Boundary-Aware Uncertainty for Feature Attribution Explainers

Davin Hill
Northeastern University
dhill@ece.neu.edu

Aria Masoomi
Northeastern University
masoomi.a@northeastern.edu

Max Torop
Northeastern University
torop.m@northeastern.edu

Sandesh Ghimire
Northeastern University
drsandeshghimire@gmail.com

Jennifer Dy
Northeastern University
jdy@ece.neu.edu

Abstract

Post-hoc explanation methods have become a critical tool for understanding black-box classifiers in high-stakes applications. However, high-performing classifiers are often highly nonlinear and can exhibit complex behavior around the decision boundary, leading to brittle or misleading local explanations. Therefore there is an impending need to quantify the uncertainty of such explanation methods in order to understand when explanations are trustworthy. In this work we propose the **G**aussian **P**rocess **E**xplanation Un**C**ertainty (GPEC) framework, which generates a unified uncertainty estimate combining decision boundary-aware uncertainty with explanation function approximation uncertainty. We introduce a novel geodesic-based kernel, which captures the complexity of the target black-box decision boundary. We show theoretically that the proposed kernel similarity increases with decision boundary complexity. The proposed framework is highly flexible; it can be used with any black-box classifier and feature attribution method. Empirical results on multiple tabular and image datasets show that the GPEC uncertainty estimate improves understanding of explanations as compared to existing methods.

Proceedings of the 27[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Copyright 2024 by the author(s).

1 INTRODUCTION

Post-hoc explainability methods have become a crucial tool for understanding and diagnosing their black-box model predictions. Recently, many such *explainers* have been introduced in the category of local feature attribution methods; that is, methods that return a real-valued score representing each feature's relative importance for the model prediction. These explainers are *local* in that they are not limited to using the same decision rules throughout the data distribution, therefore they are better able to represent nonlinear and complex black-box models.

However, recent works have shown that local explainers can be inconsistent or unstable. For example, explainers may yield highly dissimilar explanations for similar samples (Alvarez-Melis and Jaakkola, 2018; Khan et al., 2023), exhibit sensitivity to imperceptible perturbations (Dombrowski et al., 2019; Ghorbani et al., 2019; Slack et al., 2020), or lack stability under repeated application (Visani et al., 2022). When working in high-stakes applications, it is imperative to provide the user with an understanding of whether an explanation is reliable, potentially problematic, or even misleading. A way to guide users regarding an explainer's reliability is to provide corresponding *uncertainty quantification* estimates.

One can consider explainers as function approximators; as such, standard techniques for quantifying the uncertainty of estimators can be utilized to quantify the uncertainty of explainers. This is the strategy utilized by existing methods that estimate explainer uncertainty (e.g. (Slack et al., 2021; Schwab and Karlen, 2019)). However, we observe that for explainers, this is not sufficient; in addition to uncertainty due to the *function approximation* of explainers, explainers also have to deal with the uncertainty due to the complexity of the *decision boundary (DB)* of the black-box

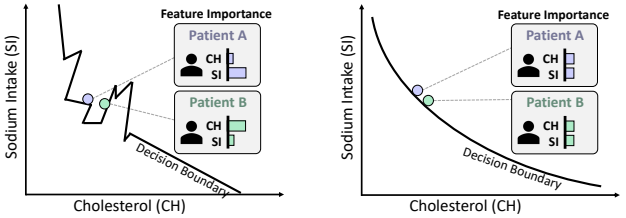

Figure 1: Illustrative example of potential pitfalls when relying on local explainers for samples near complex regions of the decision boundary (left) as compared with a smoothed decision boundary (right).

model in the local region being explained.

Previous works investigating DB geometry have related higher DB complexity to increased model generalization error (Valle-Perez et al., 2019) and increased adversarial vulnerability (Moosavi-Dezfooli et al., 2019; Fawzi et al., 2018). Smoother DBs have been shown to improve feature attributions (Wang et al., 2020) and produce more consistent counterfactual explanations (Black et al., 2022). Dombrowski et al. (2019) show that, in ReLU networks, samples with similar predictions can yield widely disparate explanations, which can be regulated through model smoothing. Consider the following example (Fig. 1): a prediction model is used for a medical diagnosis using two features: cholesterol level and sodium intake. We use the gradient with respect to each feature as an estimate of feature importance. Patients A and B have similar cholesterol and sodium levels and receive the same prediction, however, the complex decision boundary (left) results in a different top feature for each patient. In contrast, the smoothed decision boundary (right) yields more consistent explanations.

We approach this problem from the perspective of similarity: given two samples and their respective explanations, how closely related should the explanations be? From the previous intuition, we define this similarity based on a geometric perspective of the DB complexity between these two points. Specifically, we propose the novel Weighted Exponential Geodesic (WEG) kernel, which encodes our expectation that two samples close in Euclidean space may not actually be similar if the DB within a local neighborhood of the samples is highly complex.

Using this similarity formulation, we propose the **G**aussian **P**rocess **E**xplanation Un**C**ertainty (GPEC) framework (Fig. 2), which is an instance-wise, model-agnostic, and explainer-agnostic method to quantify the explanation uncertainty. The proposed notion of uncertainty is complementary to existing quantification methods. Existing methods primarily estimate the uncertainty related to the choice in model parameters and fitting the explainer, which we call *function approximation uncertainty*, and does not capture uncertainty related to DB complexity. GPEC can combine the DB-based uncertainty with function approximation uncertainty derived from any local feature attribution method.

In summary, we make the following contributions:

- We introduce a novel geometric perspective on capturing explanation uncertainty and define a geodesic-based similarity between explanations. We prove theoretically that the proposed similarity captures the complexity of the decision boundary from a given black-box classifier.
- We propose a novel Gaussian Process-based framework that combines 1) uncertainty from decision boundary complexity and 2) explainer-specific function approximation uncertainty to generate uncertainty estimates for any given feature attribution method and black box model.
- Empirical results show GPEC uncertainty improves understanding of feature attribution methods.

2 RELATED WORKS

Explanation Methods. A variety of methods have been proposed for improving the transparency of pretrained black-box prediction models (Guidotti et al., 2018; Barredo Arrieta et al., 2020). Within this category of *post-hoc* methods, many methods focus on *local* explanations, that is, explaining individual predictions rather than the entire model. Some of these methods implement local *feature selection* (Chen et al., 2018; Masoomi et al., 2020); others return a real-valued score for each feature, termed *feature attribution* methods, which are the primary focus of this work. For example, LIME (Ribeiro et al., 2016) trains a local linear regression model to approximate the black-box model. Lundberg and Lee (2017) generalizes LIME and five other feature attribution methods using the SHAP framework, which fulfill a number of desirable axioms. While LIME and SHAP are model-agnostic, others are model-specific, such as neural networks (Bach et al., 2015; Shrikumar et al., 2017; Sundararajan et al., 2017; Erion et al., 2021), tree ensembles (Lundberg et al., 2020), or Bayesian neural networks (Bykov et al., 2020). Another class of methods involves training surrogate models to explain the black-box model (Dabkowski and Gal, 2017; Chen et al., 2018; Schwab and Karlen, 2019; Guo et al., 2018; Jethani et al., 2022).

Explanation Uncertainty. One option for improving explainer trustworthiness is to quantify their associated uncertainty. Bootstrap resampling techniques have been proposed to estimate uncertainty from

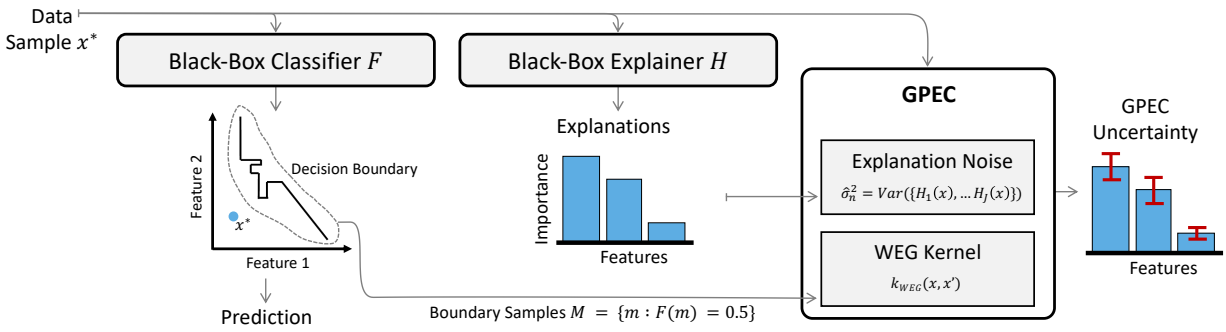

Figure 2: Overview of the GPEC framework. GPEC takes samples from the classifier's decision boundary plus (possibly noisy) explanations and fits a GP model with the novel WEG Kernel. The GPEC estimate incorporates both the uncertainty derived from the decision boundary complexity and also the explanation approximation uncertainty from the explainer.

surrogate-based explainers (Schwab and Karlen, 2019; Schulz et al., 2022). Guo et al. (2018) also proposes a surrogate explainer parameterized with a Bayesian mixture model. Alternatively, Bykov et al. (2020) and Patro et al. (2019) introduce methods for explaining Bayesian neural networks, which can be transferred to their non-Bayesian counterparts. Covert and Lee (2021) derive an unbiased version of KernelSHAP and investigates an efficient way of estimating its uncertainty. Zhang et al. (2019) categorizes different sources of variance in LIME estimates. Several methods also investigate LIME and KernelSHAP in a Bayesian context; for example, calculating a posterior over attributions (Slack et al., 2021), investigating priors for explanations (Zhao et al., 2021), or using active learning during sampling (Saini and Prasad, 2022).

However, existing methods for quantifying explanation uncertainty only consider the uncertainty of the explainer as a function approximator. This work introduces an additional notion of uncertainty for explainers that considers the complexity of the classifier DB.

3 UNCERTAINTY FRAMEWORK FOR EXPLAINERS

We now outline the GPEC framework (Fig. 2), which is parametrized with a Gaussian Process (GP) regression model[1]. Consider a sample $x^* \in \mathcal{X}$ that we want to explain in the context of a black-box classifier $F : \mathcal{X} \to [0, 1]$, where $\mathcal{X} \subseteq \mathbb{R}^D$ is the data space and D is the number of features. For convenience we consider the binary classification case; this is extended to multiclass in App. C. We apply a local feature attribution explainer $H : \mathcal{X} \to \mathbb{R}^D$.

Recent works (e.g. Alvarez-Melis and Jaakkola (2018); Dombrowski et al. (2019)) have shown that local explanations can lack robustness and stability related to model complexity. Therefore, when explaining samples in high-stakes applications, it is critical to understand the behavior of the explainer, especially in relation to other samples near x^*. More concretely, let $X \in \mathbb{R}^{N \times D}$ represent a dataset of N samples. Here, each row vector $X_n \in \mathbb{R}^D$, $n \in N$ represents a data point. We apply H to the rows of X generating N observed explanations, $E_n \in \mathbb{R}^D$, $n \in N$, which are grouped into $E \in \mathbb{R}^{N \times D}$. We can use these observed sample-explanation pairs to infer the behavior of H around x^*, however there are two main challenges. First, we expect the similarity between the explanations of X and x^* to be dependent on F. In particular, we expect that as the DB in a neighborhood around x^* and a given sample X_n becomes increasingly complex, $H(x^*)$ and $H(X_n)$ may become more dissimilar; i.e. $H(X_n)$ may not contain useful information towards inferring $H(x^*)$. In this situation, the user should be prompted to either draw additional samples near x^*, or otherwise be warned of higher uncertainty. Second, the observed explanations E can be noisy; many explainers are stochastic and approximated with sampling methods or a learned function.

To solve these challenges, we can model the explainer with a vector-valued GP regression by treating the explainer as a latent function inferred using samples X and explanations E. We model each explanation E_n as being generated from a latent function $\mathcal{H}$ plus independent Gaussian noise η_n. For convenience, we consider each feature d independently; see App. C for extensions.

$$E_{n,d} = \mathcal{H}_d(X_n) + \eta_{n,d} \;\; s.t. \quad \underbrace{\mathcal{H}_d(X_n) \sim \mathcal{GP}(0, k(\cdot, \cdot))}_{\text{Decision Boundary-Aware Uncertainty}} \tag{1}$$

$$s.t. \quad \underbrace{\eta_{n,d} \sim \mathcal{N}(0, \sigma^2_{n,d})}_{\text{Function Approximation Uncertainty}}$$

where $k(\cdot, \cdot)$ is the specified kernel function for the GP

[1] A brief review of GP regression is provided in App. B.

prior. We disentangle each explanation into two components, $\mathcal{H}(X_n)$ and η_n, which represent two separate sources of uncertainty: 1) a *decision boundary-aware* uncertainty which we capture using the kernel similarity, and 2) a *function approximation* uncertainty from the explainer. After specifying $\mathcal{H}(X_n)$ and η_n, we can combine the two sources by calculating the predictive distribution for x^*. We take the variance of this distribution as the GPEC uncertainty estimate:

$$\mathbb{V}_d[x^*] = k(x^*, x^*) - k(X, x^*)^\intercal [K + \sigma_d^2 I_N]^{-1} k(X, x^*) \tag{2}$$

where $K \in \mathbb{R}^{N \times N}$ is the kernel matrix s.t. $K_{ij} = k(X_i, X_j)\ \forall i, j \in \{1...N\}$, $k(X, x^*) \in \mathbb{R}^{N \times 1}$ has elements $k(X, x^*)_i = k(X_i, x^*)\ i \in \{1...N\}$, $\sigma_d^2 \in \mathbb{R}_+^N$ is the variance parameter for explanation noise, and I_N is the identity matrix. From Eq. (2) we see that predictive variance captures DB-aware uncertainty through the kernel function $k(\cdot, \cdot)$, and also the function approximation uncertainty through the $\sigma_d^2 I_N$ term.

Function Approximation Uncertainty. The η_n component of Eq. (1) represents the uncertainty stemming from explainer estimation. For example, η_n can represent the variance due to sampling (e.g. perturbation-based explainers) or explainer training (e.g. surrogate-based explainers). Explainers that include some estimate of uncertainty (e.g. BayesLIME, BayesSHAP, CXPlain) can be directly used to estimate σ_n^2. For other stochastic explanation methods, we can estimate σ_n^2 empirically by resampling J explanations for the same sample X_n:

$$\hat{\sigma}_n^2 = \frac{1}{|J|} \sum_{i=1}^{J} \left(H_i(X_n) - \frac{1}{|J|} \sum_{j=1}^{J} H_j(X_n) \right)^2 \tag{3}$$

where each $H_i(X_n)$ is a sampled explanation. Alternatively, for deterministic explanation methods we can omit the η_n term and assume noiseless explanations.

Decision Boundary-Aware Uncertainty. In contrast, the $\mathcal{H}(X_n)$ component of Eq. (1) represents the distribution of functions that could have generated the observed explanations. The choice of kernel $k(\cdot, \cdot)$ encodes our *a priori* assumption regarding the similarity between explanations based on the similarity of their corresponding inputs. In other words, given two samples $x, x' \in \mathcal{X}$, how much information do we expect a given explanation $H(x)$ to provide for a nearby explanation $H(x')$? As the DB between $H(x)$ and $H(x')$ becomes more complex, we would expect for this information to decrease. In Section 4, we consider a novel kernel formulation that reflects the complexity of the DB in a local neighborhood of the samples.

4 WEG KERNEL

Intuitively, the GP kernel encodes the assumption that each explanation provides some information about other nearby explanations, which is defined through kernel similarity. To capture boundary-aware uncertainty, we want to define a similarity $k(x, x')$ that is inversely related to the complexity or smoothness of the DB between $x, x' \in \mathcal{X}$.

4.1 Geometry of the Decision Boundary

We represent the DB as a hypersurface embedded in $\mathbb{R}^D$ with co-dimension one. Given the classifier F, we define the DB[2] as $\mathcal{M}_F = \{m \in \mathbb{R}^D : F(m) = \frac{1}{2}\}$. For any two points $m, m' \in \mathcal{M}_F$, let $\gamma : [0, 1] \to \mathcal{M}_F$ be a differentiable map such that $\gamma(0) = m$ and $\gamma(1) = m'$, representing a 1-dimensional curve on $\mathcal{M}_F$. We can then define distances along the DB as geodesic distances in $\mathcal{M}_F$ (Fig 3A):

$$d_{geo}(m, m') = \min_{\gamma} \int_0^1 ||\dot{\gamma}(t)|| \mathrm{d}t \quad \forall m, m' \in \mathcal{M}_F \tag{4}$$

The relative complexity of the DB can be characterized by the geodesic distances between points on the DB. For example, the simplest form that the DB can take is a linear boundary. Consider a black-box model with linear DB $\mathcal{M}_1$. For two points $z, z' \in \mathcal{M}_1$, $d_{geo}(z, z') = ||z - z'||_2$ which corresponds with the minimum geodesic distance in the ambient space. For any nonlinear DB $\mathcal{M}_2$ that also contains z, z', it follows that $d_{geo}(z, z') > ||z - z'||_2$. As the complexity of the DB increases, there is a general corresponding increase in geodesic distances between fixed points on the DB. We can adapt geodesic distance in our kernel selection through the exponential geodesic (EG) kernel (Feragen et al., 2015).

$$k_{\text{EG}}(m, m') = \exp\left[-\lambda d_{geo}(m, m')\right] \tag{5}$$

The EG kernel has been previously investigated in the context of Riemannian manifolds (Feragen et al., 2015; Feragen and Hauberg, 2016). In particular, while prior work shows that the EG kernel fails to be positive definite for all values of λ in non-Euclidean space, there exists large intervals of $\lambda > 0$ for which the EG kernel is positive definite. Appropriate values can be selected through grid search and cross validation; we assume that a valid value of λ has been selected.

Therefore, by sampling $\mathcal{M}_F$, we can use the EG kernel matrix to capture DB complexity. However, a challenge remains in relating points $x, x' \in \mathcal{X} \setminus \mathcal{M}_F$ to the nearby DB. In Section 4.2 we consider a continuous weighting over $\mathcal{M}_F$ based on distance to x, x'.

[2]Without loss of generality, we assume that the classifier decision rule is $\frac{1}{2}$

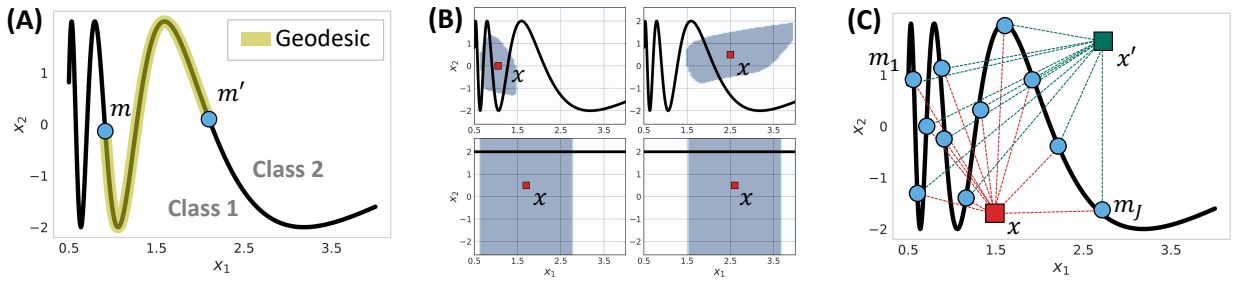

Figure 3: Consider a classifier with DB defined as $\mathcal{M}_0 = \{(x_1, f(x_1)) : x_1 \in \mathbb{R}_{>0}\}$ where $f(x_1) = 2\cos(\frac{10}{x_1})$. **(A)** Illustration of geodesic distance $d_{geo}(m, m')$ between two points $m', m \in \mathcal{M}_0$. **(B)** Evaluation of the WEG kernel for $\mathcal{M}_0$ (top) and a linear DB (below). The gray region highlights the set $\{x' : k(x, x') \geq 0.9\}$ for a given x (red). This region increases as the local DB become more linear. **(C)** During WEG approximation, we calculate Euclidean distances between x, x' (red, green) and DB samples $m_1, ..., m_J \in \mathcal{M}_0$ (blue). When appropriately normalized (Eq. (6)), this acts as a weighting for each element of the EG kernel.

4.2 Weighting Decision Boundary Samples

Let $p(m)$ denote a distribution with support defined over $\mathcal{M}_F$ such that we can draw DB samples $m_1 ... m_J \sim p(m)$ using a DB sampling algorithm (see Sec. 4.4). We weight $p(m)$ according to the ℓ_2 norm between m and a fixed data sample x to create a weighted distribution $q(m|x, \rho)$:

$$q(m|x, \rho) \propto \exp\left[-\rho||x - m||_2^2\right] p(m) \tag{6}$$

where ρ represents a hyperpameter that controls the sensitivity of the weighting. We can then define the kernel function $k_{\text{WEG}}(x, x')$ by taking the expected value over the weighted distributions.

$$k_{\text{WEG}}(x, x') = \int\int k_{\text{EG}}(m, m') \times q(m|x, \rho)\, q(m'|x', \rho)\, \mathrm{d}m\mathrm{d}m' \tag{7}$$

$$= \frac{1}{Z_m Z_{m'}} \int\int \exp\left[-\lambda d_{geo}(m, m')\right] \times \exp\left[-\rho(||x - m||_2^2 + ||x' - m'||_2^2)\right] p(m)p(m')\mathrm{d}m\mathrm{d}m' \tag{8}$$

where $Z_m, Z_{m'}$ are normalizing constants for $q(m|x, \rho)$ and $q(m'|x', \rho)$, respectively. Eq. (8) is an example of a marginalized kernel (Tsuda et al., 2002): a kernel defined by the expected value of observed samples x, x' over latent variables m, m'. Given that the underlying EG kernel is positive definite, it follows that the WEG kernel forms a valid kernel.

With the WEG kernel, we can calculate a similarity between $x, x' \in \mathcal{X}$ that decreases as the complexity of the DB segments between the two points increases. In Fig. 3B we evaluate the WEG kernel similarity on nonlinear and a linear DB. We observe that WEG similarity reflects the complexity of the DB; as the decision boundary becomes more linear in a local region, the similarity between neighboring points increases. To evaluate the WEG kernel theoretically, we consider two properties. Theorem 1 establishes that the EG kernel is a special case of the WEG kernel for when $x, x' \in \mathcal{X} \cap \mathcal{M}_F$.

Theorem 1. *Given two points $x, x' \in \mathcal{X} \cap \mathcal{M}_F$, then* $\lim_{\rho\to\infty} k_{WEG}(x, x') = k_{EG}(x, x')$

Proof details are shown in App. C.1. Intuitively, as ρ increases the manifold distribution closest to the points x, x' becomes weighted increasingly heavily. At the limit, the weighting concentrates entirely on x, x' themselves, which recovers the EG kernel. Therefore we see that the WEG kernel is a generalization of the EG kernel with a weighting controlled by ρ.

Theorem 2 establishes the inverse relationship between DB complexity and WEG similarity. Given a classifier with a piecewise linear DB, we show that this DB represents a local maximum with respect to WEG kernel similarity; i.e. as we perturb the DB to be nonlinear, kernel similarity decreases. We first define *perturbations* on the DB. Note that $\text{int}(S)$ indicates the interior of a set S and id indicates the identity mapping.

Definition 1 (Manifold Perturbation)**.** Let $\{U_\alpha\}_{\alpha\in I}$ be charts of an atlas for a manifold $\mathcal{P} \subset \mathbb{R}^D$, where I is a set of indices. Let $\mathcal{P}$ and $\widetilde{\mathcal{P}}$ be differentiable manifolds embedded in $\mathbb{R}^D$, where $\mathcal{P}$ is a Piecewise Linear manifold. Let $R : \mathcal{P} \to \widetilde{\mathcal{P}}$ be a diffeomorphism. We say $\widetilde{\mathcal{P}}$ is a *perturbation* of $\mathcal{P}$ on the i^{th} chart if R satisfies the following two conditions: ① There exists a compact subset $K_i \subset U_i$ s.t. $R|_{\mathcal{P}\setminus\text{int}(K_i)} = \text{id}|_{\mathcal{P}\setminus\text{int}(K_i)}$ and $R|_{\text{int}(K_i)} \neq \text{id}|_{\text{int}(K_i)}$. ② There exists a linear homeomorphism between an open subset $\widetilde{U_i} \subseteq U_i$ with $\mathbb{R}^{d-1}$ which contains K_i.

Theorem 2. *Let $\mathcal{P}$ be a (d–1)-dimension Piecewise Linear manifold embedded in $\mathbb{R}^D$. Let $\widetilde{\mathcal{P}}$ be a perturbation of $\mathcal{P}$ and define $\tilde{k}(x, x')$ and $k(x, x')$ as the*

WEG kernels defined on $\widetilde{\mathcal{P}}$ and $\mathcal{P}$ respectively. Then $\tilde{k}(x,x') < k(x,x')\ \forall x,x' \in \mathbb{R}^D$.

Proof details are shown in App. C.2. Theorem 2 implies that, for any two fixed points x, x', their kernel similarity $k_{\text{WEG}}(x,x')$ decreases as the black-box DB complexity increases. Within GPEC, the explanations for x, x' become less informative for other nearby explanations and induce a higher explanation uncertainty estimate.

To improve the WEG kernel interpretation, we can normalize k_{WEG} to scale similarity values to be between $[0,1]$. We define the normalized kernel k^*_{WEG}:

$$k^*_{\text{WEG}}(x,x') = \frac{k_{\text{WEG}}(x,x')}{\sqrt{k_{\text{WEG}}(x,x)k_{\text{WEG}}(x',x')}} \tag{9}$$

4.3 WEG Kernel Approximation

In practice, the integrals in Eq. (8) are intractable; we can instead use Monte Carlo integration to approximate $k_{\text{WEG}}(x,x')$ with J samples $m_1, ..., m_J \sim p(m)$.

$$k_{\text{WEG}}(x,x') \approx \frac{1}{Z_m Z_{m'} J^2} \sum_{i=1}^{J}\sum_{j=1}^{J} \exp[-\lambda d_{geo}(m_i,m_j)] \times \exp[-\rho(||x-m_i||_2^2 + ||x'-m_j||_2^2)] \tag{10}$$

We can similarly estimate constants Z_m, $Z_{m'}$:

$$Z_m \approx \frac{1}{J}\sum_{i=1}^{J} \exp\left[-\rho||x-m_i||_2^2\right] \tag{11}$$

4.4 GPEC Algorithm

GPEC has separate training (Alg. 1) and inference (Alg. 2) stages. During training, GPEC constructs the EG kernel matrix by sampling the DB. Note that in Eq. (10) we calculate $k_{\text{EG}}(m_i,m_j)$ independently of x and x' $\forall i,j \in \{1...J\}$. Therefore, the EG kernel only needs to be calculated once for a set of DB samples. During training and inference, the WEG kernel weights the precalculated EG kernel based on x, x' (Fig 3C). Once the GPEC model is trained, either the variance or confidence interval width of the predictive distribution can be used as the uncertainty estimate. The training cost of GPEC is amortized during inference; GP inference generally has time complexity of $\mathcal{O}(N^3)$, which can be reduced to $\mathcal{O}(N^2)$ using BBMM (Gardner et al., 2018), and further with variational methods (e.g., Hensman et al. (2015)).

DB sampling and geodesic distance estimation are ongoing areas of research. In our implementation, we adapt DeepDIG (Karimi et al., 2019) for sampling the DB of neural networks and DBPS (Yan and Xu, 2008) for all other models. We utilize ISOMAP (Tenenbaum et al., 2000) for estimating geodesic distances. Additional implementation detail is provided in App. D.

Algorithm 1 GPEC Training

Input : Training Samples $X \in \mathbb{R}^{N\times D}$, Explainer.
Output : WEG Kernel $K \in [0,1]^{N\times N}$, Explainer Variance $U \in \mathbb{R}_+^{N\times D}$, Weighting $W \in [0,1]^{N\times J}$, EG Kernel $G \in [0,1]^{J\times J}$, DB Samples $M \in \mathbb{R}^{J\times D}$.

Get Explainer Variance U from Explainer
Draw J DB samples M from DB Sampling Function
for *each pair of DB samples* m_i, $m_j \in M$ **do**
  $G_{i,j} \leftarrow \exp(-\lambda d_{geo}(m_i,m_j))$ \\ Eq. (5)
end
for *each data sample* x_i *and DB sample* m_j **do**
  $W_{i,j} \leftarrow \exp(-\rho||x_i - m_j||_2^2)$ \\ Eq. (6)
end
$W_{i,:} \leftarrow \frac{W_{i,:}}{\sum_{j=1}^{P} W_{i,j}}$ \\ Normalize weighting
$K \leftarrow WGW^\intercal$ \\ WEG Kernel
Return K, U, W, G, M

Algorithm 2 GPEC Inference

Input : Sample $x \in \mathbb{R}^D$; K, U, W, G, M from Alg. 1.
Output : GPEC Uncertainty $V \in \mathbb{R}_+^D$

\\ Calculate weighting Eq. (6)
for *each DB sample* $m_i \in M$ **do**
  $W_i^* \leftarrow \exp(-\rho||x - m_i||_2^2)$
end
$W^* \leftarrow \frac{W^*}{\sum_{i=1}^{P} W_i}$
for *each explanation dimension* $d \in D$ **do**
  $V_d = W^*GW^{*\intercal}$
  $- W^*GW^\intercal[K + I_N U_{:,d}]^{-1}W^\intercal GW^*$
end
Return V

5 EXPERIMENTS

We evaluate GPEC on a variety of datasets and classifiers. In section 5.2 we visually compare GPEC uncertainty with competing models. Section 5.3 evaluates how GPEC captures DB complexity. Section 5.4 is an ablation test that disentangles the two sources of uncertainty. All experiments were run on an internal cluster using AMD EPYC 7302 16-Core processors. CIFAR10 results were run on Nvidia A100 GPUs. All source code is available at `https://github.com/davinhill/GPEC`.

5.1 Experiment Setup

Unless otherwise stated, we set $\lambda = 1.0$ and $\rho = 0.1$ (see App. F.4 for experiments on parameter sensitivity), and use GPEC with the KernelSHAP explainer.

Datasets. Experiments are performed on three tabular datasets (Census, Online Shoppers (Sakar et al.,

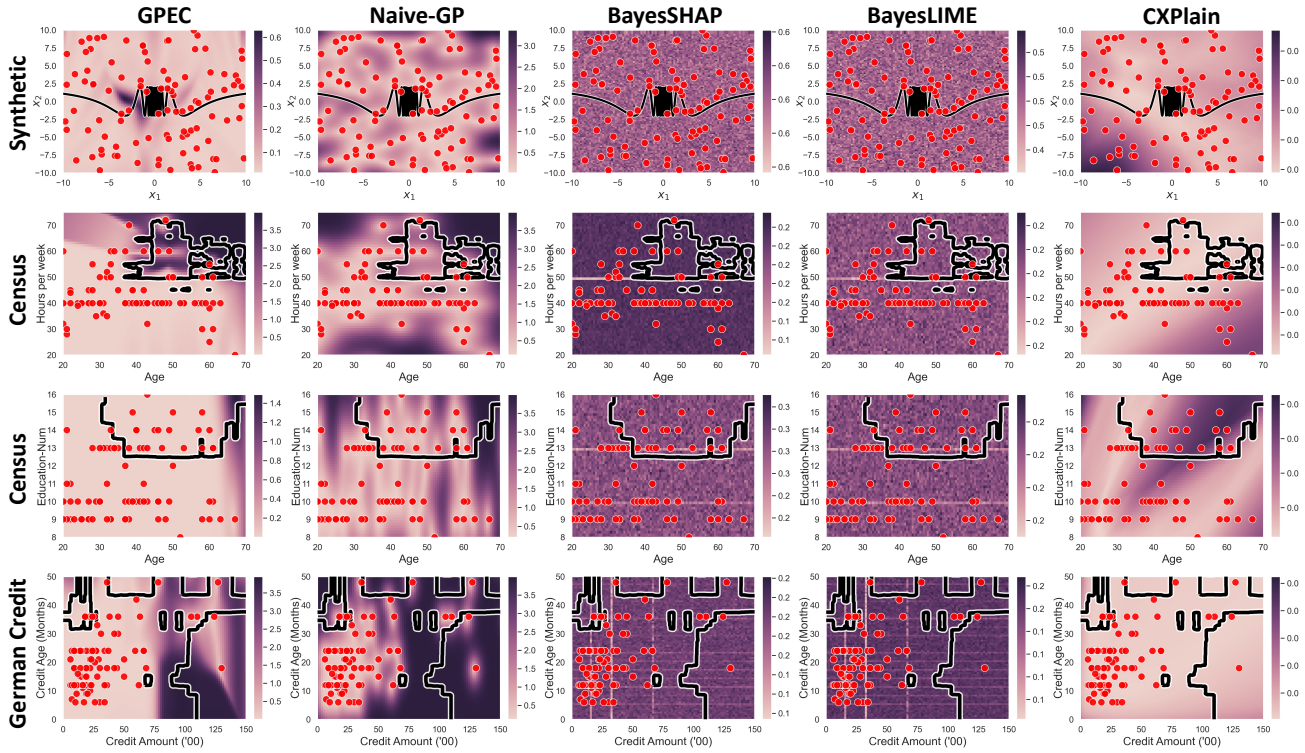

Figure 4: Visualization of estimated explanation uncertainty for different datasets and competing methods. The heatmap represents uncertainty level for a grid of explanations for the x-axis feature; *darker heatmap regions represent higher uncertainty.* The black line represents the black-box DB, and red points represent training samples. The heatmap shows that GPEC uncertainty is elevated for samples near complex decision boundaries. In contrast, heatmaps for BayesSHAP, BayesLIME, and CXPlain are relatively uniform.

2019), German Credit) from the UCI data repository (Dua and Graff, 2017), and three image datasets (MNIST (LeCun and Cortes, 2010), f-MNIST (Xiao et al., 2017)), and CIFAR10 (Krizhevsky et al., 2009)). We additionally create a toy example (Synthetic) where training and test samples are drawn from the uniform distribution over $[-10, 10]^2$ and the model DB is defined as follows:

$$\mathcal{M}_{\text{synth}} = \{(x_1, f(x_1)) : x_1 \in \mathbb{R}\}$$
$$f(x_1) = \begin{cases} 2\cos(\frac{10}{x_1}) & |x_1| \geq \frac{20}{(5e6+1)\pi} \\ 0 & |x_1| < \frac{20}{(5e6+1)\pi} \end{cases}$$

GPEC can be used with any black-box model; we use XGBoost (Chen and Guestrin, 2016) for tabular datasets, 4-layer neural network for MNIST and f-MNIST, and Resnet18 (He et al., 2015) for CIFAR10. Additional dataset details are outlined in App. E.1.

Comparisons. We compare GPEC to a baseline GP implementation plus three other competing explanation uncertainty estimation methods. Naive-GP similarly uses a GP parametrization (Eq. (1)) but instead uses the Radial Basis Function kernel, which does not incorporate DB information. BayesSHAP and BayesLIME (Slack et al., 2021) are extensions of KernelSHAP and LIME, respectively, that fit local Bayesian linear regression models. CXPlain (Schwab and Karlen, 2019) trains a surrogate model and uses bootstrapping to estimate explanation uncertainty. Additional details on competing methods are outlined in App. E.2.

5.2 Uncertainty Visualization

To visualize explanation uncertainty, we plot uncertainty estimates as a heatmap for the explanations derived from an XGBoost binary classifier trained on two selected features. Figure 4 plots the uncertainty heatmap for the x-axis feature (y-axis feature results shown in App. F.7), where *darker heatmap regions indicate higher uncertainty.* Red points represent training samples for GPEC, Naive-GP, and CXPlain, and represent background samples for BayesSHAP and BayesLIME. The DB is plotted as a black line.

We expect to see higher GPEC uncertainty (dark heatmap regions) for test samples farther away from training samples (red) and close to nonlinearities in the DB. We observe that this holds true, especially for high uncertainty regions in the center of Synthetic (top

Dataset	**Census**			**Online Shoppers**			**German Credit**			**CIFAR10**					
Regularization	γ			γ			γ			ℓ_2			Softplus β		
Magnitude	0	5	10	0	5	10	0	5	10	0	1e-5	10e-5	1.0	0.5	0.25
GPEC	**1.573**	**1.177**	**1.158**	**0.209**	**0.123**	**0.092**	**2.665**	**1.747**	**0.250**	**0.029**	**0.029**	**0.028**	**0.033**	**0.032**	**0.032**
Naive-GP	**0.498**	**0.472**	**0.467**	3.699	3.724	3.730	0.330	0.412	2.533	0.902	0.902	0.902	0.903	0.903	0.903
BayesSHAP	0.037	0.037	0.037	0.031	0.031	0.031	0.019	0.019	0.018	–	–	–	–	–	–
BayesLIME	**0.097**	**0.096**	**0.095**	**0.098**	**0.097**	**0.093**	**0.085**	**0.066**	**0.045**	–	–	–	–	–	–
CXPlain	0.064	0.064	0.069	0.004	0.003	0.006	1.7e-4	1.1e-4	4.3e-4	9.5e-5	0.2e-5	2.6e-5	1.6e-5	0.2e-5	9.1e-5

Dataset	**MNIST**						**Fashion MNIST**					
Regularization	ℓ_2			Softplus β			ℓ_2			Softplus β		
Magnitude	0	1e-5	10e-5	1.0	0.5	0.25	0	1e-5	10e-5	1.0	0.5	0.25
GPEC	**0.236**	**0.157**	**0.078**	**0.087**	**0.073**	**0.056**	**0.378**	**0.187**	**0.063**	**0.112**	**0.075**	**0.061**
Naive-GP	4.00	4.00	3.99	0.226	0.232	0.236	3.98	3.99	3.99	0.301	0.261	0.262
BayesSHAP	**0.025**	**0.016**	**0.008**	**0.013**	**0.011**	**0.010**	**0.030**	**0.014**	**0.007**	**0.018**	**0.010**	**0.009**
BayesLIME	**2.452**	**1.573**	**0.737**	0.868	0.866	0.721	**2.605**	**1.364**	**0.666**	**1.178**	**0.861**	**0.779**
CXPlain	0.1e-5	5.0e-5	8.6 e-5	5.3e-5	8.0e-5	5.4e-5	7.2e-5	4.8e-5	6.2 e-5	9.0e-5	6.6e-5	9.6e-5

Table 1: Average explanation uncertainty for classifiers with increasing (left to right) levels of regularization, which controls relative model complexity. We evaluate how well GPEC reflects model complexity; increased regularization should result in lower uncertainty. Methods that have decreasing estimates are bolded; the method with the greatest percentage decrease is highlighted in blue. CIFAR10 results for BayesSHAP and BayesLIME are omitted due to computational expense.

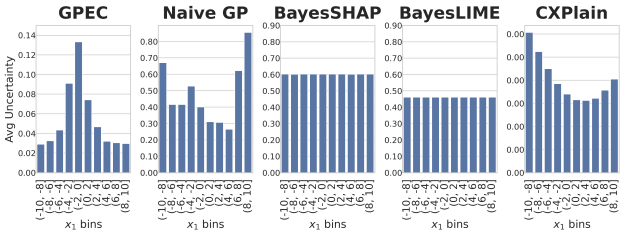

Figure 5: Average uncertainty values for different regions in Synthetic, binned by x_1. Synthetic is designed to have higher DB complexity for $x_1 \in [-4, 4]$, which is reflected by high GPEC uncertainty in bins $(-4, 2]$, $(-2, 0]$, $(0, 2]$, $(2, 4]$. Other methods do not capture the high DB complexity for $x_1 \in [-4, 4]$.

row), the top-right of of Census (2^{nd} and 3^{rd} rows), and the bottom-right of German Credit (bottom). In particular, Synthetic is constructed such that the DB for $x_1 \in [-4, 4]$ is more complex than $x_1 \notin [-4, 4]$. To highlight this, in Figure 5 we bin the values of x_1 and calculate the average uncertainty over each bin. Indeed we observe that the bins within $[-4, 4]$ exhibit the highest average uncertainty values under GPEC.

In contrast to GPEC, Naive-GP provides uncertainty estimates that relate only to the training samples; test sample uncertainty is proportional to distance from the training samples. The competing methods BayesSHAP, BayesLIME, and CXPlain result in relatively uniform uncertainty estimates over the test samples. CXPlain shows areas of higher uncertainty for Census, however the magnitude of these estimates are small. The uncertainty estimates produced by these competing methods are unable to capture the properties of the black-box model.

5.3 Regularization Test

In this section we evaluate how well GPEC captures uncertainty due to DB complexity. DB and model complexity is generally difficult to quantify; we instead use regularization methods to control relative model complexity. By examining the average uncertainty across different models, we can better understand how well GPEC uncertainty reflects the underlying DB complexity. For XGBoost models, we vary γ, which penalizes the number of leaves in each tree (Chen and Guestrin, 2016). For neural networks, we regularize: 1) ℓ_2 penalty on the weights, and 2) we change the ReLU activation to Softplus: a smooth approximation of ReLU with smoothness inversely proportional to parameter β (Dombrowski et al., 2019).

Results are presented in Table 1 (standard error and parameters reported in App. F.6). GPEC shows a decreasing average uncertainty with increased regularization, indicating its ability to reflect the complexity of the underlying black-box model. For tabular datasets, the estimates for BayesSHAP, BayesLIME, and CXPlain stay relatively flat. Interestingly, the estimates from these methods decrease for the image datasets; we hypothesize that the neural network regularization also increases overall stability of the explanations.

5.4 GPEC Ablation Test

GPEC can capture both the uncertainty from WEG kernel and also the estimated uncertainty from the noisy explanation labels. These noisy explanation labels can either originate directly from the explainer, or can be estimated empirically (Eq. (3)). Here, we calculate GPEC uncertainty with two different explainers, BayesSHAP and Shapley Sampling Values (SSV) (Strumbelj and Kononenko, 2013), and ablate the DB-

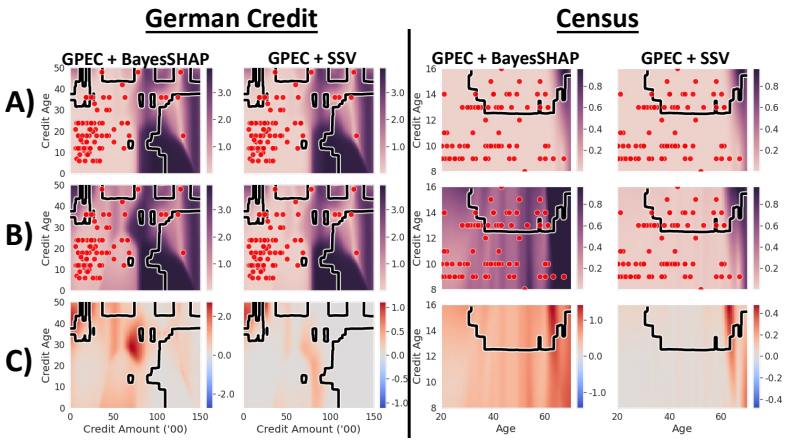

Figure 6: Ablation test. **Row (A)** visualizes the GPEC uncertainty estimate using BayesSHAP and SSV. **Row (B)** shows the DB-aware uncertainty component in GPEC for the estimate in row (A). **Row (C)** subtracts row (B) from row (A), which isolates the function approximation uncertainty in GPEC. GPEC is able to combine and also disentangle the two sources of uncertainty.

	Census	Shop	Credit	MNIST	fMNIST	CIFAR
GPEC	0.11	0.37	0.07	12.90	18.15	1.77
Naive-GP	0.00	0.00	0.02	8.95	7.41	1.29
CXPlain	0.05	0.06	0.04	9.76	18.18	8.27
BayesSHAP	140.40	54.56	4.86	42,467	42,361	–
BayesLIME	91.29	54.60	4.83	41,832	41,992	–

Table 2: Execution time comparison (inference) for uncertainty estimation of 100 samples, in seconds. CIFAR10 results for BayesLIME and BayesSHAP are ommitted due to computational cost. CIFAR10 results use a single Nvidia A100 GPU; all other results use CPU only.

aware uncertainty in order to evaluate function approximation uncertainty. The two selected explainers are different SHAP approximations; the former has an in-built uncertainty estimate whereas we use empirical sampling (Eq. (3)) for SSV. In Figure 6 row (A) we plot the heatmap for the combined GPEC estimate. In row (B) we exclusively show the DB-aware uncertainty by training GPEC with noiseless explanations. The difference (i.e. the effects of function approximation uncertainty) is shown in the row (C). We observe that the combined GPEC estimate is able to effectively integrate and disentangle both sources of uncertainty. Interestingly, the BayesSHAP explanations have higher function approximation uncertainty, which results in higher GPEC estimates. This suggests that users wanting to reduce explanation uncertainty might prefer SSV over BayesSHAP in this scenario.

5.5 Time Complexity

In Table 2 we show an execution time comparison for generating uncertainty estimates for 100 explanations (inference time). During inference, GPEC is comparable to methods that amortize training time (Naive-GP and CXPlain) and is significantly faster than perturbation methods (BayesSHAP and BayesLIME). Execution time results for training are shown in App. F.1.

5.6 Additional Results

Due to space constraints, we include additional experiments in the appendix, including a case study on diabetes prediction (App. F.2), illustrative examples for image datasets (App. F.3), and parameter sensitivity analysis (App. F.4).

6 CONCLUSION

Generating uncertainty estimates for feature attribution explainers is essential for establishing reliable explanations. We introduce a novel GP-based approach that can be used with any black-box classifier and feature attribution method. GPEC generates explanation uncertainty that combines 1) boundary-aware uncertainty, which captures the complexity of the DB, and 2) functional approximation uncertainty. Experiments show that capturing this uncertainty improves understanding of the explanations and the black-box model itself.

Regarding limitations, GPEC relies on DB estimation methods which is an ongoing area of research. Due to the time complexity of DB estimation, this can result in a tradeoff between computation time and approximation accuracy or sample bias. However, the effects of DB sampling time are minimized during inference as the DB only needs to be sampled during training. Additionally, in its current implementation GPEC is limited to classification; we leave the extension to regression as future work.

Acknowledgements

This work was supported in part by NIH 2T32HL007427-41 and U01HL089856 from the National Heart, Lung, and Blood Institute; NIH R01CA240771 and U24CA264369 from the National Cancer Institute; and the Institute of Experiential AI at Northeastern University.

References

David Alvarez-Melis and Tommi S. Jaakkola. On the robustness of interpretability methods. *CoRR*, abs/1806.08049, 2018.

Zulqarnain Khan, Davin Hill, Aria Masoomi, Joshua Bone, and Jennifer Dy. Analyzing explainer robustness via lipschitzness of prediction functions, 2023.

Ann-Kathrin Dombrowski, Maximillian Alber, Christopher Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. Explanations can be manipulated and geometry is to blame. *Advances in Neural Information Processing Systems*, 32, 2019.

Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of Neural Networks Is Fragile. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3681–3688, July 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33013681.

Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, pages 180–186, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 978-1-4503-7110-0. doi: 10.1145/3375627.3375830.

Giorgio Visani, Enrico Bagli, Federico Chesani, Alessandro Poluzzi, and Davide Capuzzo. Statistical stability indices for LIME: Obtaining reliable explanations for machine learning models. *Journal of the Operational Research Society*, 73(1):91–101, 2022. doi: 10.1080/01605682.2020.1865846.

Dylan Slack, Sophie Hilgard, Sameer Singh, and Himabindu Lakkaraju. Reliable Post hoc Explanations Modeling Uncertainty in Explainability. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Patrick Schwab and Walter Karlen. CXPlain: Causal explanations for model interpretation under uncertainty. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Guillermo Valle-Perez, Chico Q. Camargo, and Ard A. Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rye4g3AqFm.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9078–9086, 2019.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. Empirical study of the topology and geometry of deep networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3762–3770, 2018. doi: 10.1109/CVPR.2018.00396.

Zifan Wang, Haofan Wang, Shakul Ramkumar, Piotr Mardziel, Matt Fredrikson, and Anupam Datta. Smoothed geometry for robust attribution. *Advances in neural information processing systems*, 33: 13623–13634, 2020.

Emily Black, Zifan Wang, and Matt Fredrikson. Consistent counterfactuals for deep models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=St6eyiTEHnG.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *Acm Computing Surveys*, 51(5), August 2018. ISSN 0360-0300. doi: 10.1145/3236009.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020. ISSN 1566-2535. doi: 10.1016/j.inffus.2019.12.012.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 883–892. PMLR, July 2018.

Aria Masoomi, Chieh Wu, Tingting Zhao, Zifeng Wang, Peter Castaldi, and Jennifer Dy. Instance-wise feature grouping. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 13374–13386. Curran Associates, Inc., 2020.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "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*, KDD '16, pages 1135–1144, New York, NY, USA, 2016.

Association for Computing Machinery. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939778.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10, 2015.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3145–3153. PMLR, August 2017.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3319–3328. JMLR.org, 2017.

Gabriel Erion, Joseph D. Janizek, Pascal Sturmfels, Scott M. Lundberg, and Su-In Lee. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. *Nature Machine Intelligence*, 3(7):620–631, July 2021. ISSN 2522-5839. doi: 10.1038/s42256-021-00343-w.

Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1):56–67, January 2020. ISSN 2522-5839. doi: 10.1038/s42256-019-0138-9.

Kirill Bykov, Marina M-C Höhne, Klaus-Robert Müller, Shinichi Nakajima, and Marius Kloft. How much can I trust You?–Quantifying uncertainties in explaining neural networks. *arXiv preprint arXiv:2006.09000*, 2020.

Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Wenbo Guo, Sui Huang, Yunzhe Tao, Xinyu Xing, and Lin Lin. Explaining deep learning models – a bayesian non-parametric approach. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Neil Jethani, Mukund Sudarshan, Ian Connick Covert, Su-In Lee, and Rajesh Ranganath. FastSHAP: Real-time shapley value estimation. In *International Conference on Learning Representations*, 2022.

Jonas Schulz, Raul Santos-Rodriguez, and Rafael Poyiadzi. Uncertainty quantification of surrogate explanations: An ordinal consensus approach. In *Proceedings of the Northern Lights Deep Learning Workshop*, volume 3, 2022.

Badri N. Patro, Mayank Lunayach, Shivansh Patel, and Vinay P. Namboodiri. U-cam: Visual explanation using uncertainty based class activation maps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Ian Covert and Su-In Lee. Improving kernelSHAP: Practical shapley value estimation using linear regression. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 3457–3465. PMLR, April 2021.

Yujia Zhang, Kuangyan Song, Yiming Sun, Sarah Tan, and Madeleine Udell. " why should you trust my explanation?" understanding uncertainty in lime explanations. *arXiv preprint arXiv:1904.12991*, 2019.

Xingyu Zhao, Wei Huang, Xiaowei Huang, Valentin Robu, and David Flynn. BayLIME: Bayesian local interpretable model-agnostic explanations. In Cassio de Campos and Marloes H. Maathuis, editors, *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 887–896. PMLR, July 2021.

Aditya Saini and Ranjitha Prasad. Select wisely and explain: Active learning and probabilistic local post-hoc explainability. In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '22, pages 599–608, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 978-1-4503-9247-1. doi: 10.1145/3514094.3534191.

Aasa Feragen, François Lauze, and Søren Hauberg. Geodesic exponential kernels: When curvature and linearity conflict. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3032–3042, 2015. doi: 10.1109/CVPR.2015.7298922.

Aasa Feragen and Søren Hauberg. Open Problem: Kernel methods on manifolds and metric spaces. What is the probability of a positive definite

geodesic exponential kernel? In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 1647–1650, Columbia University, New York, New York, USA, June 2016. PMLR.

Koji Tsuda, Taishin Kin, and Kiyoshi Asai. Marginalized kernels for biological sequences. *Bioinformatics (Oxford, England)*, 18(suppl$_1$):S268–S275, July 2002. ISSN 1367-4803. doi: 10.1093/bioinformatics/18.suppl_1.S268.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable Variational Gaussian Process Classification. In Guy Lebanon and S. V. N. Vishwanathan, editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 351–360, San Diego, California, USA, 09–12 May 2015. PMLR.

Hamid Karimi, Tyler Derr, and Jiliang Tang. Characterizing the decision boundary of deep neural networks. *arXiv preprint arXiv:1912.11460*, 2019.

Zhiyong Yan and Congfu Xu. Using decision boundary to analyze classifiers. In *2008 3rd International Conference on Intelligent System and Knowledge Engineering*, volume 1, pages 302–307, Nov 2008. doi: 10.1109/ISKE.2008.4730945.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science (New York, N.Y.)*, 290 (5500):2319–2323, December 2000. ISSN 0036-8075. doi: 10.1126/science.290.5500.2319.

C. Okan Sakar, S. Olcay Polat, Mete Katircioglu, and Yomi Kastro. Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks. *Neural Computing and Applications*, 31(10):6893–6908, October 2019. ISSN 1433-3058. doi: 10.1007/s00521-018-3523-0.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL http://arxiv.org/abs/1708.07747.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

Erik Strumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41:647–665, 2013.

Andre Esteva, Alexandre Robicquet, Bharath Ramsundar, Volodymyr Kuleshov, Mark DePristo, Katherine Chou, Claire Cui, Greg Corrado, Sebastian Thrun, and Jeff Dean. A guide to deep learning in healthcare. *Nature medicine*, 25(1):24–29, 2019.

Kivanc Kose, Alican Bozkurt, Christi Alessi-Fox, Melissa Gill, Caterina Longo, Giovanni Pellacani, Jennifer G Dy, Dana H Brooks, and Milind Rajadhyaksha. Segmentation of cellular patterns in confocal images of melanocytic lesions in vivo via a multiscale encoder-decoder network (med-net). *Medical Image Analysis*, 67:101841, 2021.

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Mohammad Ahmad Sheikh, Amit Kumar Goel, and Tapas Kumar. An approach for prediction of loan approval using machine learning algorithm. In *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pages 490–494, 2020. doi: 10.1109/ICESC48915.2020.9155614.

Ashutosh Singh, Christiana Westlin, Hedwig Eisenbarth, Elizabeth A. Reynolds Losin, Jessica R. Andrews-Hanna, Tor D. Wager, Ajay B. Satpute, Lisa Feldman Barrett, Dana H. Brooks, and Deniz Erdogmus. Variation is the norm: Brain state dynamics evoked by emotional video clips. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 6003–6007, 2021. doi: 10.1109/EMBC46164.2021.9630852.

Davin Hill, Max Torop, Aria Masoomi, Peter Castaldi, Edwin K. Silverman, Sandeep Bodduluri, Surya P Bhatt, Taedong Yun, Cory Y McLean, Farhad Hormozdiari, Jennifer Dy, Michael Cho, and Brian D Hobbs. Deep learning utilizing suboptimal spirometry data to improve lung function and mortality prediction in the uk biobank. *medRxiv*, 2023. doi: 10.1101/2023.04.28.23289178. URL https://www.medrxiv.org/content/early/2023/04/29/2023.04.28.23289178.

Mahsa Bazzaz and Seth Cooper. Active learning for classifying 2d grid-based level completability. In *2023 IEEE Conference on Games (CoG)*, pages 1–4. IEEE, 2023.

Ian Covert, Scott Lundberg, and Su-In Lee. Feature Removal Is a Unifying Principle for Model Explanation Methods. *arXiv:2011.03623 [cs, stat]*, November 2020.

Aria Masoomi, Davin Hill, Zhonghui Xu, Craig P. Hersh, Edwin K. Silverman, Peter J. Castaldi, Stratis Ioannidis, and Jennifer Dy. Explanations of black-box models based on directional feature interactions. In *International Conference on Learning Representations*, 2021.

Max Torop, Aria Masoomi, Davin Hill, Kivanc Kose, Stratis Ioannidis, and Jennifer Dy. Smoothhess: ReLU network feature interactions via stein's lemma. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=dwIeEhbaD0.

Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T. Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. *The (Un)reliability of Saliency Methods*, pages 267–280. Springer International Publishing, Cham, 2019. ISBN 978-3-030-28954-6. doi: 10.1007/978-3-030-28954-6_14. URL https://doi.org/10.1007/978-3-030-28954-6_14.

Naman Bansal, Chirag Agarwal, and Anh Nguyen. SAM: The Sensitivity of Attribution Methods to Hyperparameters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8673–8683, 2020.

Cheng-Yu Hsieh, Chih-Kuan Yeh, Xuanqing Liu, Pradeep Ravikumar, Seungyeon Kim, Sanjiv Kumar, and Cho-Jui Hsieh. Evaluations and Methods for Explanation through Robustness Analysis. *arXiv:2006.00442 [cs, stat]*, April 2021.

Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye, and Pradeep Ravikumar. On the (In)fidelity and Sensitivity for Explanations. *arXiv:1901.09392 [cs, stat]*, November 2019.

Umang Bhatt, Adrian Weller, and José M. F. Moura. Evaluating and Aggregating Feature-based Model Explanations. *arXiv:2005.00631 [cs, stat]*, May 2020.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 11 2005. ISBN 9780262256834. doi: 10.7551/mitpress/3206.001.0001. URL https://doi.org/10.7551/mitpress/3206.001.0001.

George Casella and Roger Berger. *Statistical Inference*. Duxbury Resource Center, June 2001. ISBN 0-534-24312-6.

Walter Rudin. *Principles of Mathematical Analysis*. McGraw-Hill Book Co., New York, third edition, 1976.

Dmitri Burago, Yuri Burago, and Sergei Ivanov. *A course in metric geometry*, volume 33. American Mathematical Society, 2001.

Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. 2012.

Daniel Sheldon. Graphical multi-task learning. 2008.

Theodoros Evgeniou, Charles A. Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6(21):615–637, 2005. URL http://jmlr.org/papers/v6/evgeniou05a.html.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017. URL http://arxiv.org/abs/1707.04131.

Jonas Rauber, Roland Zimmermann, Matthias Bethge, and Wieland Brendel. Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax. *Journal of Open Source Software*, 5(53):2607, 2020. doi: 10.21105/joss.02607. URL https://doi.org/10.21105/joss.02607.

Nutan Chen, Francesco Ferroni, Alexej Klushyn, Alexandros Paraschos, Justin Bayer, and Patrick van der Smagt. Fast approximate geodesics for deep generative models. In *Artificial Neural Networks and Machine Learning–ICANN 2019: Deep Learning: 28th International Conference on Artificial Neural Networks, Munich, Germany, September*

17–19, 2019, Proceedings, Part II 28, pages 554–566. Springer, 2019.

Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. 1 2008. URL https://www.osti.gov/biblio/960616.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4.

Mihaela Rosca, Theophane Weber, Arthur Gretton, and Shakir Mohamed. A case for new neural network smoothness constraints. In Jessica Zosa Forde, Francisco Ruiz, Melanie F. Pradier, and Aaron Schein, editors, *Proceedings on "I Can't Believe It's Not Better!" at NeurIPS Workshops*, volume 137 of *Proceedings of Machine Learning Research*, pages 21–32. PMLR, 12 Dec 2020. URL https://proceedings.mlr.press/v137/rosca20a.html.

H. W. Miller. Plan and operation of the health and nutrition examination survey. United states–1971-1973. *Vital and health statistics. Ser. 1, Programs and collection procedures*, (10a):1–46, February 1973. ISSN 0083-2014.

An Dinh, Stacey Miertschin, Amber Young, and Somya D. Mohanty. A data-driven approach to predicting diabetes and cardiovascular disease with machine learning. *BMC medical informatics and decision making*, 19(1):211, November 2019. ISSN 1472-6947. doi: 10.1186/s12911-019-0918-5.

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A BROADER IMPACT STATEMENT

Machine learning models are increasingly relied upon in a diverse set of high-impact domains, ranging from healthcare to financial lending (Esteva et al., 2019; Kose et al., 2021; Doshi-Velez and Kim, 2017; Sheikh et al., 2020; Singh et al., 2021; Hill et al., 2023; Bazzaz and Cooper, 2023). Therefore, it is crucial that users of these models can accurately interpret why predictions are made. For instance, a doctor may wish to know if a skin-cancer classifier's high performance is due to the use of truly diagnostic features, or rather due to spurious imaging device artifacts. However, further spurred by the advent of deep learning's increasing popularity, many of the models being deployed in these high-stakes fields are opaque and complex black boxes, producing predictions which are non-trivial to understand. Many methods for explaining black-box predictions have been developed to address this issue (Ribeiro et al., 2016; Lundberg and Lee, 2017; Covert et al., 2020; Masoomi et al., 2021; Torop et al., 2023), but explanations may have varying quality and consistency. Before utilizing explanations in practice, it is essential that users know when, and when not, to trust them. Explanation uncertainty is one proxy for this notion of trust, in which more uncertain explanations may be deemed less trustworthy. In this work, we explore a new way to model explanation uncertainty, in terms of local decision-boundary complexity. In tandem with the careful consideration of domain experts, our methodology may be used to assist in determining when explanations are reliable.

B BACKGROUND

B.1 Related Works: Reliability of Explanations

While feature attribution methods have gained wide popularity, a number of issues relating to the reliability of such methods have been uncovered. Alvarez-Melis and Jaakkola (2018) investigate the notion of robustness and show that many feature attribution methods are sensitive to small changes in input. This has been further investigated in the adversarial setting for perturbation-based methods (Slack et al., 2020) and neural network-based methods (Ghorbani et al., 2019). Kindermans et al. (2019) show that many feature attribution methods are affected by distribution transformations such as those common in preprocessing. The generated explanations can also be very sensitive to hyperparameter choice (Bansal et al., 2020). A number of metrics have been proposed for evaluating explainer reliability, such as with respect to adversarial attack (Dombrowski et al., 2019; Ghorbani et al., 2019; Hsieh et al., 2021), local perturbations (Alvarez-Melis and Jaakkola, 2018; Visani et al., 2022), black-box smoothness (Khan et al., 2023), fidelity to the black-box model (Yeh et al., 2019), or combinations of these metrics (Bhatt et al., 2020).

B.2 Gaussian Process Review

A single-output Gaussian Process represents a distribution over *functions* $f : \mathcal{X} \to \mathbb{R}$

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')). \tag{12}$$

where $m : \mathcal{X} \to \mathbb{R}$ and $k : (\mathcal{X}, \mathcal{X}) \to \mathbb{R}$ are the mean and kernel (or covariance) functions respectively, which are chosen *a priori* to encode the users assumptions about the data. The kernel function $k(x, x')$ reflects a notion of similarity between data points for which predictive distributions over $f(x), f(x')$ respect.

The prior $m(x)$ – frequently considered to be less important – is commonly chosen to be the constant $m(x) = 0$.

Specifically, a GP is an infinite collection of random variables $f(x)$, each indexed by an element $x \in \mathcal{X}$. Importantly, any finite sub-collection of these random variables

$$f(X_{tr}) = (f(x_1) \ldots, f(x_n)) \in \mathbb{R}^D, \tag{13}$$

corresponding to some index set $X_{tr} = \{x_i\}_{i=1}^n \subset \mathcal{X}$, follows the multivariate normal distribution, i.e.

$$f(X_{tr}) \sim \mathcal{N}(m(X_{tr}), K(X_{tr}, X_{tr})). \tag{14}$$

The mean vector $m(X_{tr}) = (m(x_1), \ldots, m(x_n)) \in \mathbb{R}^n$ represents the mean function applied on each $x \in X_{tr}$ and the covariance matrix $K \in \mathbb{R}^{n \times n}$, also known as the gram matrix, contains each pairwise kernel-based similarity value $K_{ij} = k(x_i, x_j)$. Kernel function outputs correspond to dot products in potentially infinite dimensional

expanded feature space, which allows for the encoding of nuanced notions of similarity; e.g. the exponential geodesic kernel referenced in this work (Feragen et al., 2015).

Making predictions with a GP is analogous to simply conditioning this normal distribution on our data. Considering a set of input,noise-free label pairs

$$\mathcal{D} = \{(x_i, f(x_i)\}_{i=1}^n \tag{15}$$

we may update our posterior over *any subset* of the random variables $f(x)$ by considering the joint normal over the subset and $\mathcal{D}$ and conditioning on $\mathcal{D}$. For instance, when choosing a singleton index set $\{x_0\}$, the posterior over $f(x_0)|D$ is another normal distribution which may be written as[3]

$$f(x_0) \sim \mathcal{N}(\bar{f}(x_0), \mathbb{V}[f(x_0)]) \tag{16}$$

where

$$\bar{f}(x_0) = K(x_0, X_{tr})^T K(X_{tr}, X_{tr})^{-1} f(X_{tr}) \tag{17}$$

$$\mathbb{V}[f(x_0)] = k(x_0, x_0) - K(x_0, X_{tr})^T K(X_{tr}, X_{tr})^{-1} K(x_0, X_{tr}) \tag{18}$$

and $K(x_0, X_{tr}) \in \mathbb{R}^D$ is defined element-wise by $K(x_0, X_{tr})_i = k(x_0, x_i)$.

Now we may consider the situation where our labels are noisy:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n,\ y_i = f(x_i) + \epsilon,\ \epsilon \sim \mathcal{N}(0, \sigma^2),\ \sigma^2 \in \mathbb{R}_+. \tag{19}$$

Here y_i is equal to the function output $f(x_i)$, with the addition of noise variable ϵ. The conditional still follows a multivariate normal distribution, but the mean and variance equations are updated:

$$\bar{f}(x_0) = K(x_0, X_{tr})^T (K(X_{tr}, X_{tr}) + \sigma^2 I)^{-1} Y \tag{20}$$

$$\mathbb{V}[f(x_0)] = k(x_0, x_0) - K(x_0, X_{tr})^T (K(X_{tr}, X_{tr}) + \sigma^2 I)^{-1} K(x_0, X_{tr}) \tag{21}$$

where $Y \in \mathbb{R}^n$ has elements $Y_i = y_i$.

Note that the variance $\sigma^2 I$ is added to $K(X_{tr}, X_{tr})$ in the quadratic form in Eq. (21), resulting in smaller eigenvalues after matrix inversion. Since this quadratic form is subtracted, the decision to model labels as noisy increases the uncertainty (variance) of the estimates that the GP posterior provides. This agrees with the intuition that noisy labels should result in more uncertain predictions.

While GPs may also be defined over vector-valued functions, in this work the independence of each output component is assumed, allowing for modeling with $c \geq 1$ independent GPs. For more details see Ch.2 of Rasmussen and Williams (2005).

C PROOF OF THEOREMS AND EXTENSIONS

C.1 Theorem 1: Relation to Exponential Geodesic Kernel

$$k(x, y) = \int \int \exp[-\lambda d_{\text{geo}}(m, m')] q(m|x, \rho) q(m'|y, \rho)\ \mathrm{d}m' \mathrm{d}m$$
$$s.t. \quad q(m|x, \rho) \propto \exp[-\rho ||x - m||_2^2] p(m)$$

Note that ρ controls how to weight manifold samples close to x, y. We take $\lim_{\rho \to \infty}$:

$$\lim_{\rho \to \infty} q(m|x, \rho) q(m'|y, \rho) = \begin{cases} 1 & x = m \text{ and } y = m' \\ 0 & \text{Otherwise} \end{cases}$$

Therefore the function within the integral of $k(x, y)$ evaluates to zero at all points except $x = m$ and $y = m'$. Since $x, y \in \mathcal{M}_F$ we can evaluate the integral:

$$k(x, y) = \exp[-\lambda d_{\text{geo}}(x, y)]$$

[3] assuming prior $m(x) = 0$

C.2 Theorem 2: Kernel Similarity and Decision Boundary Complexity

From definition 1, given any perturbation $\widetilde{\mathcal{P}}$ on $\mathcal{P}$, there must exist a compact subset $K_i \subset U_i$ s.t. $R|_{\mathcal{P}\setminus\text{int}(K_i)} = \text{id}|_{\mathcal{P}\setminus\text{int}(K_i)}$ and $R|_{\text{int}(K_i)} \neq \text{id}|_{\text{int}(K_i)}$. Furthermore there exists a linear homeomorphism between an open subset $\widetilde{U_i} \subseteq U_i$ with $\mathbb{R}^{d-1}$ which contains K_i.

We parametrize K_i using a smooth function $g : \mathcal{T} \to K_i$ s.t. $g(t) \in \partial K_i \; \forall t \in \partial\mathcal{T}$.

We further define $g_\epsilon(t) = g(t) + \epsilon\eta(t)$, for some perturbation $\epsilon \in \mathbb{R}$ and a smooth function $\eta : \mathcal{T} \to \mathbb{R}^{d-1}$ We also restrict η such that $\eta(t) = \mathbf{0} \; \forall t \in \partial\mathcal{T}$ and $\exists\; t_0 \in \mathcal{T}$ s.t. $\eta(t_0) \neq g(t_0)$. In other words, η is a smooth function where $g_\epsilon(t) = g(t) \; \forall \epsilon > 0, \forall t \in \partial\mathcal{T}$, but is not identical to g for all $t \in \mathcal{T}$. Using $g_\epsilon(t)$, we define the manifold $\mathcal{P}_\epsilon = \{g_\epsilon(t) : t \in \mathcal{T}\}$.

To complete the proof, we want to show that the kernel similarity between any two given points $x, y \in \mathbb{R}^D$ is lower when using the manifold $\mathcal{P}_\epsilon$ for $\epsilon > 0$ as opposed to the manifold $\mathcal{P}_0$. We therefore want to compare the two respective kernels $k_\epsilon(x, y)$ and $k_0(x, y)$. Note that in this proof we consider the local effects of $\mathcal{P}$ on the kernel similarity through $\mathcal{P}_0$ and $\mathcal{P}_\epsilon$ exclusively, ignoring the manifold $\mathcal{P} \setminus U_0$ Using Euler-Lagrange, we can calculate a lower bound for $d_{\text{geo}}(g_\epsilon(t)), g_\epsilon(t')))$. In particular, for any $t, t' \in \mathcal{T}$, $d_{\text{geo}}(g_\epsilon(t)), g_\epsilon(t'))) \geq d_{\text{geo}}(g_0(t), g_0(t'))$.

$$d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t')) \geq d_{\text{geo}}(g_0(t), g_0(t')) \tag{22}$$

$$\exp[-\lambda d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t'))] \leq \exp[-\lambda d_{\text{geo}}(g_0(t), g_0(t'))] \tag{23}$$

$$\int_{\mathcal{T}}\int_{\mathcal{T}} \exp[-\lambda d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t'))]\, \text{d}t\text{d}t' \leq \int_{\mathcal{T}}\int_{\mathcal{T}} \exp[-\lambda d_{\text{geo}}(g_0(t), g_0(t'))]\, \text{d}t\text{d}t' \tag{24}$$

Note that in Eq. (24) we are integrating over all possible values of t, t', therefore the inequality is tight iff $g_\epsilon(t) = g_0(t) \; \forall t \in \mathcal{T}$; i.e. $\epsilon = 0$ (see proof in C.2.1). The case of $\epsilon = 0$ is trivial; we instead assume $\epsilon > 0$, in which case we can establish the following strict inequality:

$$\int_{\mathcal{T}}\int_{\mathcal{T}} \exp[-\lambda d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t'))]\, \text{d}t\text{d}t' < \int_{\mathcal{T}}\int_{\mathcal{T}} \exp[-\lambda d_{\text{geo}}(g_0(t), g_0(t'))]\, \text{d}t\text{d}t' \tag{25}$$

Define uniform random variables T, T' over the domain of g, i.e. $T, T' \sim \mathcal{U}_{\mathcal{T}}$. Then we have:

$$\mathbb{E}_{T,T'\sim\mathcal{U}_{[0,1]}}[\exp[-\lambda d_{\text{geo}}(g_\epsilon(T), g_\epsilon(T'))]] < \mathbb{E}_{T,T'\sim\mathcal{U}_{[0,1]}}[\exp[-\lambda d_{\text{geo}}(g_0(T), g_0(T'))]] \tag{26}$$

$$\mathbb{E}_{M,M'\sim p_\epsilon(M)}[\exp[-\lambda d_{\text{geo}}(M, M')]] < \mathbb{E}_{M,M'\sim p_0(M)}[\exp[-\lambda d_{\text{geo}}(M, M')]] \tag{27}$$

We define the random variable $M = g_\epsilon(T)$ with distribution $p_\epsilon(M)$. The distribution $p_\epsilon(M)$ represents the uniform distribution $\mathcal{U}_{\mathcal{T}}$ mapped to the manifold $\mathcal{P}_\epsilon$ using $g_\epsilon(T)$. The step from Eq. (26) to Eq. (27) uses a property of distribution transformations (Eq. 2.2.5 in Casella and Berger (2001)).

Next, compare either side of Eq. (27) to our kernel formulation shown below in Eq. (28). The kernel $k_\epsilon(x, y|\rho, \lambda)$ takes an expected value over $q_\epsilon(M|x, \rho)$ and $q_\epsilon(M'|y, \rho)$, which are equivalent to $p_\epsilon(M)$ and $p_\epsilon(M')$ weighted with respect to x, y, and a hyperparameter $\rho \geq 0$.

$$\begin{aligned} k_\epsilon(x, y|\rho, \lambda) = \mathbb{E}_{M\sim q_\epsilon(M|x,\rho), M'\sim q_\epsilon(M'|y,\rho)}[\exp[-\lambda d_{\text{geo}}(M, M')]] \\ s.t. \quad q_\epsilon(M|x, \rho) \propto \exp[-\rho||x - M||_2^2]p_\epsilon(M) \\ s.t. \quad q_\epsilon(M'|y, \rho) \propto \exp[-\rho||y - M'||_2^2]p_\epsilon(M') \end{aligned} \tag{28}$$

Note that when ρ is set to zero, $q(M|x,0) = p(M)$ and $q(M'|y,0) = p(M')$. Therefore Eq. (27) is equivalent to the inequality $k_\epsilon(x,y|0,\lambda) < k_0(x,y|0,\lambda)$.

We next want to prove that the inequality $k_\epsilon(x,y|\rho,\lambda) < k_0(x,y|\rho,\lambda)$ also holds for non-zero values of ρ. For convenience, define

$$f(\rho) = k_0(x,y|\rho,\lambda) - k_\epsilon(x,y|\rho,\lambda) \tag{29}$$

Under this definition, we want to prove there exists $\rho_0 > 0$ such that $f(\rho) > 0 \ \forall \rho < \rho_0$. From Eq. (27), we established that $f(0) > 0$. Assume that

$$\lim_{\rho \to 0} f(\rho) = c \tag{30}$$

It therefore follows that $c > 0$. In addition, note that $f(\rho)$ is continuous with respect to ρ (see proof in section C.2.3). Therefore for any $\epsilon > 0$ there exists $\delta > 0$ s.t. $\rho < \delta$ implies $|f(\rho) - c| < \epsilon$.

We choose $\epsilon = c$ and the define the corresponding δ to be ρ_0. Therefore:

$$\rho < \rho_0 \Rightarrow |f(\rho) - c| < c \tag{31}$$

$$\rho < \rho_0 \Rightarrow 0 < f(\rho) < 2c \tag{32}$$

Since this result holds for any i, it follows that the piecewise linear manifold $\mathcal{P}$ is a local minimum under any perturbation along a specific chart or combination of charts with respect to the kernel similarity $k(x,y) \ \forall x, y \in \mathbb{R}^D$.

C.2.1 Proof: Inequality Tightness

From Eq. (24) to Eq. (25), we want to prove:

$$\int_{\mathcal{T}} \int_{\mathcal{T}} \exp[-\lambda d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t'))] \, \mathrm{d}t\mathrm{d}t' = \int_{\mathcal{T}} \int_{\mathcal{T}} \exp[-\lambda d_{\text{geo}}(g_0(t), g_0(t'))] \, \mathrm{d}t\mathrm{d}t' \tag{33}$$
$$\Rightarrow g_\epsilon(t) = g_0(t) \quad \forall t \in \mathcal{T}$$

Consider the LHS of Eq. (33):

$$\int_{\mathcal{T}} \int_{\mathcal{T}} \exp[-\lambda d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t'))] \, \mathrm{d}t\mathrm{d}t' = \int_{\mathcal{T}} \int_{\mathcal{T}} \exp[-\lambda d_{\text{geo}}(g_0(t), g_0(t'))] \, \mathrm{d}t\mathrm{d}t' \tag{34}$$

$$\int_{\mathcal{T}} \int_{\mathcal{T}} \underbrace{\exp[-\lambda d_{\text{geo}}(g_0(t), g_0(t'))] - \exp[-\lambda d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t'))]}_{h(t,t')} \, \mathrm{d}t\mathrm{d}t' = 0 \tag{35}$$

Define $h(t,t')$ as the function inside the integrals in Eq. (35). From Eq. (23), $h(t,t') \geq 0 \ \forall t, t' \in \mathcal{T}$. Since h is continuous (see proof in C.2.2) and $\int_{\mathcal{T}} \int_{\mathcal{T}} h(t,t') \mathrm{d}t\mathrm{d}t' = 0$, it follows that $h(t,t') = 0 \ \forall t, t' \in \mathcal{T}$ (Ch.6 Rudin (1976)).

It therefore follows that:

$$\exp[-\lambda d_{\text{geo}}(g_0(t), g_0(t'))] = \exp[-\lambda d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t'))] \quad \forall t, t' \in \mathcal{T} \tag{36}$$

From the definition of $\eta(t)$ in $g_\epsilon(t) = g(t) + \epsilon\eta(t)$, there must exist $t \in \mathcal{T}$ s.t. $\eta(t) \neq 0$. Therefore ϵ must be zero for Eq. (36) to hold. It follows that $g_\epsilon(t) = g_0(t) \ \forall t \in \mathcal{T}$.

C.2.2 Proof: Continuity of $h(t, t')$

We prove that $h(t, t')$ is continuous with respect to t, t'. First note that by definition, $g_\epsilon(t)$ is a continuous parametrization of the manifold $\mathcal{P}_\epsilon$. From Burago et al. (2001), it follows that for any two points $g_\epsilon(t), g_\epsilon(t') \in \mathcal{P}_\epsilon$, $d_{\text{geo}}(g_\epsilon(t), g_\epsilon(t'))$ is continuous. Since the exponential functional preserves continuity and the sum of continuous functions are also continuous, it follows that $h(t, t')$ is continuous.

C.2.3 Proof: Continuity of $k(x, y)$ With Respect To ρ

We prove that $k(x, y)$ is continuous with respect to ρ.

$$k(x,y) = \int\int \exp[-\lambda d_{\text{geo}}(m,m')]\; q(m|x,\rho)\; q(m'|y,\rho)\; \mathrm{d}m\mathrm{d}m' \tag{37}$$

$$= \frac{1}{Z_m(\rho) Z_{m'}(\rho)} \int\int \mathcal{A} \underbrace{\exp[-\rho(||x-m||_2^2 + ||y-m'||_2^2)]}_{Z(\rho)}\; \mathrm{d}m\mathrm{d}m' \tag{38}$$

$$s.t. \quad Z_m(\rho) = \int \exp[-\rho||x-m||_2^2] p(m)\; \mathrm{d}m$$

$$Z_{m'}(\rho) = \int \exp[-\rho||y-m'||_2^2] p(m')\; \mathrm{d}m'$$

$$\mathcal{A} = \exp[-\lambda d_{\text{geo}}(m,m')] p(m) p(m')$$

Define $h(\rho) = \rho\mathcal{B}$, where $\mathcal{B}$ is a constant. Consider $h(\rho) - h(\rho_0)$, where ρ_0 is a fixed positive constant:

$$|h(\rho) - h(\rho_0)| = |\rho\mathcal{B} - \rho_0\mathcal{B}| \tag{39}$$

$$= |(\rho - \rho_0)\mathcal{B}| < \delta|\mathcal{B}| \tag{40}$$

It follows that $\forall\, \epsilon > 0$, $\exists\, \delta = \frac{\epsilon}{|\mathcal{B}|} > 0$ such that $|\rho - \rho_0| < \delta \;\Rightarrow\; |h(\rho) - h(\rho_0)| < \epsilon$. Therefore h is continuous for all $\rho \in \mathbb{R}^+$.

We set $\mathcal{B}$ to be $||x-m||_2^2$, $||y-m'||_2^2$, and $||x-m||_2^2 + ||y-m'||_2^2$, which shows that $Z_m(\rho)$, $Z_{m'}(\rho)$, and $Z(\rho)$ are also continuous, respectively. It then follows that the entirety of Eq. (38) is continuous.

C.3 Extending to Multiclass Classifiers

In the multiclass case we define a black-box prediction model $F : \mathcal{X} \to \mathbb{R}^c$. We consider the one-vs-all DB for every class $y \in \mathcal{Y} = \{1, \ldots, c\}$, defined as $\mathcal{M}_{F_y} = \{x \in \mathbb{R}^D : F_y(x) = \max_{i \in \mathcal{Y}}\; F_i(x) = \max_{j \neq y \in \mathcal{Y}} F_j(x)\}$, where F_y indicates the model output for class y. We then apply the GPEC framework separately to each class using the respective DB. The uncertainty estimate of the GP model would be of dimension $d \times c$.

C.4 Feature Dependency in GPEC Output

A vector-valued GP is an extension of the traditional GP which has vector-valued output. Let $\mathcal{X} \subseteq \mathbb{R}^D$ be the data space and $E : \mathcal{X} \to \mathbb{R}^D$ be an explainer with explanations $e = E(x)\; \forall x \in \mathcal{X}$. We sample $\mathcal{X}$ and generate N pairs $\mathcal{S} = \{(x_1, e_1), ..., (x_N, e_N)\}$ In the main text, we train a vector-valued GP on each explanation dimension independently; i.e. we train D independent scalar-valued GPs. This approach has the advantage of simplicity and implementation efficiency. However, alternative approaches can be used to enforce *a priori* dependency between the dimensions of the vector-valued GP output. Many matrix-valued kernels for vector-valued GPs have been investigated (see Álvarez et al. (2012) for a review). In particular, we review *separable* kernels below, which allow an intuitive decomposition for the matrix-valued kernel.

Let $\mathbb{N}_D = \{z \in \mathbb{N} : z \leq D\}$. Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ and $k_T : \mathbb{N}_D \times \mathbb{N}_D \to \mathbb{R}_{\geq 0}$ be a scalar-valued kernel functions. The function k represents the standard kernel function for a GP (e.g. the WEG kernel for GPEC). The function

k_T represents an encoded similarity between tasks[4] $\{1, ..., D\}$. In the context of multi-task learning, k and k_T are sometimes referred to as the *base kernel* and *task kernel*, respectively. We next define the respective kernel matrices K and B.

$$K = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_N) \\ \vdots & \ddots & \vdots \\ k(x_N, x_1) & \dots & k(x_N, x_N) \end{bmatrix} \qquad B = \begin{bmatrix} k_T(1,1) & \dots & k_T(1,D) \\ \vdots & \ddots & \vdots \\ k_T(D,1) & \dots & k_T(D,D) \end{bmatrix} \tag{41}$$

We can then define the block matrix $R = B \otimes K$, where $\otimes$ represents the Kronecker product. We define the class of kernels which can be written in such a form as *separable kernels*.

$$R = \begin{bmatrix} \begin{bmatrix} k(x_1,x_1)k_T(1,1) & \dots & k(x_1,x_N)k_T(1,1) \\ \vdots & \ddots & \vdots \\ k(x_N,x_1)k_T(1,1) & \dots & k(x_N,x_N)k_T(1,1) \end{bmatrix} & \cdots & & \cdots & \\ \vdots & \ddots & \vdots & & \\ \dots & \cdots & \begin{bmatrix} k(x_1,x_1)k_T(D,D) & \dots & k(x_1,x_N)k_T(D,D) \\ \vdots & \ddots & \vdots \\ k(x_N,x_1)k_T(D,D) & \dots & k(x_N,x_N)k_T(D,D) \end{bmatrix} \end{bmatrix} \tag{42}$$

The kernel matrix R of dimension $ND \times ND$ can therefore be used in the vector-valued GP. In the simple case, where we assume independent output, we can set B to be the identity matrix (i.e. $k_T(i,j) = \delta_{ij}\ \forall i,j \in \mathbb{N}_D$, where δ is the Kronecker delta). This is the case we assume in the GPEC formulation in the main text. However, we can alternatively encode relatedness between outputs by selecting an appropriate k_T.

For example, Sheldon (2008) define a user-defined adjacency matrix of a graph where the nodes of the graph represent tasks, and the edges represent task similarity. This allows the user to encode *a priori* relationships between outputs. Alternatively, Evgeniou et al. (2005) define k_T based on clustering, and enforce within-cluster similarity for tasks.

Any such methods can be used with GPEC by setting the base kernel k to be the WEG kernel defined in Section 4 and then selecting the desired task kernel k_T.

D IMPLEMENTATION DETAILS

D.1 Adversarial Sample Filtering for Multi-class Models

Following Karimi et al. (2019), we elect to sample from multi-class neural network decision boundaries by using a binary search algorithm on pairs of adversarial samples. Specifically, given a test-point $x_0 \in \mathbb{R}^D$ and model prediction $y = \text{argmax}_{k \in \mathcal{Y}} F(x_0)$, decision boundary points may be generated by the following procedure:

First, for each class $v \in \mathcal{Y}$ a set of M_v points is randomly sampled from the set of train points on which the model predicts class v:

$$\mathcal{X}_v \subseteq \{x : \text{argmax}_{k \in \mathcal{Y}}\ F(x) = v,\ x \in \mathcal{X}_{tr}\},\ |\mathcal{X}_v| = M_v \tag{43}$$

$\forall v \in \mathcal{Y}$. An untargeted adversarial attack using a given l_p norm and radius ϵ is generated for each point in $\mathcal{X}_y$, the set of points with the same class prediction as x_0. Each attack output $Attack_U(x, \epsilon) \in \mathbb{R}^D$ is paired with its corresponding input, resulting in the set

$$\mathcal{X}_{y'} = \{(x, Attack_U(x, \epsilon)) : x \in \mathcal{X}_y\}, \tag{44}$$

where for an element $(a, b) \in \mathcal{X}_{y'}$ we have $\text{argmax}_{k \in \mathcal{Y}}\ F(a) = y, \text{argmax}_{k \in \mathcal{Y}}\ F(b) = v \neq y$, where v is an unspecified class.

[4] To avoid confusion, we adopt the terminology of multi-task learning and refer to each explanation dimension $\{1, ..., D\}$ as *tasks*.

Likewise, a targeted adversarial attack, with target class y, is run on each point in each of the sets of points that are not predicted as class y. Each attack output $Attack_y(x, \epsilon) \in \mathbb{R}^D$ may be paired with its input x resulting in sets

$$\mathcal{X}_{v'} = \{(x, Attack_y(x, \epsilon)) : x \in \mathcal{X}_v\} \tag{45}$$

$\forall v \neq y \in \mathcal{Y}$. Here, for an element $(a, b) \in \mathcal{X}_{v'}$ we have $\text{argmax}_{k \in \mathcal{Y}}\ F(a) = v, \text{argmax}_{k \in \mathcal{Y}}\ F(b) = y$.

Thus, we have generated a diverse set of $\sum_{v \in \mathcal{Y}} M_v$ pairs of points that lie on opposite sides of the decision boundary for class y. The segment between any pair from a given set $\mathcal{X}_{v'}$ $v \neq y$ will necessarily contain a point on the class v v.s. class y decision boundary. Likewise, in the interest of further diversity, segments between any pair from the set $\mathcal{X}_{y'}$ will contain a point on the class v v.s. class y decision boundary, where $v \neq y \in \mathcal{Y}$ is unspecified. A binary search may be applied to each pair of samples to find the boundary point.

In practice, the entire procedure may be applied for all classes as a single post-processing step immediately after training. The results may be saved as a dictionary of boundary points which may be efficiently queried via the model predicted class of any given test point.

In our implementation, each adversarial attack is attempted multiple times, once using each radius value ϵ in the list: $[0.0, 2e^{-4}, 5e^{-4}, 8e^{-4}, 1e^{-3}, 1e^{-3}, 1.5e^{-3}, 2e^{-3}, 3e^{-3}, 1e^{-2}, 1e^{-1}, 3e^{-1}, 5e^{-1}, 1.0]$. For a given input, the output of the successful attack with smallest ϵ is used. If no attack is successful at any radius, the input is discarded from further consideration. We apply Projected Gradient Descent (PGD) (Madry et al., 2018) attacks with the l_∞ norm for both targeted and untargeted attacks, using the implementation provided in Rauber et al. (2017, 2020). The M_c values used for the relevant datasets are indicated below in Appendix E.1.

D.2 Geodesic Distance Approximation

We utilize the ISOMAP algorithm Tenenbaum et al. (2000) to approximate geodesic distances. We adapt the code[5] from Chen et al. (2019) in our implementation. Given a set of samples from a manifold, ISOMAP constructs a graph where each sample is node. Graph edges are populated by Euclidean distances between samples. After defining the graph, we use a shortest path length algorithm from NetworkX Hagberg et al. (2008) to approximate geodesic distance.

E EXPERIMENT SETUP

E.1 Datasets and Models

Census. The UCI Census dataset consists of 32,561 samples from the 1994 census dataset. Each sample is a single person's response to the census questionaire. An XGBoost model is trained using the 12 features to predict whether the individual has income $\geq$ \$50k.

Online Shopper. The UCI Online Shoppers dataset consists of clickstream data from 12,330 web sessions. Each session is generated from a different individual and specifies whether a revenue-generating transaction takes place. There are 17 other features including device information, types of pages accessed during the session, and date information. An XGBoost model is trained to predict whether a purchase occurs.

German Credit. The German Credit dataset consists of 1,000 samples; each sample represents an individual who takes credit from a bank. The classification task is to predict whether an individual is considered a good or bad risk. Features include demographic information, credit history, and information about existing loans. Categorical features are converted using a one-hot encoding, resulting in 24 total features.

MNIST. The MNIST dataset (LeCun and Cortes, 2010) consists of 70k grayscale images of dimension 28x28. Each image has a single handwritten numeral, from 0-9. A fully connected network with layer sizes 784-700-400-200-100-10 and ReLU activation functions was trained and validated on on 50,000 and 10,000 image label pairs, respectively. Training lasted for 30 epochs with initial learning rate of 2 and a learning rate decay of $\gamma = 0.5$ when training loss is plateaued. During adversarial example generation we used $M_y = 500$ and $M_c = 50$ $\forall c \neq y$.

Fashion MNIST. The Fashion MNIST dataset (Xiao et al., 2017) contains 70,000 grayscale images of dimension 28x28. There are 10 classes, each indicating a different article of clothing. We train a MLP model with the same

[5] https://github.com/redst4r/riemannian_latent_space

architecture used for the MNIST dataset, however we increase training to 100 epochs and increase the initial learning rate to 3. During adversarial example generation we used $M_y = 500$ and $M_c = 50$ $\forall c \neq y$.

CIFAR10 The CIFAR10 dataset (Krizhevsky et al., 2009) contains 60,000 color images of dimension 32x32. Each image contains an object from one of 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. We train a Resnet-18 model (He et al., 2015) for use in the experiments. During adversarial example generation we used $M_y = 500$ and $M_c = 50$ $\forall c \neq y$.

E.2 Competitor Implementation Details

BayesLIME and BayesSHAP. Slack et al. (2021) extend the methods LIME and KernelSHAP to use a Bayesian Framework. BayesLIME and BayesSHAP are fit using Bayesian linear regression models on perturbed outputs of the black-box model. The posterior distribution of the model weights are taken as the feature attributions instead of the frequentist estimate that characterizes LIME and KernelSHAP. We take the expected value of the posterior distribution as the point estimate for feature attributions, and the 95% credible interval as the estimate of uncertainty. To implement BayesLIME and BayesSHAP we use the public implementation[6]. We set the number of samples to 200, disable discretization for continuous variables, and calculate the explanations over all features. Otherwise, we use the default parameters for the implementation.

CXPlain. Schwab and Karlen (2019) introduces the explanation method CXPlain, which trains a surrogate explanation model based on a causal loss function. After training the surrogate model, the authors propose using a bootstrap resampling technique to estimate the variance of the predictions. In our experiments we implement the publicly available code[7]. We use the default parameters, which include using a 2-layer UNet model Ronneberger et al. (2015) for the image datasets and a 2-layer MLP model for the tabular datasets. We take a 95% confidence interval from the bootstrapped results as the estimate of uncertainty.

E.3 Regularization Parameter Overview

L_2 regularization. Let f be a neural network with parameters $\hat{\theta} = \arg\min_\theta \ell(y, f_\theta) + \lambda||\theta||_2^2$, where ℓ is a given loss function. The component $\lambda||\theta||_2^2$ adds a penalty for the magnitude of the parameters θ, which is controlled by parameter λ. In our experiments we increase λ to increase the regularization of the model f. Increasing λ encourages the model to have smaller values of θ, which results in a lower complexity model. We can see that $\lim_{\lambda\to\infty} \arg\min_\theta \ell(y, f_\theta) + \lambda||\theta||_2^2$ becomes the zero vector, which implies that the model becomes linear.

Softplus β. For a given neural network f with ReLU activation functions, we replace the ReLU functions with the Softplus function: $\text{Softplus}(x;\beta) = \frac{1}{\beta}\log(1+\exp(\beta x))$. The Softplus function is a smooth approximation of ReLU, which has been previously investigated (Dombrowski et al., 2019; Wang et al., 2020) in the context of improving neural network smoothness. Smoothness regularizers have been shown to reduce the complexity of neural networks and improve generalization (Rosca et al., 2020). Decreasing β increases the smoothing effect of the Softplus, which reduces the complexity of the model.

γ parameter for XGBoost. We increase the γ parameter in the XGBoost loss function (Eq. (2) in Chen and Guestrin (2016)):

$$\mathcal{L}(\phi) = \sum_i l\left(\hat{y}_i, y_i\right) + \sum_k \Omega\left(f_k\right)$$
$$\text{where } \Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\|^2$$

The $\Omega(f_k)$ component regularizes the complexity of the XGBoost model, which is an ensemble of functions f_k. The γ parameter penalizes the magnitude of T, which represents the number of leaves in each tree. Reducing the number of leaves in each tree function corresponds to smaller trees with less complexity. Therefore, increasing γ results in XGBoost models reduces the overall complexity of the model.

[6] https://github.com/dylan-slack/Modeling-Uncertainty-Local-Explainability
[7] https://github.com/d909b/cxplain

	Census	Online Shoppers	German Credit	MNIST	f-MNIST	CIFAR10
GPEC-WEG	0.11	0.37	0.07	12.90	18.15	1.77
GPEC-RBF	0.00	0.00	0.02	8.95	7.41	1.29
CXPlain	0.05	0.06	0.04	9.76	18.18	8.27
BayesSHAP	140.40	54.56	4.86	42,467	42,361	–
BayesLIME	91.29	54.60	4.83	41,832	41,992	–

Table 3: Inference time comparison (*in seconds*) for estimating the uncertainty for all features for 100 samples. For MNIST and f-MNIST datasets, results represent execution time for calculating uncertainty estimates with respect to all ten classes. CIFAR10 results were calculated using a single A100 GPU; all other results were calculated using CPU only. CIFAR10 results for BayesSHAP and BayesLIME are omitted due to computational expense

	Census	Online Shoppers	German Credit	MNIST	f-MNIST	CIFAR10
GPEC *(total)*	35.7	22.5	22.4	94.4	78.0	254
Sample DB	35.7	22.4	22.4	34.5	27.8	220
Naive-GP	0.02	0.02	0.02	7.03	7.11	1.29
CXPlain	1.07	0.34	1.71	3.54	3.73	3.85
BayesSHAP	–	–	–	–	–	–
BayesLIME	–	–	–	–	–	–

Table 4: Training time comparison (*in minutes*) for various explanation methods. The "Sample DB" step for GPEC is included for clarity and indicates the execution time for drawing samples from the black-box DB (for all classes). CIFAR10 results were calculated using a single A100 GPU; all other datasets were calculated on CPU only. Note that BayesLIME and BayesSHAP methods do not have a training step.

F ADDITIONAL RESULTS

In Section F.1 we show an execution time comparison for explanation uncertainty methods. In Section F.2 we perform a case study using GPEC on a diabetes prediction task. In Section F.3 we show illustrative examples from MNIST, f-MNIST, and CIFAR10. In Section F.4 we evaluate the sensitivity of GPEC to changes in parameters ρ and λ. In Section F.5 we perform an experiment evaluating function approximation uncertainty component of GPEC. In Section F.6 we report standard error and parameters for the regularization experiment in Section 5.3. In Section F.7 we show additional results from the Uncertainty Visualization Experiment.

F.1 Execution Time Results

In Table 3 we include inference time comparison for the methods implemented in this paper. Results are averaged over 100 test samples. For MNIST and f-MNIST datasets, results evaluate the time to calculate uncertainty estimates with respect to all classes. All experiments were run on an internal cluster using AMD EPYC 7302 16-Core processors. The CIFAR10 dataset was run using a single Nvidia A100 GPU; the other datasets were run on CPU only. We observe from the results that the methods that amortization methods (GPEC-WEG, GPEC-RBF, CXPlain) are significantly faster than perturbation methods BayesLIME and BayesSHAP.

In Table 4 we include training time results for the implemented methods. The "Sample DB" step for GPEC is highlighted for clarity and indicates the execution time for drawing samples from the black-box DB. The GP regression model in GPEC can be retrained with different hyperparameter choices (ρ, λ) and/or training samples (X) using the same DB samples, therefore this step only needs to be performed once for each given dataset and black-box model combination. This step is dependent on DB sampling algorithms (see App. D.1); improvements in these algorithms will decrease training time for GPEC. Note that BayesLIME and BayesSHAP methods do not have a training step.

F.2 Case Study: Diabetes Prediction with GPEC uncertainty

In this section we evaluate how GPEC can be used to improve understanding of model predictions and feature attributions. We used the NHANES (Miller, 1973) 2013-2014 dataset, which is an annual survey conducted by the Center for Disease Control and Prevention (CDC) and the National Center for Health Statistics (NHCS). It contains demographic, dietary, health exam, and survey data for 3,329 patients. We follow Dinh et al. (2019) in training an XGBoost model to predict the incidence of type-2 diabetes using a pre-selected set of 27 features.

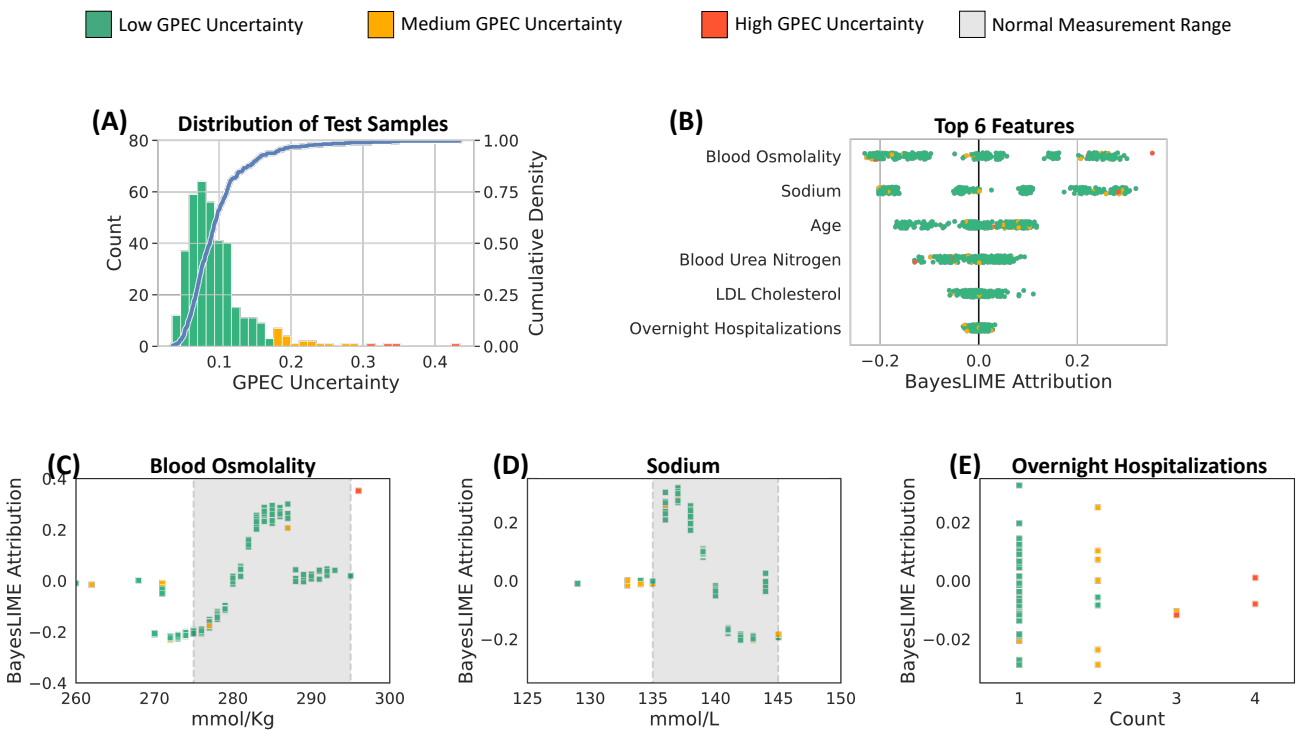

Figure 7: Case study of using GPEC to improve understanding for diabetes prediction. **(A)** We categorize samples into tertiles for low, medium, and high GPEC uncertainty. **(B)** We plot the top features using BayesLIME attributions while also overlaying the GPEC uncertainty (green, yellow, red). **(C) - (E)** We visualize three top features by value, importance, and uncertainty. Patients with high GPEC uncertainty for a given feature may require further investigation due to function approximation uncertainty and DB-aware uncertainty.

The model achieves an AUROC of 0.92, which replicates the results in Dinh et al. (2019). After training the model, we apply a feature attribution method, BayesLIME, to estimate the importance of each feature for a given patient's prediction of diabetes status. Since BayesLIME is a *local* feature attribution method, different patients may have different features (e.g. lab tests, physical attributes) that may be indicative of diabetes.

To establish a confidence estimate that includes *both* function approximation and DB-aware uncertainty, we apply GPEC over the test samples and categorize the samples in tertiles for low, medium, and high uncertainty (Fig. 7(A)). In Figure 7(B) we plot the distribution of test samples for three features of high overall importance for the incidence of diabetes, as calculated using BayesLIME attributions. We observe that the majority of patients have low uncertainty.

In Figures 7(C) - (E), we investigate three top features to with respect to feature value, feature importance, and GPEC uncertainty. In Figure 7(D), sodium, we see that there are 5 patients below the normal measurement range with medium uncertainty. For these patients, the BayesLIME attribution indicates that sodium level has minimal importance towards diabetes prediction, however the uncertainty score indicates that this explanation may not be reliable – more investigation is suggested.

In Figure 7(E), overnight hospitalizations, we see that having two hospitalizations is generally significant (higher magnitude of BayesLIME attribution), however these explanations have elevated uncertainty. The plot also indicates that having 3-4 hospitalizations generally has minimal impact on the prediction. This result is somewhat unexpected, and the high uncertainty suggests that these results should be investigated further. We hypothesize that there are relatively few patients with 2-4 overnight hospitalizations, leading to model overfitting and a higher GPEC uncertainty estimate.

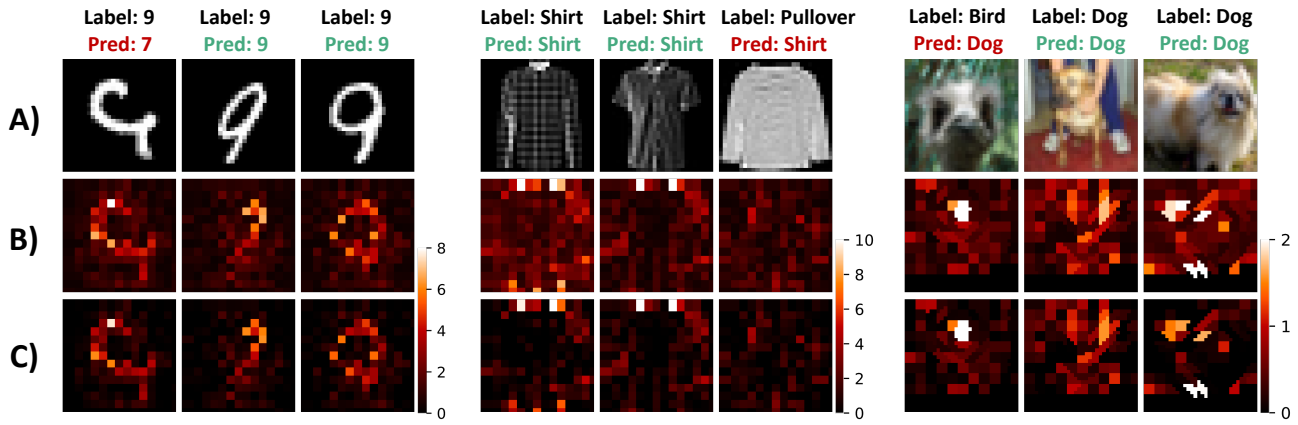

Figure 8: Illustrative samples from MNIST (left), f-MNIST (middle), and CIFAR10 (right). Row (A) shows the original sample. Row (B) visualizes the upper bound of the confidence interval (CI) for feature attributions using GPEC uncertainty. Row (C) visualizes the corresponding lower bound of the CI. We use BayesLIME attributions with GPEC uncertainty to cacluate the intervals. SLIC superpixels are used to improve the interpretability of results.

F.3 Illustrative Examples for MNIST, f-MNIST, and CIFAR10

We present illustrative examples from MNIST, f-MNIST, and CIFAR10 in Fig. 8. For each image, we apply BayesLIME to generate feature attributions, then apply GPEC to estimate a confidence interval. To improve interpretability of the results, we use the *simple linear iterative clustering* (SLIC) (Achanta et al., 2012) method, which clusters similar pixels into *superpixels*, and explain each superpixel rather than the individual pixels. The MNIST and f-MNIST datasetse us 196 superpixels, and the CIFAR10 dataset uses 96 superpixels. The upper and lower limit of the confidence interval is plotted in Fig. 8 row (B) and row (C). We observe that the GPEC confidence interval gives an estimate of uncertainty for the features for each image, which improves the interpretation of the feature attribution heatmap. The difference between upper and lower bounds is especially large for the background pixels in MNIST and f-MNIST datasets.

F.4 Sensitivity Analysis of WEG Kernel Parameters

The WEG kernel formulation uses two parameters, ρ and λ. The parameter ρ controls the weighting between each datapoint and the manifold samples. As ρ increases, the WEG kernel places more weight on manifold samples close in ℓ_2 distance to the given datapoint. The parameter λ acts as a bandwidth parameter for the exponential geodesic kernel. Increasing λ increases the effect of the geodesic distance along the manifold. Therefore decision boundaries with higher complexity will have an increased effect on the WEG kernel similarity. Bayesian model selection methods such as log marginal likelihood maximization (see Rasmussen and Williams (2005)) can be used for selecting hyperparameters ρ and λ. In practice, it is also important to select λ such that the EG kernel (Eq. (5)) is positive-definite (Feragen et al., 2015), which can be identified through cross-validation.

In Figure 9 we extend the kernel similarity analysis in Figure 3B to evaluate the WEG kernel for different DBs and ρ values. We observe that the similarity-ball $\{x' : k(x, x') \geq 0.9\}$ (blue) for points near complex DB are generally smaller in size. Increasing ρ increases sensitivity to nearby complex DB segments; i.e. values near complex DB segments will have correspondingly smaller similarity-balls for larger ρ.

In Figures 10, 11, and 12 we plot heatmaps for various combinations of ρ and λ parameters to evaluate the change in the uncertainty estimate. The black line is the decision boundary and the red points are the samples used for training GPEC. Please note that the heatmap scales are not necessarily the same for each plot.

F.5 Visualizing effects of Explainer Uncertainty in GPEC Estimate

In section 5.4 we evaluate GPEC's ability to combine uncertainty from the black-box decision boundary and the uncertainty estimate from BayesSHAP and SSV explainers. In Figure 13 we extend this experiment to evaluate

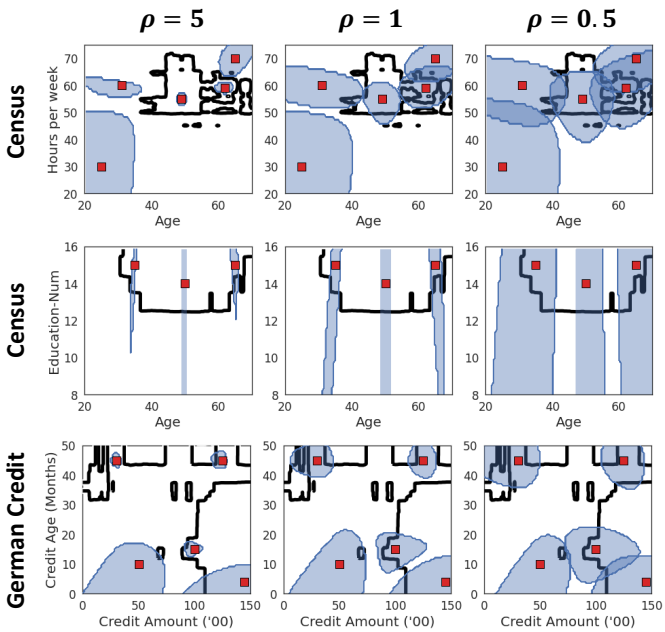

Figure 9: Evaluation of the WEG kernel for various DBs and values of ρ. The black line indicates the DB for the associated black-box model. The blue region highlights the set $\{x' : k(x, x') \geq 0.9\}$ for a given x (red). This region decreases in size when the local DB near x becomes more complex. Increasing the hyperparameter ρ increases the sensitivity of the WEG kernel similarity to DB complexity.

how well GPEC can capture the explainer uncertainty. We calculate the combined GPEC and explainer estimate using different numbers of approximation samples.

Both BayesSHAP and SSV depend on sampling to generate their explanations; having fewer samples increases the variance of their estimates. As we decrease the number of samples from 200 (Row A) to 5 (Row B) we would expect that the explainer uncertainty, and consequently the combined GPEC uncertainty, would increase. We see in Row Δ that the results follow our intuition; uncertainty increases for most of the plotted test points and uncertainty does not decrease for any points.

F.6 Regularization Experiment: Standard Error

In Table 5 and 6, we present the standard error measurements and parameters, respectively, for the regularization experiment.

F.7 Additional Results for Uncertainty Visualization Experiment

In Figure 14 we visualize the estimated explanation uncertainty as a heatmap for a grid of explanations. The generated plots only visualize the uncertainty for the feature on the x-axis. Due to space constraints, we list the results for the y-axis feature in the appendix, in Figure 14. We can see that the results are in line with those from the x-axis figure.

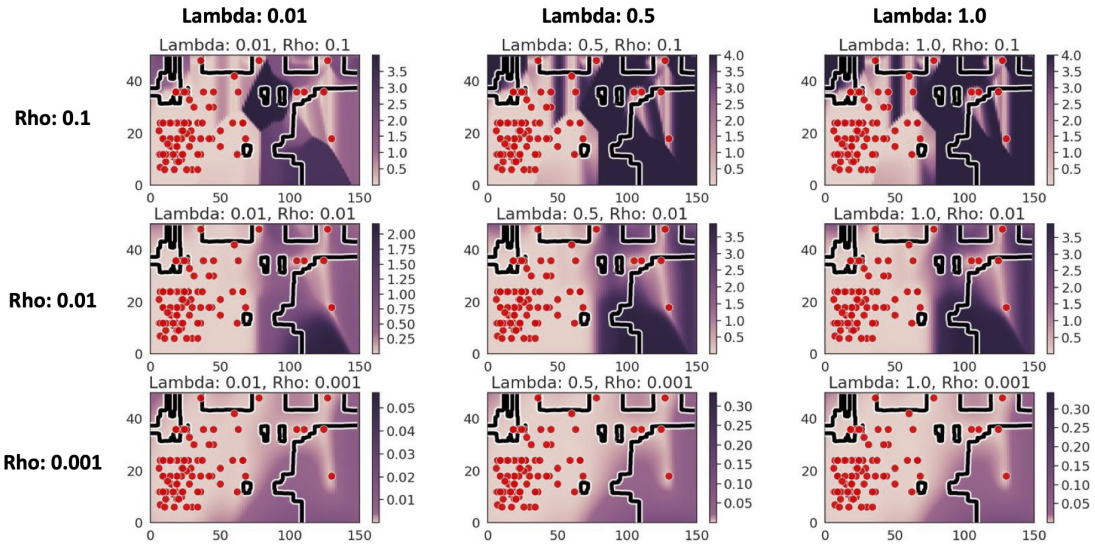

Figure 10: Hyperparameter sensitivity analysis for the German Credit Dataset. Heatmap of estimated uncertainty for the x-axis variable under different ρ and λ parameter choices.

Dataset	**Census**			**Online Shoppers**			**German Credit**			**CIFAR10**					
Regularization	γ			γ			γ			ℓ_2			Softplus β		
Magnitude	0	5	10	0	5	10	0	5	10	0	1e-5	10e-5	1.0	0.5	0.25
GPEC	0.037	0.037	0.036	0.017	0.013	0.010	0.026	0.030	0.012	2.8e-5	2.8e-5	2.8e-5	2.5e-5	2.5e-5	2.5e-5
Naive-GP	0.032	0.032	0.032	0.036	0.036	0.036	0.004	0.001	0.001	1.6e-3	1.6e-3	1.6e-3	1.6e-3	1.6e-3	1.6e-3
BayesSHAP	1.9e-4	1.9e-4	1.9e-4	1.9e-4	1.9e-4	1.9e-4	1.2e-4	1.0e-4	0.7e-4	–	–	–	–	–	–
BayesLIME	0.001	0.001	0.001	0.002	0.002	0.002	0.001	0.001	0.001	–	–	–	–	–	–
CXPlain	0.006	0.006	0.006	7.8e-5	9.9e-5	2.2e-4	2.2e-6	1.4e-6	6.9e-6	1.9e-7	5.7e-7	5.2e-7	2.8e-8	5.8e-8	1.7e-8

Dataset	**MNIST**						**Fashion MNIST**				
Regularization	ℓ_2			Softplus β			ℓ_2		Softplus β		
Magnitude	0	1e-5	10e-5	1.0	0.5	0.25	0	1e-5	10e-5	1.0	0.5	0.25
GPEC	3.2e-5	3.1e-5	3.1e-5	2.2e-6	2.2e-6	2.2e-6	8.3e-5	8.3e-5	8.1e-5	2.1e-6	1.9e-6	1.9e-6
Naive-GP	1.6e-7	1.6e-7	1.6e-7	2.3e-7	2.3e-7	2.2e-7	3.6e-7	3.6e-7	3.6e-7	3.6e-7	3.5e-7	3.6e-7
BayesSHAP	4.9e-4	3.1e-4	2.1e-4	4.5e-4	4.2e-4	4.2e-4	1.3e-3	0.5e-3	0.2e-3	0.002	0.002	0.002
BayesLIME	0.009	0.009	0.003	0.005	0.005	0.005	0.067	0.038	0.009	0.010	0.011	0.011
CXPlain	0.1e-5	5.0e-5	8.6 e-5	5.3e-5	8.0e-5	5.4e-5	7.2e-5	4.8e-5	6.2 e-5	9.0e-5	6.6e-5	9.6e-5

Table 5: Standard Error for results in Table 1

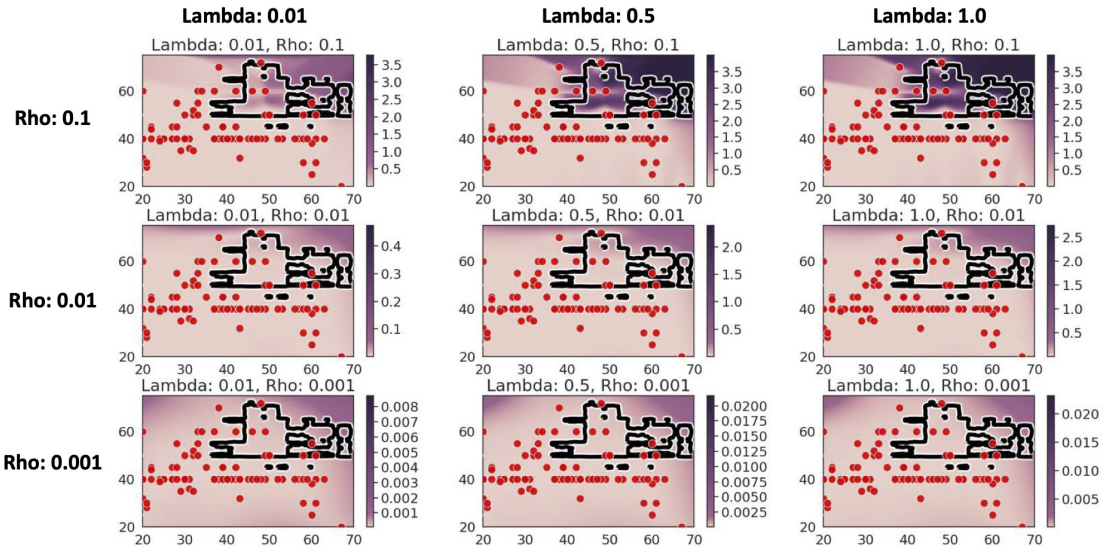

Figure 11: Hyperparameter sensitivity analysis for the Census Dataset. Heatmap of estimated uncertainty for the x-axis variable under different ρ and λ parameter choices.

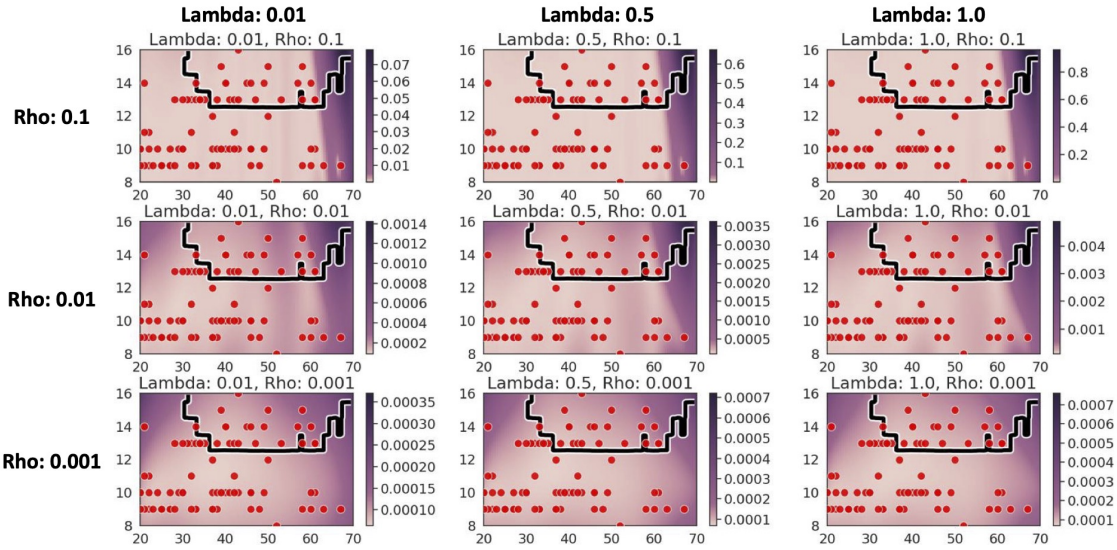

Figure 12: Hyperparameter sensitivity analysis for the Census Dataset. Heatmap of estimated uncertainty for the x-axis variable under different ρ and λ parameter choices.

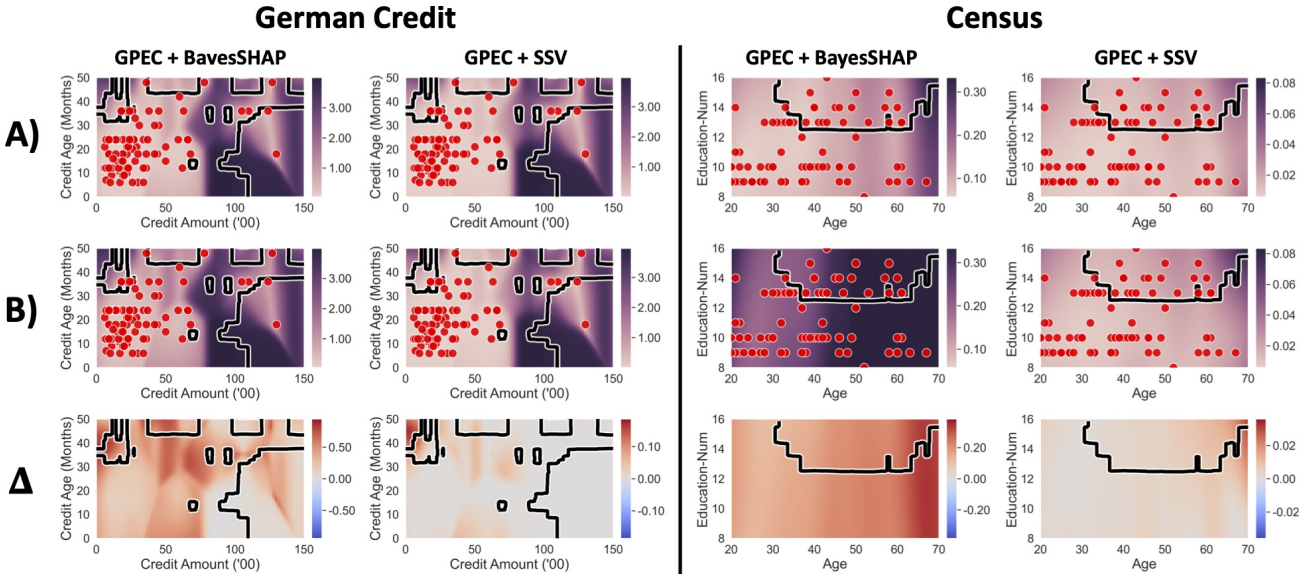

Figure 13: Comparison of the change in quantified uncertainty of explanations as we change the number of samples for BayesSHAP and SSV. Row (A) visualizes the combined uncertainty estimate using GPEC and either BayesSHAP or SSV, using 200 samples for approximating the BayesSHAP / SSV explanation. In Row (B) we decrease the number of samples to 5 and recalculate the estimated uncertainty. Row (Δ) represents the change in uncertainty estimate between (A) and (B). We see that the average uncertainty changes as we decrease the number of samples, which indicates that GPEC is able to capture the uncertainty arising from BayesSHAP / SSV approximation.

Dataset	λ	ρ
Census	1.0	0.1
Online Shoppers	1.0	0.1
German Credit	1.0	0.1
MNIST	1.0	0.01
f-MNIST	1.0	0.01
CIFAR10	1.0	0.05

Table 6: GPEC Parameters for results in Table 1

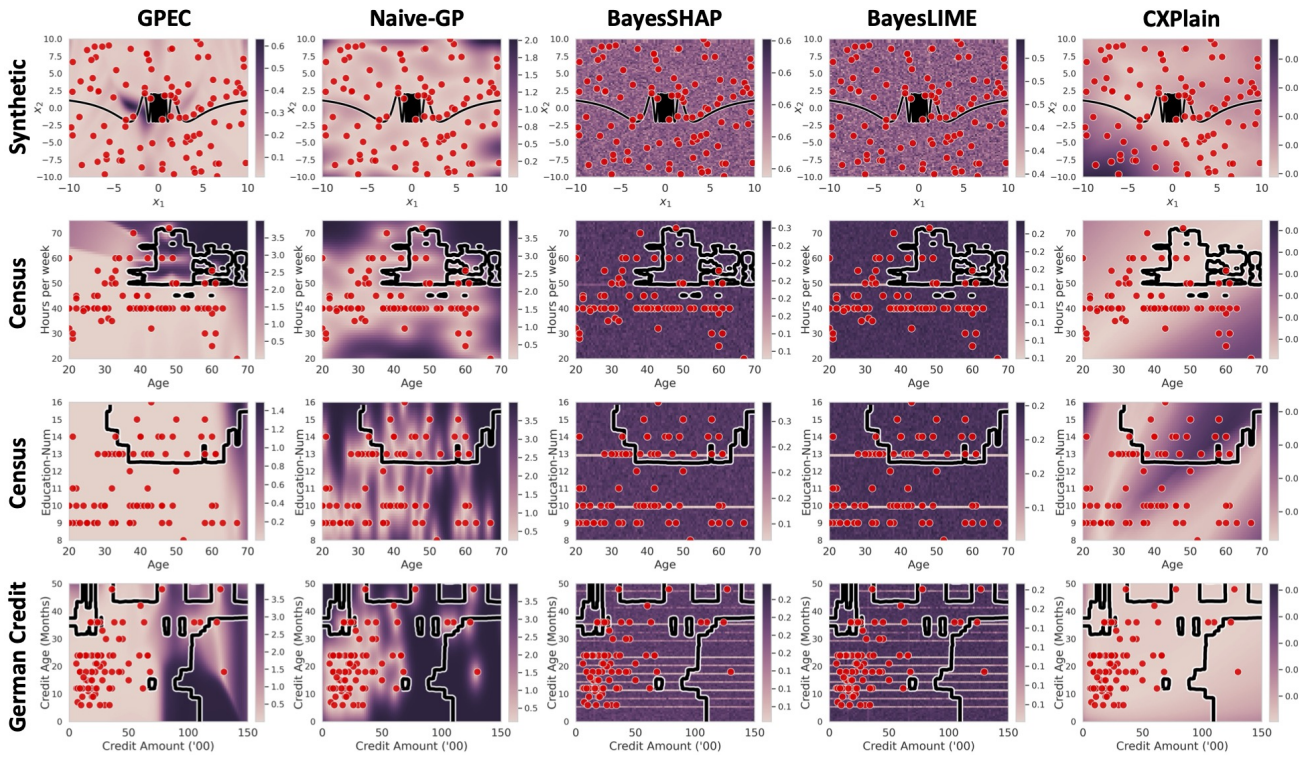

Figure 14: Complement to Figure 4. Visualization of estimated explanation uncertainty where the heatmap represents level of uncertainty for the feature on the **y-axis**.