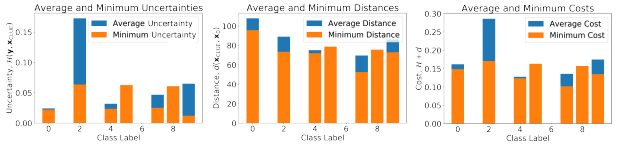


Figure 28: Left: Average and minimum uncertainties $\mathcal{H}$ for each class in the δ-CLUE set. Centre: Average and minimum distances d. Right: Average and minimum costs, where the weight λ_x is multiplied by the distance function and added to the uncertainty.

Future Work

Future work might include a human subject experiment to determine the metric most aligned with human ideas of diversity; or better still, what each of the metrics represent themselves with regards to human intuition. The set of diversity metrics proposed in this paper are not exhaustive either, and further investigation of other metrics, perhaps with inspiration drawn from said human subject experiments, could provide meaningful insights.

D ∇-CLUE

We observed that maximising the diversity of the *initialisations* within the δ ball before performing the gradient descent helps to improve diversity; we thus perform an initial gradient descent to maximise the diversity of the initialisations, where n_i is the number of gradient descent steps performed. However, as n_i becomes large and diversity converges, we experience starting initialisations moving as far away from each other as possible in the δ ball, which can degrade performance with respect to uncertainty. As in the main text, we perform experiments at a fixed δ value and optimise for DPP diversity in $\mathbf{z}$-space. Results are displayed in Figures 36 through 38 (where $k = 10$).

Sequential ∇-CLUE optimisation

Figure 39 corresponds to Figure 1 of the main text for a sequential ∇-CLUE scheme. Note the presence of regions of cost added with every new CLUE found (analogous to hills in the objective function) due to the diversity component in the objective function. The lower-center image demonstrates how this might affect the gradient descent of future CLUEs when in close proximity of older ones i.e. the optimisation path curves towards a new minima. Possible pitfalls of this method include the manner with which D scales with k (it is undesirable to have to re-tune the hyperparameter λ_D at each iteration). For instance, we might wish to remove the normalisation term $\binom{k}{2}$ of APD diversity when performing sequential ∇-CLUE.

Sequential ∇-CLUE with Penalty Terms

This section trials the use of a penalty term instead of a diversity term in the objective of sequential ∇-CLUE. The following experiments are conducted with $\delta = 4$, $D = 0$ and an added penalty term of $\frac{\lambda_D}{d(\mathbf{z}, \mathbf{z}_{\text{CLUE}})}$ for each new counterfactual $\mathbf{z}_{\text{CLUE}}$. This is run on the 8 most uncertain MNIST digits, generating $k = 100$ CLUEs for each. We calculate

Figure 29: The 100 δ-CLUEs yielded in this experiment (**Distance Random** with $\delta = 3.5$). Above digits: Label prediction and uncertainty. Below: Distance from original in input space. Low uncertainty CLUEs may be found at the expense of a greater distance from the original input.

diversity/performance of each set of 100 and average over all 8. Note that at $\lambda_D = 0$, this is equivalent to running δ-CLUE, and the loss function is not weighted by distance (i.e. $\mathcal{L} = \mathcal{H}$).

Figure 42 details the effect of λ_D on performance. This analysis suggests that δ-CLUE performs better on all fronts for $k = 100$ CLUEs (though ∇-CLUE performance may change for a smaller number of CLUEs i.e. $k = 10$). Figures 40 and 41 show that diversity actually decreases as a function of λ_D, implying that simple penalty terms, as described above, are insufficient for achieving diversity in the metrics put forward.

Future Work

We devote this section to performing a full ablative analysis of all diversity metrics, since only DPP diversity in $\mathbf{z}$-space was trialled in the main paper. We would also experiment more fully the strategy of finding diverse initialisations as opposed to optimising for diversity; the latter method, used in this paper, has been shown to compromise the performance of the CLUEs found, and thus finding the best start-

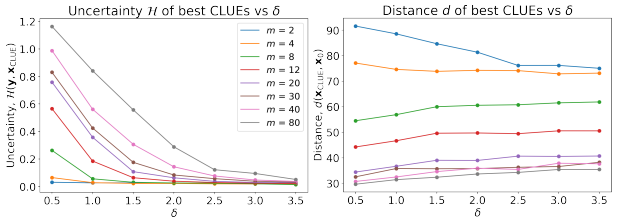


Figure 30: Reproduction of Figure 11 for a range of VAE latent dimensions m. Reconstruction error of high dimensional VAEs is small (uncertainty remains high and distance remains low).

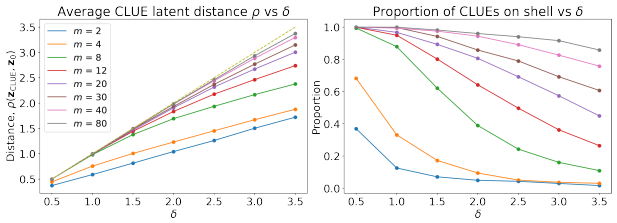


Figure 31: We highlight the change in the optimal δ value with m (as latent dimension increases, most minima occur at larger δ, leading to a higher proportion of CLUEs on the shell).

ing initialisations and performing δ-CLUE (where $\lambda_D = 0$) might yield equally diverse sets that perform better.

E GLAM-CLUE: GLobal AMortised CLUE

Although drawing inspiration from Transitive Global Translations (TGTs), as proposed by Plumb et al. (2020), our method performs a different operation; instead of learning translations in input space that result in high quality mappings in a lower dimensional latent space, we find that results are best when learning translations in latent space, as described in the main text. This is seen also in the fact that the latent space DBM baseline outperforms input DBM; the difference between means translation is a special case of the GLAM-CLUE translation that we propose, and is the value we use as an initialisation during gradient descent. We also provide a visualisation for the input space DBM baseline in Figure 43. In the case of image data, the resulting image when DBM was added to the original input had to be clipped to match the scale of the data (in our case, between 0 and 1).

Future Work

While a GLAM-CLUE translation shows very good performance in the experiments demonstrated in the main text, it is not clear that performance would be maintained in all situations. We question the performance in cases where a group of uncertain points are not easily separated from a group of certain points by a simple translation (as in Figure 44).

To this end, there are two further avenues to explore: the use of more complex mapping functions, or the potential to split the uncertain groups into groups that translations perform well on (clustering the uncertain points in Figure 44 into 3 groups). This latter approach would maintain GLAM-CLUE's utility in computational efficiency, as we demonstrate that learning simple translations is extremely fast.

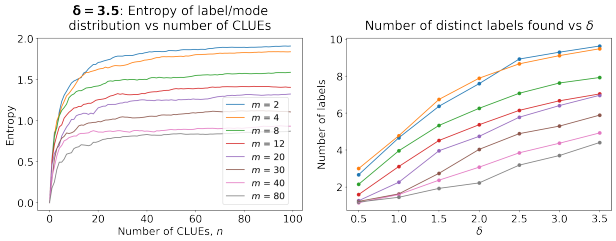


Figure 32: We again highlight that larger latent dimensions require larger δ balls to capture necessary minima.

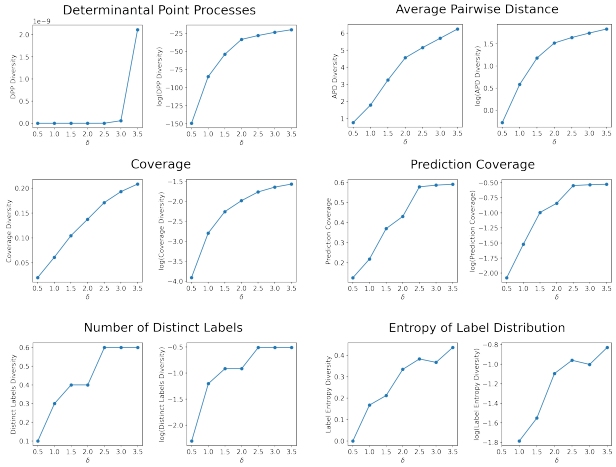


Figure 33: For a particular δ value and uncertain input x_0, we compute 100 δ-CLUEs (as in Figure 29), repeating this over a range of δ values. For each of these sets, we apply the various diversity metrics proposed in Table 1 of the main text, plotting also the logarithms of the metrics (the latter only appearing to be meaningful with Determinantal Point Processes).

We have the additional issue of selecting an appropriate λ_θ parameter in the algorithm to best tune the trade-off between uncertainty and distance. Dosovitskiy and Djolonga (2020) propose a method that replaces multiple models trained on one loss function each by a single model trained on a distribution of losses. A similar approach could be taken by using a distribution over individual terms of our objective and varying the hyperparameter weight at the **inference step**. This could yield a powerful technique for minimising uncertainty and distance but allowing the trade-off between the two to be selected **post-training**.

As far as more complex datasets are concerned, preliminary trials on the black and white Synbols dataset showed that the DBM baselines produced almost incoherent results. Our understanding is that, in input space, taking the mean of a particular class that contains an equal distribution of points with white backgrounds and points with black backgrounds will result in a cancellation between the two, such that the mean vector is close to zero. The same analogy in latent space might be that black points within a particular class may not be clustered in a similar region to those of white points for the same class. As such, further clustering, as alluded to above and in Figure 44 is probably necessary.

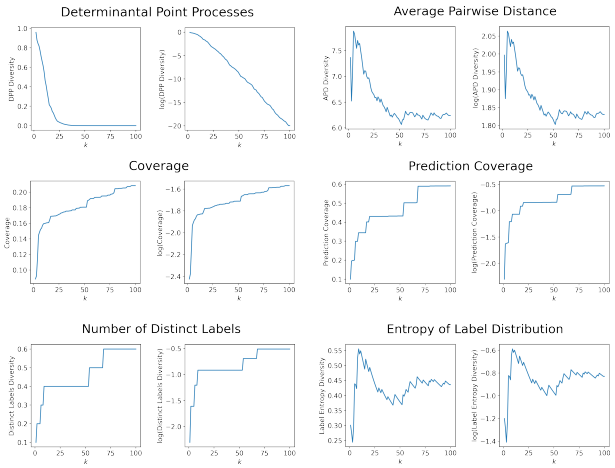


Figure 34: For a particular uncertain input x_0 and δ value, we compute 100 δ-CLUEs, and proceed to plot the diversity of the first k CLUEs in the range $1 \leq k \leq 100$.

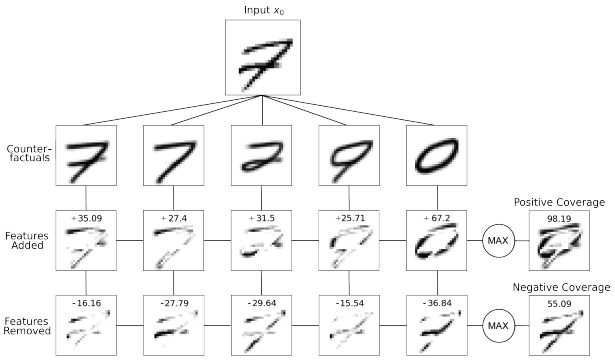


Figure 35: To compute positive and negative coverage, we take the positive and negative differences between counterfactuals and the original input, and further combine these by selecting the maximum change observed in a given feature (pixels in this case). We see that the 5 counterfactual explanations shown demonstrate changes that almost completely remove the original input, whilst adding features across a range of other areas. Total coverage is the sum of the positive and negative coverages.

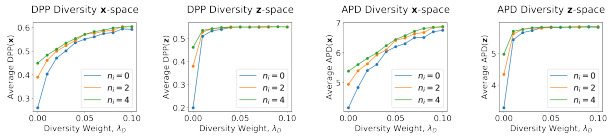


Figure 36: Effect of λ_D and n_i on diversity. We observe that either parameter can provide a route to achieving more diverse sets of counterfactuals.

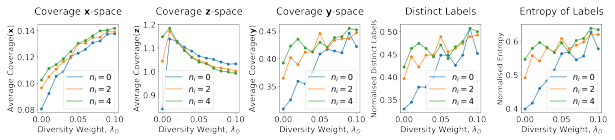


Figure 37: Continuation of Figure 36 for the remaining diversity metrics.

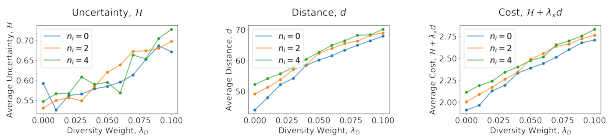


Figure 38: Effect of λ_D and n_i on performance. Unfortunately, performance degrades as we strive for greater diversity in the optimisation, although the gain in diversity can outweigh this.

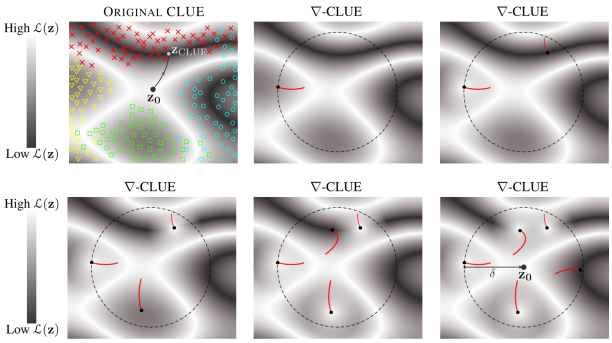


Figure 39: Conceptual colour map of objective function $\mathcal{L}(z)$ with $\mathbf{z}_0$ located in high cost region. Upper Left: Gradient descent to region of low cost (original CLUE). Training points in colour. Left to Right, Top to Bottom: Gradient descent constrained to δ-ball using sequential ∇-CLUE. The effect of the updated diversity term at each step is to push the current solution away from previous ones.

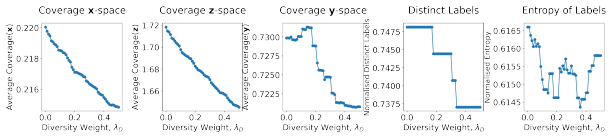


Figure 40: DPP and APD Diversity in $\mathbf{x}$ and $\mathbf{z}$-space as a function of λ_D. We see that similar patterns are exhibited in both $\mathbf{x}$ and $\mathbf{z}$-space.

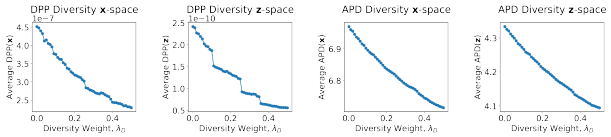


Figure 41: Coverage in $\mathbf{x}$, $\mathbf{z}$ and $\mathbf{y}$-space, Distinct Labels and Entropy of Labels as a function of λ_D.

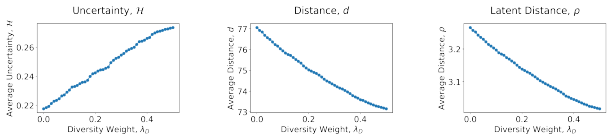


Figure 42: Uncertainty, $\mathcal{H}$ and distance in $\mathbf{x}$ and $\mathbf{z}$-space (d and ρ respectively) as a function of λ_D. Adding stronger diversity terms appears to push solutions towards the centre of the δ ball as suggested by the graph on the right. Our belief that on average input space distance decreases roughly monotonically as latent space distance decreases is confirmed.

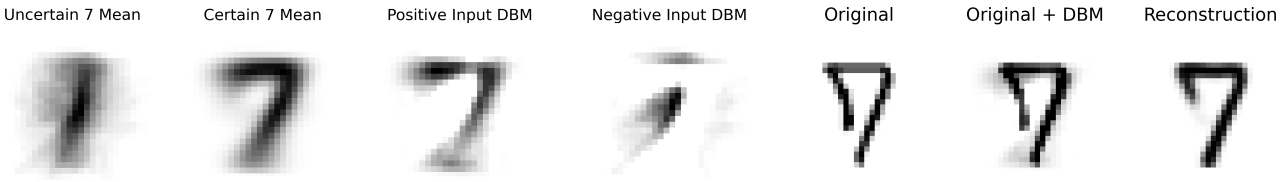


Figure 43: Visualisation of the input DBM baseline. The mean of all uncertain 7s in the MNIST training data is taken, followed by the mean of all certain 7s is shown in the 1st and 2nd plots. The 3rd and 4th plots show the positive and negative changes made when moving from uncertainty to certainty. The 5th to 7th plots illustrate how a final, certain counterfactual explanation is produced using this baseline (by adding the difference between means in input space and reconstructing the result).

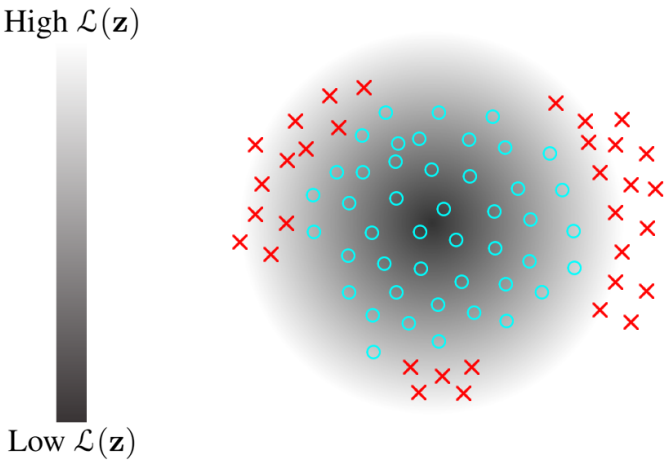


Figure 44: 2D visualisation of a shortcoming of GLAM-CLUE. The group of uncertain points (red) is not easily mapped onto the group of certain points (blue) by a single translation, unless further division of the uncertain group (into 3 clusters for instance) is performed, or a more complex mapper is learnt.