# Diverse, Global, and Amortised Counterfactual Explanations for Uncertainty Estimates

## Seminar: Uncertainty quantification in machine learning

Sruthi Aikkara
Matriculation Number: 229386
Summer Semester 2021/22

Supervised by: Jelle Hüntelmann

## Abstract

*Machine learning modeling can often result in predictions that are uncertain. When there is uncertainty in the prediction, there is a possibility to give the interpreter different counterfactual explanations which might be the reason of this uncertainty. A technique called Counterfactual Latent Uncertainty Explanation (CLUE) (Antoran et al. (2021)) uses an uncertain data point to find a single change in the on-manifold that increases the certainty of the input, thereby increasing the prediction's accuracy. The multiplicity issue of CLUE is addressed by the introduction of $\delta$-CLUE, which creates a set of CLUEs inside the $delta$-ball distance of the original input. A development over $\delta$-CLUE is DIVerse CLUE ($\nabla$-CLUE), which focuses only on the distinct counterfactual explanations. Finally the GLobal AMortised CLUE (GLAM-CLUE) is introduced which uses a single function call that makes the input more certain and also makes the model computationally more efficient.*

## 1. Introduction

Estimating uncertainty is crucial to machine learning. Uncertainty can lead to inaccurate predictions. Therefore, it's essential to comprehend the factors that contribute to uncertainty before modeling is done. The experts can then analyze these features and estimate the value ranges in which they will provide a more accurate prediction. Therefore, more counterfactuals can be produced if there is a large degree of prediction uncertainty. Therefore, it is possible to think of uncertainty explanations as the first step towards model explanation.

Antoran et al. (2021) proposed CLUE or Counterfactual Latent Uncertainty Explanation which is such a state-of-art method introduced which identifies the single on-manifold change to an uncertain input $x_0$ that makes it more certain. These counterfactual explanations are generated in the latent space of an auxiliary deep generative model(DGM). The CLUE minimises the loss value to make the uncertain input more similar or closer to the prediction space.

As we examine CLUE, we find that it has some downsides. Think about a situation where someone is renting out an apartment and learns that the person won't be receiving the rent they were expecting after all the calculations. They can take into account a wide range of factors, such as the balcony, floor, whether an elevator is available, the location, whether pets are permitted, etc. The counterfactual explanation offered by CLUE, however, calls for changing the location of the apartment to a more desirable area, which is something impractical to do. It would be beneficial for the owner if we offered additional suggestions that were practical and CLUE doesn't have this possibility to give multiple suggestions. Hence $\delta$-CLUE is introduced which generates a set of CLUEs within the $\delta$ ball distance of the original uncertain input $\mathbf{x_0}$ in the DGM's latent space. Some of $\delta$-CLUEs were redundant or similar, so diversity metrics are applied to gauge how unique these CLUEs are. These measures are used to optimize the generated CLUEs, and the resulting diversified CLUE is known as $\nabla$-CLUE. The GLAM-CLUE (GLobal AMortised CLUE) is developed to address the computational inefficiencies of CLUE and speed up the process by having a single function call to map a group of set of uncertain inputs to certain inputs. Figure 1 shows how all the new CLUE algorithms work in comparison to the original CLUE.

The efficiency of the CLUE techniques is evaluated

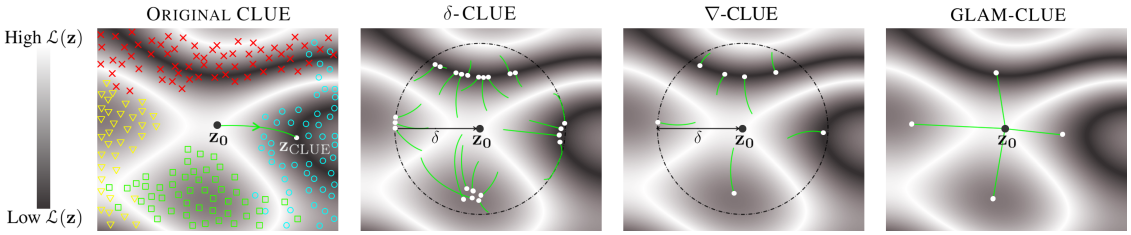


**Figure 1. (Ley et al. (2021)) $\mathbf{z}_0$ is the latent space representation of uncertain input $\mathbf{x}_0$. The left most picture shows the original CLUE with training data shown in different colours and gradient descent to regions also having low cost. The small white circles in the pictures represent the counterfactual explanations. Left center represents the $\delta$-CLUE with $\delta$ being the radius of the circle and gradient descent within this $\delta$ ball. Right center is $\nabla$-CLUE with more diverse CLUEs at each region. Right most picture is the computationally efficient GLAM-CLUE.**

by performing several experiments on various data sets using various diversity metrics, altering the initialization criteria, and so on. In the final sections of this paper, CLUE's potential for the future is also discussed.

## 2. Methods

As we go into more details of the different methods introduced, let's actually look into what a counterfactual explanation is. If we go back to the previous example of the owner renting out his apartment, if allowing pets in the apartment increases his expected rent, then that is a counterfactual explanation. The counterfactual explanations don't give advice or tell the user what action to take; instead they help the user by giving different possibilities or suggestions by which they can achieve their expected goal. The counterfactual explanations are considered to have the "Rashomon effect" named after an old film in which several characters attempt to justify a Samurai's murder. Every witness in the film has an unique account of how the Samurai was killed, and this is exactly what counterfactual explanations are. Every counterfactual explanation presents a unique "narrative" of how a particular result might occur.

The smallest on-manifold change on an uncertain input that increases prediction certainty is known as CLUE (Counterfactual Latent Uncertainty Explanation). On the latent space of an auxiliary deep generative model (DGM), the explanation search for model uncertainty is conducted. Consider $\mathbf{x}$ is the uncertain input and $\mathbf{z}$ is the latent space. DGM has an encoder $\mu_\phi(\mathbf{z} \mid \mathbf{x})$ and a decoder $\mu_\theta(\mathbf{x} \mid \mathbf{z})$. $\mathcal{H}$ is the uncertainty quantification which means if they are confident enough if the prediction $\mathbf{y}$ is correct or not. There is a distance metric that is presented as follows to determine how similar the original input and the CLUE-generated input are to one another:

$$d\left(\mathbf{x}, \mathbf{x}_0\right) = \lambda_x d_x\left(\mathbf{x}, \mathbf{x}_0\right) + \lambda_y d_y\left(f(\mathbf{x}), f\left(\mathbf{x}_0\right)\right) \quad (1)$$

Here $f(\mathbf{x}) = \mathbf{y}$ is the function which maps the input $\mathbf{x}$ to a label $\mathbf{y}$. $\lambda_x$ and $\lambda_y$ are the hyper parameters that control the trade-off between distance and uncertainty which is explained later in this report. CLUE has to minimise this loss function given as:

$$\mathcal{L}(\mathbf{z}) = \mathcal{H}\left(\mathbf{y} \mid \mu_\theta(\mathbf{x} \mid \mathbf{z})\right) + d\left(\mu_\theta(\mathbf{x} \mid \mathbf{z}), \mathbf{x}_0\right) \quad (2)$$

This yields $\mathbf{x}_{CLUE} = \mu_\theta(\mathbf{x} \mid \mathbf{z}_{CLUE})$ and $\mathbf{z}_{CLUE}$ is the minimal value of loss function given as $\mathbf{z}_{CLUE} = argmin_{\mathbf{z}}\mathcal{L}(\mathbf{z})$.

### 2.1. $\delta$-CLUE

Bhatt et al. (2021) propose $\delta$-CLUE, which generates a set of counterfactuals that are within $\delta$ ball distance of the original input in latent space, to overcome the multiplicity issue of CLUE. These multiple solution search is initialised randomly by different initialisation schemes in different regions of the $\delta$ ball. This is what CLUE does as well, however CLUE has an inclination to look for minimal value in a constrained region, and it's possible that this minimal region may be distant from original input. Figure 2 shows the comparison between the original CLUE and $\delta$-CLUE.

Similar to CLUE, the new CLUEs also employs BNNs (MacKay (1992)) as its classifiers and Variational Auto Encoders (VAEs) (Kingma and Welling (2013)) as its DGM. The distance metric used here is the L1-norm which is simply the Manhattan distance between the original input and the input in the latent space. The $\delta$-CLUE loss function is similar to the original CLUE loss function as in Equation 2 with an additional $\delta$ constraint added to it. The loss function yields

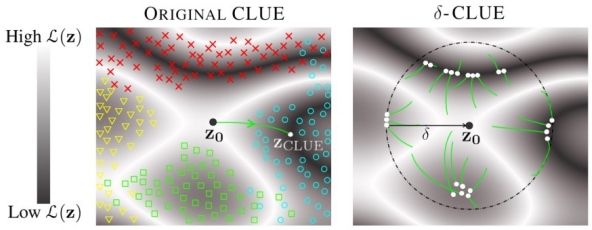


**Figure 2. Left: Original CLUE with training data shown in colours. Right: $\delta$-CLUE with white circle representing the counterfactuals in latent space.**

$\mathbf{x}_{\delta-CLUE} = \mu_\theta(\mathbf{x} \mid \mathbf{z}_{\delta-CLUE})$ and $\mathbf{z}_{\delta-CLUE} = argmin_{\mathbf{z}:\rho(\mathbf{z},\mathbf{z}_0)\leq\delta}\mathcal{L}(\mathbf{z})$. Here $\rho(\mathbf{z}, \mathbf{z}_0)$ is the L-2 norm which is the Euclidean distance and $\mathbf{z_0} = \mu_\phi(\mathbf{z} \mid \mathbf{x_0})$.

When initializing $\delta$-CLUEs in various regions of latent space, the $\delta$ constraint value is adjusted to obtain an optimal value at different stages (Figure 2). There are two initialization schemes, as was previously stated. One uses gradient descent while the other sampling the area around the uncertain input. Even though the sampling initialization technique is computationally faster, using the latter produced better performance with the highest quality of counterfactuals. The first initialization method uniformly distributes CEs in a $\delta$ ball within a random radius. The second initialization technique generates CEs for each class using the Nearest Neighbor path, as shown in Figure 3.

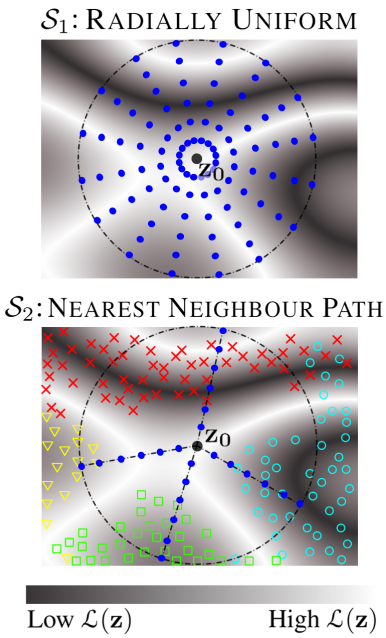


**Figure 3. Initialisation schemes. Top: Initialised along radius via random sampling. Bottom: Initialisation done by Nearest Neighbour Path.**

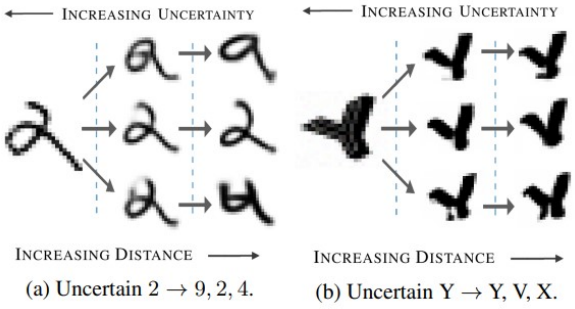


**Figure 4. Uncertainty-Distance trade-off. Left: MNIST data set. Right: Synbols data set.**

As explained before the hyper parameters $\lambda_x$ and $\lambda_y$ controls the trade-off between the distance and uncertainty. The result of decreasing uncertainty over increasing distance from original input is shown in Figure 4.

**Diversity Metrics for CE Optimisation** There is a possibility that many of the $\delta$-CLUEs produced will be identical or similar to one another. Thus, the diversity measures are introduced in order to identify distinct CLUEs and optimize the algorithm. For the distinctiveness evaluation of CLUEs, various diversity measures are available, and depending on the data set, one diversity metric or a combination of diversity metrics may be utilized. Diversity can be evaluated on points either pairwise or between labels of counterfactuals. Here are a few of these diversity metrics:

*Determinantal Point Processes:* Here the determinantal point process explained by Mothilal et al. (2020) is used.The determinant of a matrix which contains the distance between CEs are constructed. The sub matrix which gives the highest determinant value exhibits the highest diversity and that matrix is chosen. So the motive is to leverage the DPP and the determinant, $det(\mathbf{K})$ is given by:

$$\mathbf{K}_{i,j} = \frac{1}{1 + d\left(\mathbf{x}_i, \mathbf{x}_j\right)}$$

The DPP procedure requires the calculation of matrix determinants, hence it will be computationally expensive for a larger number of points.

*Average Pairwise Distance:* As the name suggests, the diversity is calculated as the average pairwise distance between the distinct pairs of counterfactuals (Bhatt et al. (2021)). Average pairwise distance is given as:

$$\frac{1}{\binom{k}{2}} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} d\left(\mathbf{x}_i, \mathbf{x}_j\right)$$

As $\delta$ value increases, the average pairwise distance also increases monotonically.

*Coverage:* Coverage is a diversity metric that can be used for the optimisation of chosen counterfactuals.Each counterfactual is given a weight in coverage based on the feature importance. Coverage is hence the sum of all these distinct features (Ribeiro et al. (2016)). This makes it easier to select a subset from the collection of all CEs while maintaining the set's diversity. When examining the features in coverage, the features added as well as the ones removed are taken into account, as merely considering one of them could result in CE deviating from the original input. The coverage calculation hence takes the sum of positive and negative feature changes that are made to the counterfactual.

$$\frac{1}{d'} \sum_{i=1}^{d'} \left( \max_j \left(\mathbf{x}_j - \mathbf{x}_0\right)_i + \max_j \left(\mathbf{x}_0 - \mathbf{x}_j\right)_i \right)$$

Figure 5 shows how coverage is calculated for an uncertain input $\mathbf{x_0}$. Five CEs are created, and if only positive changes are considered, then they are far away from $\mathbf{x_0}$. Hence the total coverage is the sum of both positive and negative changes.

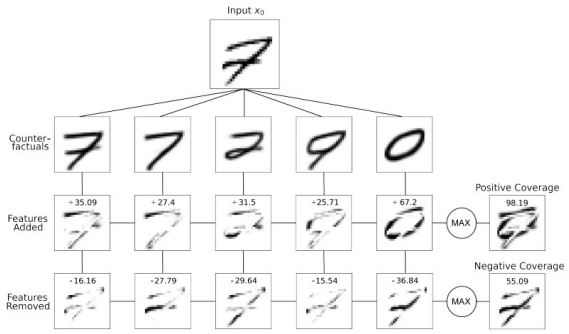


**Figure 5. The positive and negative coverage is calculated and the maximum pixel value is taken. We can see only considering either features added or removed is distant from the actual input. Hence total coverage is taken as the sum of the positive and negative coverage values.**

*Prediction Coverage:* If we maximize the prediction of one class label in a set of counterfactuals, eventually it will minimize the prediction of the other labels. Therefore, the goal of prediction coverage is to maximize the prediction of a certain class label. This count of the maximum predicted labels is then averaged over all of the predictions that are available. Prediction coverage is calculated as:

$$\frac{1}{c'} \sum_{i=1}^{c'} \max_j \left[ \left(\mathbf{y}_j\right)_i \right]$$

*Distinct Labels/Entropy of Labels:* Diversity of class also rises when a prediction has a large number of different labels. So if it is a binary classification, then diversity will be less. The probability of a class is defined as the number of counterfactuals in that class over the total number of counterfactuls. This probability of a particular class $j$ with $c'$ as the number of CEs are given as:

$$\frac{1}{c'} \sum_{j=1}^{c'} \mathbf{1}_{[\exists i: y_i = j]}$$

Below equation calculates the entropy of these class labels as just calculating probability is not good enough for diversity evaluation.

$$-\frac{1}{\log c'} \sum_{j=1}^{c'} p_j(k) \log p_j(k)$$

The Figure 6 shows how the different diversity metrics responds to increase in the $\delta$ values.

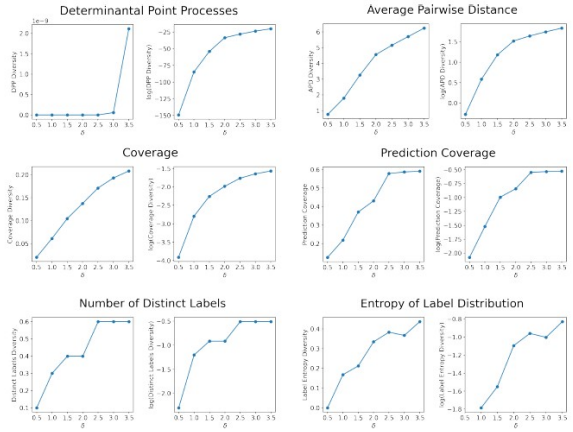


**Figure 6. Logarithmic of diversity metrics are plotted against the $\delta$ values.**

As the $\delta$ value increases, we can see that the diversity also increases monotonically for all the metrics. The DPP process appears to exhibit considerable volatility in the tests reported in this study, and there is a steep

increase in the diversity at $\delta$=3.0. Also the statement that as the number of counterfactuals increases, the diveristy also increases is not always true for all the diversity metrics. This is true for Coverage, prediction coverage and number of distinct class labels, but not true for DPP, APD and entropy of class labels as they exhibit a decrease in diversity with an increase in the number of counterfactuals.

## 2.2. $\nabla$ - CLUE

We can generate an optimized set of diverse CLUEs by applying the diversity metrics discussed in the section previously. This set of CLUEs are called DIVerse CLUE or $\nabla$-CLUE. There are mainly 2 steps to be followed in a generating $\nabla$-CLUE. The first step is to choose the diversity metric. The second step is to optimise the set of k counterfactuals either simultaneously in latent space (Mothilal et al. (2020)) or sequentially. Similarly to the $\delta$-CLUE algorithm, points are generated within a particular radius using one of the initialisation schemes. The $\nabla$-CLUE is same as $\delta$-CLUE when the diversity weight $\lambda_D$ = 0. This is in turn equals to the original CLUE algorithm if $\delta = \infty$, number of counterfactuals, $k$ = 1 and radius $r$ = 0. The $\nabla$-CLUEs generated for an uncertain input $\mathbf{x_0}$ is given in Figure 7. Here $\mathcal{H}$ is the uncertainty, $d$ represents the distance between the CE and original input, and $\rho$ is the latent distance.

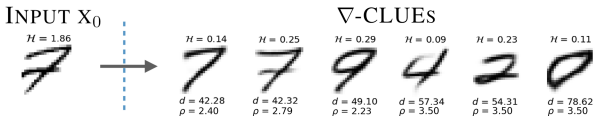


**Figure 7. A set of six diverse counterfactual explanations are generated. Original input is given on the left most side which can be a 7 or 9.**

As explained before, there are two types of optimisation, simultaneous and sequential which is explained in detail in the section below.

**Simultaneous Diversity Optimisation:** In Simultaneous Diversity Optimisation, CLUEs are optimised simultaneously in latent space. As the number of counterfactuals k increases, the diversity also monotonically increases. This can be avoided while using the simultaneous diversity optimisation. The loss function has to be minimised and it is given by:

$$\mathcal{L}\left(\mathbf{z}_1, \ldots, \mathbf{z}_k\right) = -\lambda_D D\left(\mathbf{z}_1, \ldots, \mathbf{z}_k\right) + \frac{1}{k}\sum_{i=1}^{k}\mathcal{L}\left(\mathbf{z}_i\right)$$

where $\lambda_D$ is the diversity weight and $D$ is the Diversity function and

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

This yields

$$X_{\text{CLUE}} = \mu_\theta\left(X \mid Z_{\text{CLUE}}\right)$$

where $Z_{\text{CLUE}} = \arg\min_{\mathbf{z}_1,\ldots,\mathbf{z}_k} = \mathcal{L}\left(\mathbf{z}_1, \ldots, \mathbf{z}_k\right)$.

**Sequential Diversity Optimisation:** In Sequential Diversity Optimisation, each new counterfactual is appended to the set to perform $\nabla$-CLUE. The loss function is to be minimised on each new appended CE and it is given by:

$$\mathcal{L}(\mathbf{z}) = \lambda_D D\left(Z_{CLUE} \cup \mathbf{z}\right) + \mathcal{H}\left(\mathbf{y} \mid \mu_\theta(\mathbf{x} \mid \mathbf{z})\right) + d\left(\mu_\theta(\mathbf{x} \mid \mathbf{z}), \mathbf{x}_0\right)$$

This yields $\mathbf{z}_{\text{CLUE}}$ which can be seen appended to the already existing set of CEs. Figure 8 shows how the sequential $\nabla$-CLUE works.

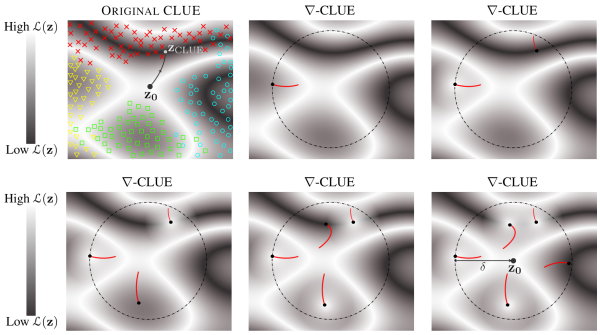


**Figure 8. Upper left image shows the original CLUE with training points given in different colours. Rest of the images depict how sequential diversity optimisation is performed with each new counterfactual being generated at a distinct region.**

We can see that each step, the new solution is added to a new region by calculating the diversity term. Here the gradient descent scheme is used to initialise the CE generation in the $\delta$ ball distance using sequential diversity optimisation.

## 2.3. GLAM-CLUE (GLobal AMortised CLUE)

As CLUE works with local uncertainty explanation, this will lead to computational inefficiency when a large set of data is involved. If there are large sets of uncertain inputs, then large sets of counterfactuals can also be generated, as explained in the first section. Dealing

with this extensive collection of CEs is unreliable. So GLAM-CLUE is proposed, which learns global uncertainty of inputs by considering only a set of CEs, and this learnt mapper is then applied to unseen or test data. This method has proved to be computationally efficient and reliable on large sets of input.

The generation of GLAM-CLUEs involves two steps: the training stage and the inference step. Firstly there will be a finite set of CEs which will be grouped according to the certainty. The global features of the uncertainty of this set of certain or uncertain points are then learned in the latent space during the training phase.This function is then applied to the unseen test data to generate CEs in the inference step. The primary distinction between GLAM-CLUE and the original CLUE is that with GLAM-CLUE, the training phase of the algorithm can be enhanced by producing CLUE from uncertain properties of the points. As defined before, there will be a mapper or function which maps the uncertain points to certain points by checking on which properties the changes have to be made. This is given by:

$$\mathbf{z}_{certain} = G(\mathbf{z}_{uncertain})$$

Between an uncertain group i to certain group j, there is only a single mapper and this is controlled by parameter $\theta$. This can be represented as:

$$\mathbf{z}_j = G_{i \to j}(\mathbf{z}_i) = \mathbf{z}_i + \theta_{i \to j}$$

where $G$ is the mapper. In the inference step, the mapper is applied on unseen data to make it more certain. The loss function is calculated as:

$$\mathcal{L}\left(\theta \mid Z_{uncertain}, X_{certain}\right) =$$

$$\lambda_\theta \mid\mid \theta \mid\mid_1 + \frac{1}{\mid Z_{uncertain} \mid}$$

$$\sum_{\mathbf{z} \in Z_{uncertain}} \min_{\mathbf{x} \in X_{certain}} \mid\mid \mu_\theta(\mathbf{z} + \theta) - \mathbf{x} \mid\mid_2^2$$

**Performance Test** Multiple performance tests are available and two of them are: Difference Between Means(DBM) and Nearest Neighbours(NN). In DBM, the difference between the means of uncertain and certain input is calculated and this is then added to unseen test data in the inference step.
The nearest neighbors for highly certain data are calculated for the second test. GLAM-CLUE demonstrated in the experiments that it outperforms these tests and is 200 times faster than the current state-of-the-art methods.

**Uncertainty Grouping** As explained in GLAM-CLUE we need to group certain and uncertain inputs. There is a higher bound on the amount of uncertain data when compared to certain data points. So there might exist a many-to-one mapping from uncertain point groups to certain point groups. This is given in Figure 9.

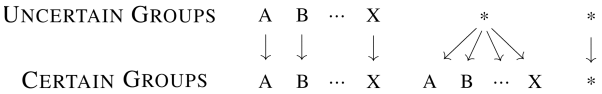


**Figure 9. Uncertain groups to certain groups mapping. Asterisk represent any group.**

The first method shows the mapping for a particular class where uncertain points are mapped to certain points. The method on the far right maps from any uncertain group to any certain group. In the first two cases of mapping, we can select a set of best CEs for the training using some selection method. As this type of selection is not done in generic mapping, there will be a higher number of CEs which will make the mapping harder to train and also might not be diverse.

## 3. Experiments and Results

Different experiments are performed on the UCI Credit Classification data set (Dua, Graff, et al. (2017)), MNIST image data set (LeCun (1998)), and Synbols image data set (Lacoste et al. (2020)). The advantages of the $\delta$-CLUE, $\nabla$-CLUE and GLAM-CLUE are demonstrated via these experiments. The main results and takeaway of these experiments in each CLUE are given in the below section:

### 3.1. $\delta$-CLUE

As given in Figure 4, we can see that as the $\delta$ value increases the distance also increases. This uncertainty-distance trade-off is controlled by tuning the hyper parameter $\lambda_x$. Low uncertainty can be reached with less distance from the original input by reaching an ideal $\lambda_x$ value (Figure 10, right). In Figure 10, the left most image shows that when the number of CLUEs generated increases, the diversity increases as well, but diversity eventually hits saturation at a certain point.

There exists different methods to select the optimum $\delta$ value before the experiments are conducted. Finding the ideal minima in a $\delta$ ball might be challenging at times when the $\delta$ value is low. Therefore, several constrained optimisation techniques are employed in the experiments. As a result, choosing the proper $\delta$ value can be a smart practice. To accomplish this,

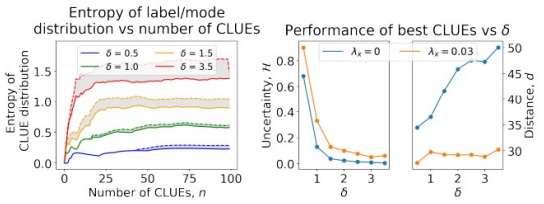


**Figure 10. Left: Entropy of the labels increasing as the number of CLUEs increases in MNIST data set. Right: Performance of CLUEs as $\delta$ increases.**

identify the relationship between the uncertainty and the distance of the data in the latent space, which gives some information about the counterfactuals.

### 3.2. $\nabla$-CLUE

Experiments are conducted on both MNIST and UCI Credit data set with a fixed value for number of counterfactuals, radius and $\delta$ value. Figure 11 is the plot which shows how the different diversity metrics changes when the diversity weight $\lambda_D$ is increased in both latent space and input space. We can see that the diversity also increases monotonically with $\lambda_D$, but it attains a saturation point in diversity with way less counterfactuals than in the $\delta$-CLUE.

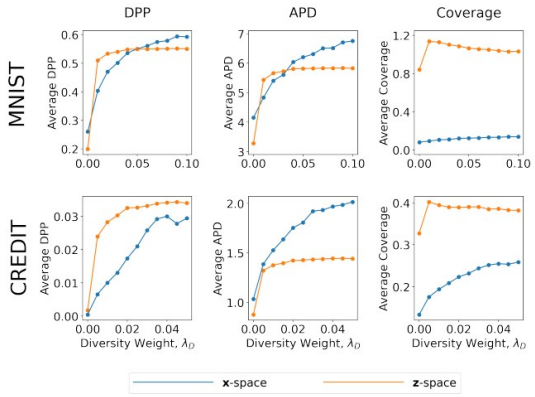


**Figure 11. Diversity weight versus Diversity metrics. Row 1: MNIST data set. Row 2: CREDIT data set. The effect of $\lambda_D$ on different diversity metrics.**

### 3.3. GLAM-CLUE

The computational efficiency of GLAM-CLUE is demonstrated with MNIST data set. There are 3 GLAM-CLUEs which are compared with the first GLAM-CLUE represented as GLAM1, with the mapper learning from all uncertain and certain 4s in the MNIST data set. GLAM2 and GLAM3 learns from all uncertain inputs and and their corresponding CLUEs generated. Although generating CLUEs at training stage requires additional computation, GLAM2 and GLAM3 performed better than GLAM1 because they use CLUEs to train the mapper. The hyper parameter $\lambda_x$ is also trained with different values ($\lambda_x$ = 0 , $\lambda_x$ = 0.03) to control the uncertainty-distance trade-off and the best performed GLAM-CLUE was achieved at $\lambda_x$ = 0.03 when trained on CLUEs as in GLAM2 and GLAM3. The uncertainty, distance and cost comparison of GLAM-CLUE with original CLUEs and other baselines are given in the Figure 12

We can see that GLAM-CLUE outperforms all the baselines and the CLUE itself. Even though the GLAM-CLUE takes extra computational time during the training phase because of the CLUE generation, the average CPU time during inference phase is exceptionally faster, almost 200 times when compared to CLUE. The Figure 13 shows the table which specifies the average CPU time of GLAM-CLUE, CLUE and other baselines in the inference step.

In conclusion, these experiments demonstrates how $\delta$-CLUE deals with the uncertainty-distance trade-off, $\nabla$-CLUE deals with the diversity issue, and GLAM-CLUE generates efficient global uncertainty explanations, thus overcoming CLUE's flaws.

## 4. Related Work

This paper majorly focuses on eliminating the multiplicity and computational efficiency issue of the CLUE algorithm. There are many additional studies that concentrate on the uncertainty explanation, but they all have certain flaws. Joshi et al. (2018) have presented a method for creating counterfactuals using a deep generative model, similar to CLUE, but not for uncertainty explanation. Additionally, Mothilal et al. (2020) and Russell (2019) use linear programs to generate counterfactuals, although they are also not designed to provide uncertainty explanations. These works are all mostly concerned with local aspects of counterfactuals.

Our algorithms place equal emphasis on increasing computational speed and on global explanations. Similar to GLAM-CLUE, there is a method in which a function is devised which maps from lower dimensional group to another group by Plumb et al. (2020) and an advanced deep generative model which generates counterfactuals in an efficient manner by Mahajan et al. (2019) and Yang et al. (2021). DiCE or Diverse Counterfactual Explanations is another algorithm by Mothilal et al. (2020) which is similar

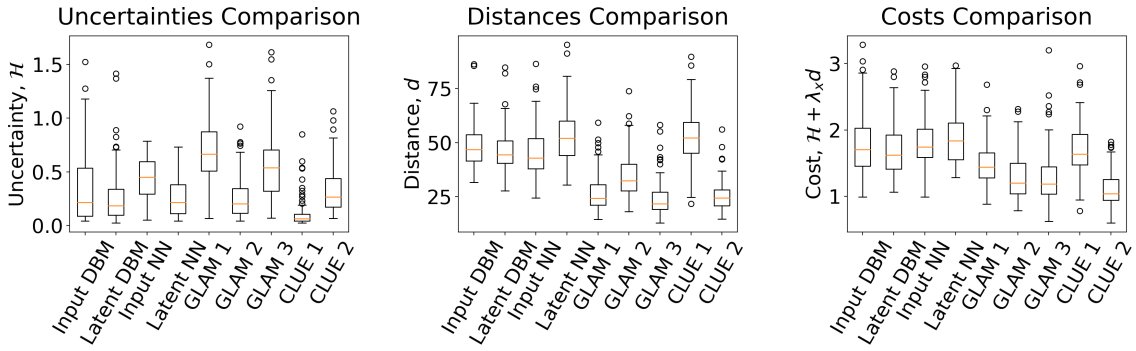


**Figure 12. The comparison of GLAM-CLUE with other base lines in MNIST data set. CLUE 1 and CLUE 2 are generated when $\lambda_x = 0$ and $\lambda_x = 0.03$ respectively. Left: Uncertainty comparison. Centre: Distance comparison. Right: Cost comparison.**

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

**Figure 13. Average CPU time in seconds for inference step of a single counterfactual generation from MNIST data set.**

to the DIVerse-CLUE or $\nabla$-CLUE algorithm, however DiCE was unable to address all of CLUE's drawbacks.

Future applications of the methods described in this research include testing with higher dimensional data sets in the algorithm. The entire potential of CLUE can be explored with higher dimensional data sets. In this paper, we have only used the simple metrics like L1-norm or L2-norm for distance calculations; instead we can use FID scores (Heusel et al. (2017)) to evaluate the performance of the algorithm in a better way. In the experiments conducted in this paper, we have used Variational Auto Encoder (VAEs) as the DGM, but instead we can employ Generative Adversarial Networks or GANs (Goodfellow et al. (2014)) as the DGM to evaluate whether there are any improvements or modifications to the performance when generating counterfactuals using GANs.

GLAM-CLUE operates well, as demonstrated in the studies, however this may not always be the case. Some times GLAM-CLUE fails to differentiate uncertain and certain groups from each other by simple translation as given in Figure 14.

This problem must be resolved by GLAM-CLUE utilizing better techniques to divide these groups, such as a clustering approach or more intricate mapping functions. Additionally, a novel approach is proposed by Dosovitskiy and Djolonga (2020) that use a single

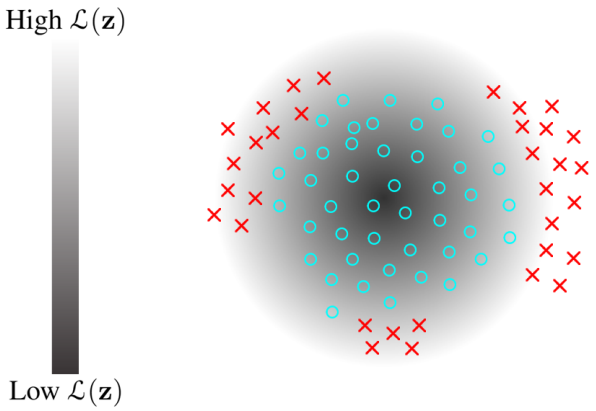


**Figure 14. The shortcomings of GLAM-CLUE is represented in this figure. Red points represent the group of uncertain points and blue points represent the group of certain points. We can see difficulty in mapping uncertain groups to certain groups unless a clustering or similar technique is employed.**

model trained on multiple losses rather than multiple models trained on a single loss can be used to quickly tune the hyper parameter$\lambda_\theta$. New experiments can also be performed on human subjects as this would open door to a wide possibility of improvements in the CLUE algorithm.

## 5. Conclusion

Most practitioners will be interested in knowing the reason behind the uncertainty in prediction and the features that lead to this uncertainty. If we can provide an explanation for this uncertainty, then it would be a great advantage in the world of machine learning and model predictions. CLUE is such a novel method that can be used for uncertainty explanations. To overcome

the shortcomings of CLUE, three enhanced CLUEs are introduced in this report.

To deal with the multiplicity issue of CLUE, the $\delta$-CLUE is introduced, which generates multiple counterfactuals within the $\delta$ ball distance in latent space. As the $\delta$-CLUE can generate redundant CLUEs, the more optimised DIVerse CLUE or $\nabla$-CLUE generates only diverse and distinct CLUEs and is optimised using different diversity metrics. The $\nabla$-CLUE demands for more calculation time because it takes several iterations to get to an optimized state. Finally, to solve this computational inefficiency, GLAM-CLUE is proposed which learns from the group of uncertain and certain inputs in an amortised single function call to generate more certain inputs.

The GLAM-CLUE is demonstrated through experiments to be computationally more efficient than other CLUEs and baselines. It also shows that $\delta$-CLUE and $\nabla$-CLUE serve their respective functions in dealing with the multiplicity and redundancy issues. As a result, these kind of uncertainty explanations can be used as a precedent for model explanations in the real world machine learning problems.

## References


Antoran, J., Bhatt, U., Adel, T., Weller, A., & Hernández-Lobato, J. M. (2021). Getting a {clue}: A method for explaining uncertainty estimates. *International Conference on Learning Representations*. https://openreview.net/forum?id=XSLF1XFq5h

Bhatt, U., Chien, I., Zafar, M. B., & Weller, A. (2021). Divine: Diverse influential training points for data visualization and model refinement. *ArXiv*, *abs/2107.05978*.

Dua, D., Graff, C. et al. (2017). Uci machine learning repository.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, *27*.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, *30*.

Joshi, S., Koyejo, O., Kim, B., & Ghosh, J. (2018). Xgems: Generating examplars to explain black-box models. *arXiv preprint arXiv:1806.08867*.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Lacoste, A., Rodrıguez López, P., Branchaud-Charron, F., Atighehchian, P., Caccia, M., Laradji, I. H., Drouin, A., Craddock, M., Charlin, L., & Vázquez, D. (2020). Synbols: Probing learning algorithms with synthetic datasets. *Advances in Neural Information Processing Systems*, *33*, 134–146.

LeCun, Y. (1998). The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

Ley, D., Bhatt, U., & Weller, A. (2021). Diverse, global and amortised counterfactual explanations for uncertainty estimates. *arXiv e-prints*, arXiv–2112.

MacKay, D. J. (1992). A practical bayesian framework for backpropagation networks. *Neural computation*, *4*(3), 448–472.

Mahajan, D., Tan, C., & Sharma, A. (2019). Preserving causal constraints in counterfactual explanations for machine learning classifiers. *arXiv preprint arXiv:1912.03277*.

Mothilal, R. K., Sharma, A., & Tan, C. (2020). Explaining machine learning classifiers through diverse counterfactual explanations. *Proceedings of the 2020 conference on fairness, accountability, and transparency*, 607–617.

Plumb, G., Terhorst, J., Sankararaman, S., & Talwalkar, A. (2020). Explaining groups of points in low-dimensional representations. *International Conference on Machine Learning*, 7762–7771.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 1135–1144.

Russell, C. (2019). Efficient search for diverse coherent explanations. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 20–28.

Yang, F., Alva, S. S., Chen, J., & Hu, X. (2021). Model-based counterfactual synthesizer for interpretation. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1964–1974.