INTERPRETING NEURAL NETWORKS THROUGH THE LENS OF HEAT FLOW

Anonymous authors
Paper under double-blind review

ABSTRACT

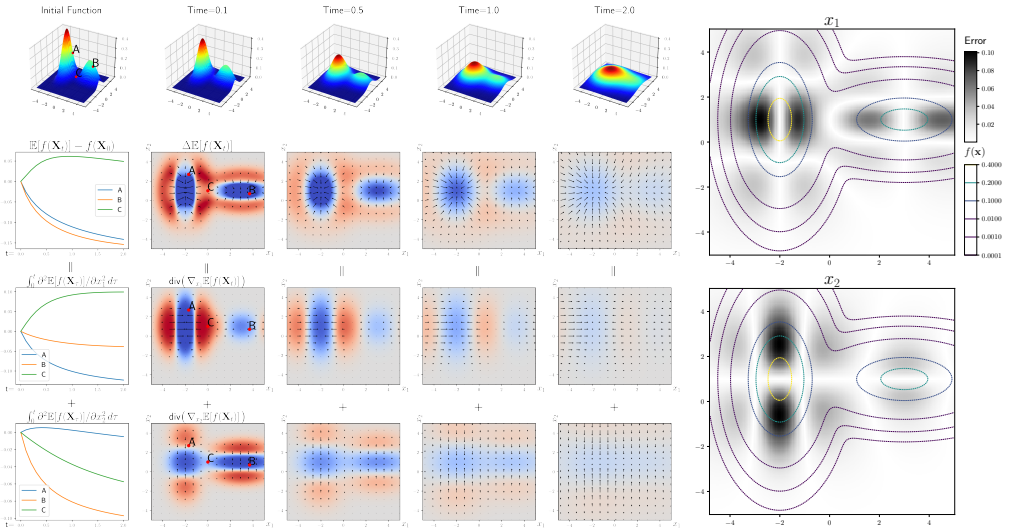
Machine learning models are often developed in a way that prioritizes task-specific performance but defers the understanding of how they actually work. This is especially true nowadays for deep neural networks. In this paper, we step back and consider the basic problem of understanding a learned model represented as a smooth scalar-valued function. We introduce HeatFlow, a framework based upon the heat diffusion process for interpreting the multi-scale behavior of the model around a test point. At its core, our approach looks into the heat flow initialized at the function of interest, which generates a family of functions with increasing smoothness. By applying differential operators to these smoothed functions, summary statistics (i.e., explanations) characterizing the original model on different scales can be drawn. We place an emphasis on studying the heat flow on data manifold, where the model is trained and expected to be well behaved. Numeric approximation procedures for implementing the proposed method in practice are discussed and demonstrated on image recognition tasks.

1 INTRODUCTION

In recent years, thanks to the growing availability of computation power and data, together with the rapid advancement of methodology, the machine learning community is witnessing the success of creating models with increasingly higher capacity and performance. Although scaling the model complexity is believed to be an effective approach towards stronger models, a downside of this trend is that it complicates the understanding of how the learned models work and why sometimes they fail. Such requirements for interpretability arise from both scientific research and engineering practices. For instance, carefully interpreting the working mechanism of a predictive model may help uncover its weakness in robustness, informing that further improvements should be made before its deployment in high-stakes decision-making.

In this paper, we consider the interpretation of scalar-valued smooth functions, a basic hypothesis class in machine learning. Models of this type arise naturally in regression and binary classification tasks that deal with continuous input features. Multi-output models, e.g., neural networks for multi-class classification, can be treated as such functions by investigating each output separately. While 1D and 2D such functions can be understood intuitively through graphical visualization, there is no straightforward way to visualize or even imagine general higher-dimensional functions. Fortunately, mathematicians developed the *derivative* to interpret functions in a pointwise manner. The directional derivative at a point measures the instantaneous rate of change of the function along a given direction, and the gradient gives the direction of steepest ascent. This forms the basis of popular gradient-based explanation methods for neural networks (Simonyan et al., 2014; Selvaraju et al., 2017; Sundararajan et al., 2017; Smilkov et al., 2017; Ancona et al., 2018; Erion et al., 2021; Xu et al., 2020; Hesse et al., 2021; Srinivas & Fleuret, 2021; Kapishnikov et al., 2021).

We argue that in order to interpret the outcome of a learned high-dimensional function at a test point, the gradient there is only part of the story because it just characterizes the first-order behavior of the function in an infinitesimal range. Such extreme localness is to blame for several known pitfalls of vanilla gradient-based interpretation. For example, if the point falls into a locally constant region, the gradient will be zero (Shrikumar et al., 2017). On the other hand, the gradient may change dramatically even for nearby points, leading to noisy and non-robust explanations in practice (Dombrowski et al.,



(a) Heat flow and the decomposition of Laplacian in two directions. (b) Interaction strength.

Figure 1: A toy example in $\mathbb{R}^2$.

2019; Wang et al., 2020). Moreover, the gradient will also be zero at different classes of critical points, suggesting the need for higher-order derivatives.

To this end, we introduce HeatFlow, an interpretation framework that enables summarizing the behavior of a learned model at different scales from the point of view of a test input. Our approach is motivated by a natural question to ask about a function f: How much does the value of f at a point $\mathbf{x}$ deviate from the average value of f in a neighborhood of $\mathbf{x}$? In our opinion, such deviation from local average is more comprehensible than instantaneous rate of change for non-mathematical audience. If the neighborhood is taken to be a small open ball centered at $\mathbf{x}$, then the answer is related to $\Delta f(\mathbf{x})$, where Δ is the Laplace operator, a fundamental second-order differential operator. To consider increasingly larger neighborhoods in a multi-scale manner, we propose solving a *heat equation*, a fundamental partial differential equations (PDE) studied in mathematics and physics.

We show that detailed interpretation for a function of interest may be achieved by extracting a rich set of principled summary statistics from the solution of the heat equation initialized at it. Furthermore, because a learned function is expected to be well-behaved only on the data manifold embedded in Euclidean space, it is possible to restrict the function on the manifold and solve the corresponding heat equation. In doing so, the interpretation problem is treated in a principled way based on the theory of differential geometry.

After giving a toy example, the remainder of this paper is organized as follows: Section 2 introduces preliminary knowledge about heat equation on Riemannian manifolds. The proposed method is studied in Section 3. Related work is summarized in Section 4. Section 5 reports experimental results. We conclude this work with Section 6.

A Toy Example. A toy example of understanding the sum of two Gaussian functions in a 2D Euclidean space is illustrated in Figure 1a. First row demonstrates the initial function and its heat flow. The heat flow generates a sequence of functions that are increasingly smoother than the initial one. Later we will see one possible representation of these functions as $\mathbb{E}[f(X_t)|X_0 = \mathbf{x}]$, in which f is the initial function and $\{X_t\}_{t>0}$ is a path of Brownian motion. Intuitively, the values of these smoothed functions at $\mathbf{x}$ are local average values of the initial function in increasingly larger neighborhoods centered at $\mathbf{x}$. The first subplot in the second row shows the deviation between the initial function and smoothed functions at three points, while the remaining subplots show Laplacian of smoothed functions along with their gradient fields. The deviation and the Laplacian are further decomposed into two directions, x_1 and x_2, presented in the third and forth rows, respectively. Intuitively, our core idea is to distribute the deviation caused by the heat flow to each input feature by decomposing the heat flow as the sum of "sub-flows" in corresponding directions. We show that the

proposed method further enables the detection of interaction strength between one variable and other variables, as demonstrated in Figure 1b.

2 PRELIMINARIES

In this section, we introduce some basic definitions and key concepts about the Laplacian and the heat equation (Hsu, 2002). Throughout our discussion, we consider a closed and connected manifold $\mathcal{M}$ endowed with a Riemannian metric g. A function $f : \mathcal{M} \to \mathbb{R}$, being the model to be explained through a heat diffusion process, is assumed to be twice-differentiable.

2.1 THE LAPLACIAN ON A RIEMANNIAN MANIFOLD

The *Laplace-Beltrami* operator, $\Delta_{\mathcal{M}}$, is a linear operator on function spaces that generalizes the ordinary Laplacian, Δ, to Riemannian manifold. There are multiple equivalent ways of introducing this operator (Hsu, 2002, Section 3.1).

- Divergence of the gradient field: $\Delta_{\mathcal{M}} f = \operatorname{div} \operatorname{grad} f$. The gradient $\operatorname{grad} f$ is the unique vector field on $\mathcal{M}$ that satisfies $\langle \operatorname{grad} f, X \rangle_g = df(X) = Xf$ for any vector field X, where the differential df gives the directional derivative of f along X. The divergence $\operatorname{div} X$ of a vector field X measures how much it locally behaves like a sink or source.
- Trace of the Hessian: $\Delta_{\mathcal{M}} f = \operatorname{tr}(\nabla^2 f)$. In Euclidean space $\mathbb{R}^n$, Hessian $\nabla^2 f$ is a matrix of the second partial derivatives, hence $\Delta f = \sum_i \frac{\partial^2}{\partial x_i^2} f$. In Riemannian manifold with its Levi-Civita connection ∇, Hessian is the second covariant derivative, such that given local coordinates $\{x^i\}$, a local expression for the Hessian tensor is $\nabla^2 f = \left(\frac{\partial^2 f}{\partial x^i \partial x^j} - \Gamma_{ij}^k \frac{\partial f}{\partial x^k}\right) dx^i \otimes dx^j$, where Γ_{ij}^k are the Christoffel symbols of connection.
- Hodge Laplacian: $\Delta_{\mathcal{M}} f = -d^* df$. In exterior calculus, the codifferential d^* is adjoint to the differential d by $\int f d^* \theta dV_g = \int \langle df, \theta \rangle_g dV_g$, where V_g is the Riemannian volume. When acting on 1-forms, d^* can be viewed as a generalization of the divergence as $d^*(\alpha_X) = -\operatorname{div}(X)$ where α_X is the dual of X by $\alpha_X(Y) = \langle X, Y \rangle_g$ and hence $\Delta_{\mathcal{M}} f = \operatorname{div} \operatorname{grad} f = -d^* df$. The *Hodge* Laplacian for differential forms, given by $\Box_{\mathcal{M}} = -(d^* d + dd^*)$, coincides with $\Delta_{\mathcal{M}} f$ when acting on smooth functions, i.e. 0-forms.
- Infinitesimal generator that generates Brownian motion on $\mathcal{M}$: roughly speaking, it means $\frac{1}{2}\Delta_{\mathcal{M}} f(x) = \lim_{t \downarrow 0} \frac{1}{t}(\mathbb{E}[f(X_t)|X_0 = \mathbf{x}] - f(\mathbf{x}))$, where $\{X_t\}_{t>0}$ is a Brownian path.

2.2 HEAT EQUATION

With the above Laplacian defined, we can now introduce the most basic diffusion process governed by the following PDE, which describes how an initial heat distribution $f : \mathcal{M} \to \mathbb{R}$ looks after being diffused for time $t > 0$:

$$\frac{\partial}{\partial t} u(t, \mathbf{x}) = \Delta_{\mathcal{M}} u(t, \mathbf{x}), \tag{1}$$

$$u(0, \mathbf{x}) = f(\mathbf{x}). \tag{2}$$

The solution of the heat equation at time t, $u(t, \mathbf{x})$, can also be given in multiple equivalent ways:

- Convolution: $u(t, \mathbf{x}) = \int_{\mathcal{M}} k_t(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) d\mathbf{y}$, in which $k_t(\mathbf{x}, \mathbf{y})$ is the fundamental solution of this PDE, known as the *heat kernel*. When the domain is Euclidean, the heat kernel is just a Gaussian centered at the point $\mathbf{x}$.
- Expectation: $u(t, \mathbf{x}) = \mathbb{E}[f(X_t)|X_0 = \mathbf{x}]$. Interestingly, the heat kernel is also the transition density function of Brownian motion on $\mathcal{M}$ (Hsu, 2002, Proposition 4.1.6). Hence, the expectation of function value over ends of Brownian paths also yields the solution on a stochastically complete manifold (Hsu, 2002, Proposition 4.2.1). Such a stochastic representation of PDE solutions is known as the *Faynman-Kac* formula (Karatzas & Shreve, 2012, Section 4.4).
- Gradient flow (Santambrogio, 2016): It is well-known that the solution is the gradient flow for the Dirichlet energy $\int \| \operatorname{grad} f \|_g^2 dV_g$ in $L^2(\mathcal{M})$ space. Consequently, the Dirichlet energy

monotonically decreases in time under the heat flow, meaning that the smoothness of the solution is always increasing.

It is convenient to define the *heat operators* $\{P_t\}_{t>0}$ to represent the solution: $u(t, \mathbf{x}) = (P_t f)(\mathbf{x})$. The heat equation can be extended to the diffusion of tensor fields and differential forms using generalized Laplacian. For instance, heat equation on 1-forms may be defined with the Hodge Laplacian: $\frac{\partial}{\partial t}\theta(t, \mathbf{x}) = \Box_{\mathcal{M}}\theta(t, \mathbf{x})$. Since $\Box_{\mathcal{M}}$ commutes with the differential d, if the initial condition is set to be a closed 1-form, i.e., the differential of a function: $\theta(0, \mathbf{x}) = df(\mathbf{x})$, then the solution will have an interesting connection with the heat equation on functions (Hsu, 2002, Section 7.2):

$$\theta(t, \mathbf{x}) = d(P_t f)(\mathbf{x}). \tag{3}$$

It means that diffusing the differential of a function with Hodge Laplacian is equivalent to applying the differential operator to the solution of scalar heat equation.

3 Multi-Scale Interpretation based on Heat Diffusion

By treating the model to be explained as the initial heat distribution, we analyze the solution of the heat equation to summarize the model at different scales from the point of view of an input $\mathbf{x}$. The solution along has already given local weighted average value of the function centered at $\mathbf{x}$: $(P_t f)(\mathbf{x}) = \mathbb{E}[f(X_t)|X_0 = \mathbf{x}]$. The time t plays the role of vicinity quantification that defines the range of neighborhood through a stochastic view as the area reachable by Brownian motion starting at $\mathbf{x}$. For small value of t, the smoothed value is mainly affected by small neighborhood of $\mathbf{x}$, hence reflecting only local information of the explained model. While as t increases, the smoothed function tends to capture more global trends of the model. In this section, we show that it is possible to extract more detailed information from the solution by applying differential operators to $\{P_t f\}_{t>0}$.

The Manifold Hypothesis. The manifold hypothesis assumes that data in real world concentrates on a low-dimensional submanifold $\mathcal{M}$ embedded in the ambient space $\mathbb{R}^D$ of much higher dimension. The function to be explained is expected to behave consistently only on $\mathcal{M}$ despite being learned in $\mathbb{R}^D$. For a learned function $f : \mathbb{R}^D \to \mathbb{R}$, we may be interested in its mechanism in either the open world or a closed world. Arbitrary input in $\mathbb{R}^D$ will be allowed in the former setting, while the latter setting only accepts input on the data manifold. Thus, users should choose the appropriate target and manifold accordingly depending on the purpose of model interpretation. If the goal is to understand model behavior in the open world, f and $\mathbb{R}^D$ should be chosen; while $f_{|\mathcal{M}}$, the restriction of f on $\mathcal{M}$, and $\mathcal{M}$ are suggested if the analysis is restricted to the data manifold.

3.1 Summarizing Gradients

By Eq. 3 and the duality of differential and gradient, taking the gradient of $P_t f$ is equivalent to smoothing the gradient field of f through running a heat equation for vector fields up to time t. In Euclidean settings, the gradient of $P_t f$ is $\nabla P_t f(\mathbf{x}) = \mathbb{E}[\nabla f(X_t)|X_0 = \mathbf{x}]$, which is simply the locally averaged version of the initial gradient field. On more general manifolds, it does not make sense to add vectors living in different tangent spaces $T_{X_t}\mathcal{M}$, and we need the parallel transport (Lee, 2018, Chapter 4) term acting as a mechanism to connect nearby tangent spaces. There is a Feynman-Kac formula for this operation (Hsu, 2002, Theorem 7.2.1):

$$\operatorname{grad} P_t f(\mathbf{x}) = \mathbb{E}[M_t \tau_t \operatorname{grad} f(X_t)|X_0 = \mathbf{x}], \tag{4}$$

where τ_t is the stochastic parallel transport map that transfers vectors from the tangent space of X_t back to that of X_0 along the Brownian path, and M_t is related to the Ricci curvature tensor on $\mathcal{M}$. One can interpret Eq. 4 as collecting derivative information from neighborhood by sending Brownian particles that travel for a fixed time length, parallel transporting the gradients at the positions of particles back to the explained point and averaging them.

3.2 Attribution

Given a prediction made by a machine learning model, the *attribution* problem asks to quantify the contribution of each input feature to this prediction. The output of an attribution method serves as a sign of local feature importance, indicating which features are more relevant to the model's

outcome. While the gradient is useful in terms of reflecting local feature importance, the quantities that it assigns to the features cannot be easily manipulated in a physically meaningful way. For an attribution method, it is more user-friendly to *distribute* the difference between the outcome and a baseline to input features, i.e., to give each feature a value with the same physical meaning and unit of measure as the model's output.

A natural baseline to choose in our framework is the local average $\mathbb{E}[f(X_t)|X_0 = \mathbf{x}]$. Instead of comparing to just one baseline, we are interested in attributing the differences between the model's outcome and a sequence of baselines resulting from increasing larger ts. Formally, given a model function $f : \mathcal{M} \subset \mathbb{R}^D \to \mathbb{R}$, input of interest $\mathbf{x}$, and the solution $P_t f$ of heat equation initialized at f, we attribute $(P_t f)(\mathbf{x}) - f(\mathbf{x})$ to input features $\{x_i\}_{i=1}^D$ at each t.

In order to allocate the difference to the coordinates of ambient space $\mathbb{R}^D$, we try to disaggregate the Laplace-Beltrami operator using the the standard orthonormal basis $\{\mathbf{e}_i\}_{i=1}^D$. Let $\mathcal{E}_i(\mathbf{x})$ be the orthogonal projection of the unit vector $\mathbf{e}_i$ onto the tangent space $T_{\mathbf{x}}\mathcal{M}$. Since the gradient field is tangential to $\mathcal{M}$, we have $\operatorname{grad} f = \sum_i \langle \operatorname{grad} f, \mathbf{e}_i \rangle_g \mathbf{e}_i = \sum_i \langle \operatorname{grad} f, \mathcal{E}_i \rangle_g \mathcal{E}_i$. The projection of $\operatorname{grad} f$ onto the ith dimension is achieved by $\operatorname{grad}_i f = \langle \operatorname{grad} f, \mathcal{E}_i \rangle_g \mathcal{E}_i = (\mathcal{E}_i f)\mathcal{E}_i$, and we define the dual form of $\operatorname{grad}_i f$ as $d_i f$ to be the "partial differential" operator. Further, we can compute the contribution of each feature i by taking the divergence of the projected vector field, $\operatorname{div}((\mathcal{E}_i f)\mathcal{E}_i)$. In this representation, the flow of heat along the ith dimension for an input $\mathbf{x}$ up to time T is defined as,

$$\operatorname{HeatFlow}_i^{\mathcal{M}}(\mathbf{x}, T, f) := \int_{t=0}^{T} \operatorname{div} \operatorname{grad}_i (P_t f)(\mathbf{x})\, dt. \tag{5}$$

A desirable property of HeatFlow is that the attributions add up to the difference between output of f and its local average as follows,

$$(P_T f)(\mathbf{x}) - f(\mathbf{x}) = \int_{t=0}^{T} \frac{\partial}{\partial t}(P_t f)(\mathbf{x}) dt = \int_{t=0}^{T} \Delta_{\mathcal{M}}(P_t f)(\mathbf{x}) dt = \sum_{i=1}^{D} \operatorname{HeatFlow}_i^{\mathcal{M}}(\mathbf{x}, T, f), \tag{6}$$

where the last equality is an immediate result of $\Delta_{\mathcal{M}} f = \operatorname{div} \operatorname{grad} f = \sum_{i=1}^D \operatorname{div}(\sum_i \langle \operatorname{grad} f, \mathcal{E}_i \rangle_g \mathcal{E}_i)$. Instead of computing div with respect to manifold, we can also globally express both div and grad using the standard basis $\{\mathbf{e}_i\}$, i.e., $\operatorname{grad} f = \sum_{i=1}^D (\mathcal{E}_i f)\mathbf{e}_i$ and $\operatorname{div} X = \sum_{i=1}^D \mathcal{E}_i X_i$ where X_i is the i-th component of vector field X expressed in ambient coordinates. This leads to another decomposition $\Delta_{\mathcal{M}} f = \sum_{i=1}^D \mathcal{E}_i^2 f$. Furthermore, through the trace of the Hessian expression, we also have $\Delta_{\mathcal{M}} f = \sum_{i=1}^D \nabla^2 f(\mathcal{E}_i, \mathcal{E}_i)$ (Hsu, 2002, Corollary 3.1.5). Together, we have the following three decompositions,

$$\sum_{i=1}^{D} \int_{t=0}^{T} \operatorname{div} \operatorname{grad}_i (P_t f)(\mathbf{x})\, dt = \sum_{i=1}^{D} \int_{t=0}^{T} \mathcal{E}_i^2 (P_t f)(\mathbf{x})\, dt = \sum_{i=1}^{D} \int_{t=0}^{T} \nabla^2 (P_t f)(\mathcal{E}_i(\mathbf{x}), \mathcal{E}_i(\mathbf{x}))\, dt. \tag{7}$$

It is worth noticing that while they are the same when summed over D dimensions, the three summands for each dimension i are not necessarily equal to each other. For simplicity of calculation, we adopt the last decomposition strategy based on the Hessian tensor in our experiments.

The discussion above assumes that we have access to a parameterized representation of the manifold $\mathcal{M}$. In practice, the manifold may be implicitly defined via a projection function that projects an arbitrary point onto the manifold through identifying its closest on-manifold counterpart in terms of Euclidean distance (Ruuth & Merriman, 2008; März & Macdonald, 2012). Given such a closest point function $\operatorname{cp}(\mathbf{x}) : \mathbb{R}^D \to \mathcal{M}$, it is possible to extend a function $f(\mathbf{x}) : \mathcal{M} \to \mathbb{R}$ to the surrounding space in $\mathbb{R}^D$ by defining a new function $f \circ \operatorname{cp}(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}$. It has been shown that the intrinsic operations of grad and $\Delta_{\mathcal{M}}$ are simplified as $\operatorname{grad} f(\mathbf{x}) = \nabla[f \circ \operatorname{cp}](\mathbf{x})$ and $\Delta_{\mathcal{M}} f(\mathbf{x}) = \Delta[f \circ \operatorname{cp}](\mathbf{x})$. The latter equation provides the fourth possibility to disaggregate the Laplace-Beltrami operator.

In Euclidean space, all the four decompositions just reduce to sum of the diagonal terms of the Hessian matrix. Hence, attribution for i-th dimension becomes integral of the second partial derivative:

$$\operatorname{HeatFlow}_i^{\mathbb{R}^D}(\mathbf{x}, T, f) := \int_{t=0}^{T} \frac{\partial^2}{\partial x_i^2}(P_t f)(\mathbf{x})\, dt. \tag{8}$$

3.3 PROPERTIES IN EUCLIDEAN SPACE

The evolvement of the model interpretation problem brings several desirable properties that a new attribution method should satisfy (Lundberg & Lee, 2017; Sundararajan et al., 2017; Friedman, 2004). In this section, we discuss some axioms satisfied by our method in Euclidean space. The following proposition includes four axioms defined in (Sundararajan & Najmi, 2020),

Proposition 1. $\mathrm{HeatFlow}^{\mathbb{R}^D}$ *satisfies the Dummy, Efficiency, Symmetry, and Linearity axioms.*

Informal Proof. Since the derivative of a function with respect to a dummy variable is always zero, the *Dummy* axiom is satisfied. By Eq. 6, the attributions always add up to the difference $(P_t f)(\mathbf{x}) - f(\mathbf{x})$ and hence proved the *Efficiency* axiom. For a function that is symmetric in two variables x_i and x_j, we prove in the Appendix that attribution $\partial^2 P_t f / \partial x_i^2 = \partial^2 P_t f / \partial x_j^2$ as long as $x_i = x_j$. The linearity is a consequence of the fact that the heat operator is linear.

Apart from these properties, we present below an interesting characteristics of our method that the resulting attribution also provides a measurement of how each variable contribute to the explained function independently.

Deviation from Additive Contribution. A family of interpretable models called generalized additive models (Hastie & Tibshirani, 2017; Chang et al., 2021) utilize sum of univariate functions where interaction between each feature is simple and additive. For a given neural network f, it is usually impossible to decompose f into sum of univariate functions. However, we are interested in quantifying the degree of deviation from being able to contribute univariately to f for each feature. As mentioned in Sec 3.2, a partial differentiation operator is defined as $d_i f$. Since Euclidean space is the focus of this section, we abuse the notations $df = \operatorname{grad} f$, $d_i f = \operatorname{grad}_i f$ and $d^* X = -\operatorname{div} X$. Since d and d^* are formal adjoints, we have $\operatorname{im} d \perp \ker d^*$ which suggests an orthogonal decomposition of the space of vector fields as $\operatorname{im} d \oplus \ker d^*$. Consequently, $d_i f$ can be rewritten as the sum of a gradient field, df_i, and an element that is divergence-free $d^* r_i = 0$. If one can find f_i whose gradient matches partial gradient of f everywhere, i.e. $df_i = d_i f$, contribution of feature i can be represented as an univarite function that interact with no other features. By solving the Poisson equation (Stern & Tettenhorst, 2019; Lim, 2022), $d^* df_i = d^* d_i f$, we are equivalently finding the closest univariate function, since f_i solves the minimization problem $\min_u \|du - d_i f\|$. We discover that when applied in Euclidean setting, our method is solving the aforementioned Poisson equation.

Proposition 2. $f_i = \mathrm{HeatFlow}_i^{\mathbb{R}^D}(\cdot, f, t)$ *solves the Poisson equation* $d^* df_i = d^* d_i (P_t f - f)$.

As a result, the difference between the gradient of the resulting attribution and the partial gradient of $P_t f - f$ measures the degree of deviation from being univariate for that feature i at time t. If feature i rarely interacts with other features, the following residuals should be small:

$$(r_i)_t = \|d\big(\mathrm{HeatFlow}_i(\cdot, f, t)\big) - d_i(P_t f - f)\|.$$

When this residual is zero, according to Proposition 4 in Appendix A.2, $P_t f - f$ can be decomposed into $f_i(x_i) + f_{-i}(\mathbf{x}_{-i})$ where contribution of x_i is independent. In Figure 1b, we visualize such residuals for the toy example. Interestingly, similar residuals are also discovered in a different setting where the Shapley value (Shapley, 1953) is employed for attribution (Kumar et al., 2021).

4 CONNECTION WITH EXISTING WORK

Close connections can be drawn between our method and many existing attribution approaches. We discuss in this section how our method generalizes the literature of attributing prediction of a model $f(\cdot)$ on an input $\mathbf{x}$ to its features.

- **Vanilla Gradient (Grad)** (Simonyan et al., 2014). The *Vanilla Gradient* is defined as $\mathrm{Grad}(x) = \nabla_{\mathbf{x}} f(\mathbf{x})$. It only characterizes first-order behavior of the model in an infinitesimal range.
- **Smooth Gradient (SG)** (Smilkov et al., 2017). Given a user-defined variance σ, the *Smooth Gradient* is defined as $\mathrm{SG}(\mathbf{x}) = \mathbb{E}_{\mathbf{z} \sim \mathcal{N}(\mathbf{x}, \sigma^2 \mathbf{I})} \nabla_{\mathbf{z}} f(\mathbf{z})$. It is a special case of our method in Euclidean settings: $\mathrm{SG}(\mathbf{x}) = \nabla_{\mathbf{x}}[(f * \mathcal{N}(0, \sigma^2 \mathbf{I}))(\mathbf{x})] = \nabla_{\mathbf{x}} u(t = \sigma^2/2, \mathbf{x})$.
- **Integrated Gradients (IG)** (Sundararajan et al., 2017). Given a user-defined baseline input $\mathbf{x}'$, *Integrated Gradient* of feature i is defined as $\mathrm{IG}_i(\mathbf{x}, \mathbf{x}') = (x_i - x_i') \cdot \int_0^1 \frac{\partial}{\partial x_i} f(\mathbf{x}' + \alpha(\mathbf{x} -$

$\mathbf{x}'))d\alpha$, where gradients are accumulated along the straight-line path in Euclidean space. Our method can also be viewed as expectation over multiple path integrals: $\mathbb{E}[f(X_t)|X_0 = \mathbf{x}] = \mathbb{E}[\int_{X_{[0,t]}} df|X_0 = \mathbf{x}]$.

- **Expected Gradients (EG)** (Erion et al., 2019). By introducing a distribution of baselines D, *Expected Gradients* is the expectation of path integrals defined as $\mathrm{EG}_i(\mathbf{x}, D) = \mathbb{E}_{\mathbf{x}'\sim D}\mathrm{IG}_i(\mathbf{x}, \mathbf{x}')$, where gradients are integrated over multiple paths. When a Gaussian baseline with variance σ is adopted $D = \mathcal{N}(\mathbf{x}, \sigma^2\mathbf{I})$, it coinsides with our method in Euclidean settings, explaining deviation from local average as $\sum_i \mathrm{EG}_i(\mathbf{x}, \mathcal{N}(\mathbf{x}, \sigma^2\mathbf{I})) = f(\mathbf{x}) - \mathbb{E}_{\mathbf{x}'\sim\mathcal{N}(\mathbf{x},\sigma^2\mathbf{I})}f(\mathbf{x}')$. When using the training data as the distribution of baselines, $D = D_{\text{data}}$, it is explaining the same thing as our method in manifold setting when $t \to \infty$ which is the difference from global average of all data.
- **Blur Integrated Gradients (BlurIG)** (Xu et al., 2020). *BlurIG* extends IG by considering the path of successively blurring the input image using Gaussian filter. It solves a heat equation in the pixel space of a single image as contrast to data space in our method.

A relevant literature from the perspective of generative models is identifying interpretable latent dimensions (Yang et al., 2021). In (Wang & Ponce, 2020), unsupervised discovery of interpretable axes is facilitated by exploring the latent space along the eigenvectors of the metric tensor defined by the decoder. Eigenvectors at different ranks encode qualitatively different type of changes. The main difference between our work and these prior literature is that our primary goal is to interpret a learned regression/classification model with the help of a generative model to explore the data manifold, while their goal is to explain the generative model itself.

5 Experiments

To illustrate usefulness of our multi-scale attribution methods, we demonstrate a sequence of attribution maps at discretized time steps on three image recognition tasks, including a synthetic image regression task, MNIST classification and facial age prediction[1]. We compare our method, HeatFlow, with four other methods, including **Grad** (Simonyan et al., 2014), **IG**, (Sundararajan et al., 2017) **SG** (Smilkov et al., 2017) and **BlurIG** (Xu et al., 2020). As for SG, increasing levels of Gaussian noises are applied and the resulting gradients are averaged over 100 samples. We denote noise level s as the amount of noise to add to the input as fraction of the total spread, i.e., $\max \mathbf{x} - \min \mathbf{x}$. As for BlurIG, partial accumulation of gradients along the blur path is calculated, where α denotes variance of the $2D$ Gaussian kernel. For all attribution maps of following figures, red and blue pixels denote positive and negative values, respectively.

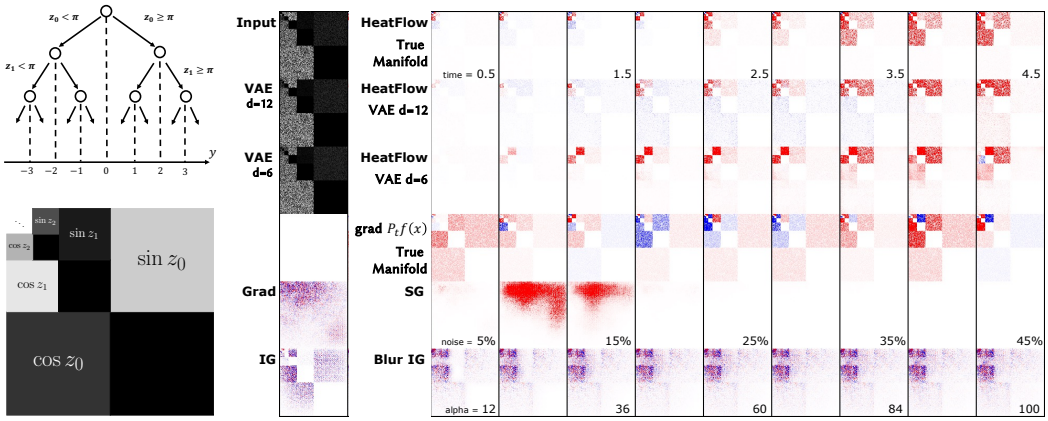
A brief discussion of the numerical implementation of our method in an end-to-end framework is presented in Appendix A.1, including detailed realization of manifold learning and high-dimensional PDE solving. In the following experiments, all data manifold is learnt by VAEs. Since our method involves many differentiation operations, all experiments are implemented using the JAX package (Bradbury et al., 2018) with Tesla V100 GPU. Further information, such as latent dimension of VAEs and image sizes are summarized in Appendix A.6.

5.1 Synthetic Experiments for Hierarchical Feature Structure

To study the ability of **HeatFlow** in uncovering different features with global or local effect, we design a synthetic dataset with hierarchical feature structure. A latent code $\mathbf{z} = [z_0, z_1, \cdots, z_{d-1}] \in [\mathrm{Uniform}(0, 2\pi)]^d$ is first sampled, based on which, labels and input images are generated as shown in Figure 2a. The underlying manifold is known, which is the Cartesian products of six circles. Local features are those pixels close to the upper-left corner, while global features are those of larger squares.

As shown in Figure 2b, vanilla **Grad** mostly highlights local features that are close to the upper-left corner. Both **Grad** and **SG** misleadingly highlight squares lying on the diagonal which stays constant among all images. For **IG** and **BlurIG**, more global features are concentrated on, however multi-scaled information is lacking. For **HeatFlow**, localness is meticulously controlled by heat diffusion on the manifold and when value of t is small, only small squares which correspond to local features

[1]A regression task for age estimation on face images using the UTKFace https://susanqq.github.io/UTKFace/ dataset



(a) Data generation.

(b) Heatmaps generated by various methods.

Figure 2: (a) **Top**: Generate labels. **Bottom**: Generate images. (b) Comparison of HeatFlow, Grad, SG, IG with black baseline and BlurIG on synthetic data. HeatFlow under true manifold, manifold learnt by VAE with latent dimension $d = 12$ and $d = 6$ are presented in the first three rows, respectively. On the fourth row, gradients on the true manifold is collected at each time step. The reconstructed inputs are shown below the original input. Vanilla Grad and IG are presented below the reconstructed inputs. For HeatFlow, SG and BlurIG, different time steps, noise levels of Gaussian blur, and partial integration up to kernel width α are shown, respectively.

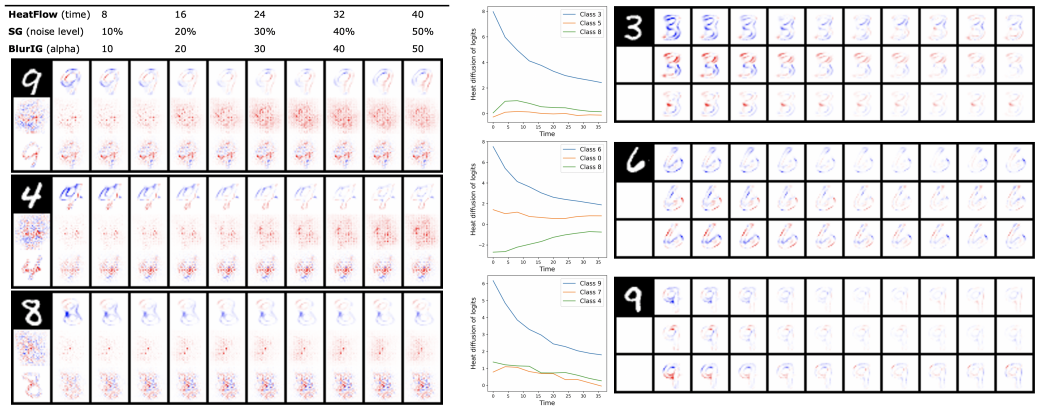
are highlighted; as value of t grows, more global pixels, i.e. larger squares are gradually shown. Furthermore, highlighted pixels tend to appear as an organized group in **HeatFlow**, suggesting its ability to attribute features with correlated contributions.

Manifold Mismatch. To study the reliance of alignment between the learned manifold and the true manifold, an ablation experiment is conducted where **HeatFlow** is run separately using the true underlying manifold and manifold learned by VAE with latent dimensionality $d = 6$ and $d = 12$, shown in first three rows of Figure 2b. It is observed that, when the correctness of the manifold increases, more detailed information is captured since up to second smallest square is attributed for VAE with $d = 12$ and up to third for VAE with $d = 6$. The multi-scale property behaves as expected regardless of degree of manifold mismatch. Although VAEs usually can not recover the true manifold exactly since the learned decoder is not guaranteed to be an immersion, the heat kernel is known to be stable under perturbations of the underlying manifold (Sun et al., 2009). Intuitively, such stability comes from the Brownian motion interpretation of heat flow: small perturbations will only affect a subset of the infinitely many Brownian paths.

5.2 MNIST AND UTKFACE

The same experiments are conducted on the MNIST classification task, where in Figure 3a, logits output of a neural network with 98% test accuracy is diffused over time. Saliency maps are particularly noisy for vanilla **Grad**. **SG** and **Blur IG** also fail to provide succinct information as Gaussian noise/filter operating in Euclidean space introduce unreliable evaluations of gradients over off-the-manifold samples. For **HeatFlow**, concise attribution with pixels only on digit area is highlighted. It is also evident that model focuses on different pixels as heat flows. More interesting findings can be drawn if we study heat flows for logits of digits that are similar. In Figure 3b, we present heat flows and multi-scale attribution maps for three digits, $3, 6, 9$, and compare them with their visually similar digits, $\{5, 8\}$, $\{0, 8\}$, $\{7, 4\}$, respectively. It is observed that, in the last column, only pixels corresponding to the common features of the compared digits remain. At small t, pixels that separate digits apart are highlighted.

As for explaining the facial age prediction task, results are presented in Appendix A.8. Our method managed to locate structured features such as eye brows, wrinkles, whether smile with teeth shown as important information for prediction of ages, where other methods failed to discover. We also



(a) Multi-scale attributions on MNIST.

(b) HeatFlow for different logits.

Figure 3: (a) Heat maps of the logit of label class on MNIST test samples. Original input is shown in the most upper-left corner. Vanilla Grad and IG are presented below the input. HeatFlow, SG and BlurIG with increasing level of time, noise and kernel width is shown on each row started from the second column, respectively. (b) Heat diffusion for MNIST samples comparing logits of different classes. **Left**: Change in function value. **Right**: HeatFlow attribution maps.

emphasize that multi-scale information is summarized in our method, such that attributed features which remains for large value of t represent more global effects, such as wrinkles.

Quantitative Evaluation. We also include a quantitative evaluation to compare HeatFlow with other methods using a strategy adopted in (Jethani et al., 2021). This evaluation strategy utilizes the remove/retain-and-retrain ideas where an evaluator was trained to predict the label given an arbitrary subset of features, and then, performance of the evaluator was assessed as the features were gradually excluded/included according to the absolute importance output by each explanation method. Using a set of 1000 test images for MNIST and 100 for UTKFace, accuracy and mean absolute error is assessed respectively as we include or exclude the most important features ranging from $0 - 100$, with curves visualized in Figure 4. It can be observed that HeatFlow is very competitive in terms of AUC, which indicates that HeatFlow is good at identifying important and discriminative features.

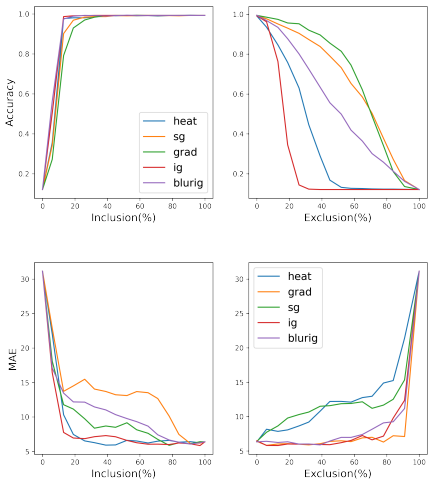


Figure 4: MNIST and UTKFace inclusion and exclusion curve. The change of accuracy for MNIST (upper) and mean absolute error for UTKFace (lower) as an increasing percentage of pixels attributed to be important are included (left) or excluded (right).

6 Conclusions

We have introduced a novel interpretation framework, HeatFlow, which generates multi-scale attributions for an interested input, explaining the differences between model's outcome and the local average values around that input. Our method generalizes many existing attribution approaches, combining satisfying properties of path methods, multiple baselines, and manifold hypothesis. Possible drawbacks of our method lie in the difficulty of manifold learning and the computational complexity of high-order derivatives, which we aim to improve in future work.

REFERENCES

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations*, 2018.

Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. In *International Conference on Learning Representations*, 2018.

Christian Beck, Sebastian Becker, Philipp Grohs, Nor Jaafari, and Arnulf Jentzen. Solving the kolmogorov pde by means of deep learning. *Journal of Scientific Computing*, 2021.

Julius Berner, Markus Dablander, and Philipp Grohs. Numerically solving parametric families of high-dimensional kolmogorov partial differential equations via deep learning. *Advances in Neural Information Processing Systems*, 2020.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, and Skye Wanderman-Milne. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Johann Brehmer and Kyle Cranmer. Flows for simultaneous manifold learning and density estimation. In *Advances in Neural Information Processing Systems*, 2020.

Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. How interpretable and trustworthy are gams? In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021.

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

Gabriel Erion, Joseph D Janizek, Pascal Sturmfels, Scott M Lundberg, and Su-In Lee. Learning explainable models using attribution priors. 2019.

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*, 2021.

Eric J. Friedman. Paths and consistency in additive cost sharing. *International Journal of Game Theory*, 32:501–518, 2004.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

Trevor J Hastie and Robert J Tibshirani. *Generalized additive models*. 2017.

Robin Hesse, Simone Schaub-Meyer, and Stefan Roth. Fast axiomatic attribution for neural networks. In *Advances in Neural Information Processing Systems*, 2021.

Elton P Hsu. *Stochastic analysis on manifolds*. Number 38. 2002.

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*, 2021.

Andrei Kapishnikov, Subhashini Venugopalan, Besim Avci, Ben Wedin, Michael Terry, and Tolga Bolukbasi. Guided integrated gradients: An adaptive path method for removing noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Ioannis Karatzas and Steven Shreve. *Brownian motion and stochastic calculus*, volume 113. Springer Science & Business Media, 2012.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Indra Kumar, Carlos Scheidegger, Suresh Venkatasubramanian, and Sorelle Friedler. Shapley residuals: Quantifying the limits of the shapley value for explanations. In *Advances in Neural Information Processing Systems*, 2021.

John M Lee. *Introduction to Riemannian manifolds*, volume 176. Springer, 2018.

Tongseok Lim. Stochastic integral representation of solutions to hodge theoretic poisson's equations on graphs, and cooperative value allocation of shapley and nash. 2022.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*. 2017.

Thomas März and Colin B. Macdonald. Calculus on surfaces with general closest point functions. *SIAM Journal on Numerical Analysis*, 50(6):3303–3328, 2012.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

Steven J. Ruuth and Barry Merriman. A simple embedding method for solving partial differential equations on surfaces. *Journal of Computational Physics*, 227(3):1943–1961, 2008. ISSN 0021-9991.

Filippo Santambrogio. {Euclidean, metric, and Wasserstein} gradient flows: an overview. *Bulletin of Mathematical Sciences*, 7:87–154, 2016.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, 2017.

Lloyd S. Shapley. A value for n-person games. In *Contributions to the Theory of Games*, 1953.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *In Workshop at International Conference on Learning Representations*. Citeseer, 2014.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. 2017.

Suraj Srinivas and Francois Fleuret. Rethinking the role of gradient-based attribution methods for model interpretability. In *International Conference on Learning Representations*, 2021.

Ari Stern and Alexander Tettenhorst. Hodge decomposition and the shapley value of a cooperative game. *Games and Economic Behavior*, 2019.

Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, volume 28, pp. 1383–1392. Wiley Online Library, 2009.

Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. In *International conference on machine learning*. PMLR, 2020.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*. PMLR, 2017.

Binxu Wang and Carlos R Ponce. A geometric analysis of deep generative image models and its applications. In *International Conference on Learning Representations*, 2020.

Zifan Wang, Haofan Wang, Shakul Ramkumar, Piotr Mardziel, Matt Fredrikson, and Anupam Datta. Smoothed geometry for robust attribution. In *Advances in Neural Information Processing Systems*, 2020.

Shawn Xu, Subhashini Venugopalan, and Mukund Sundararajan. Attribution in scale and space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Huiting Yang, Liangyu Chai, Qiang Wen, Shuang Zhao, Zixun Sun, and Shengfeng He. Discovering interpretable latent space directions of gans beyond binary attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12177–12185, 2021.

A APPENDIX

A.1 NUMERICAL IMPLEMENTATION

In this section, we briefly discuss the numerical implementation of our method in an end-to-end framework from manifold learning to multi-scale explanation. Since there are numerous approaches for manifold learning and high-dimensional PDE solving, we claim our implementation is only one of many possible realizations and can be generalized to different manifold settings for various purposes.

Manifold Learning. Deep generative models strive to infer probability distribution from observational data $\mathbf{x} \in \mathcal{X} = \mathbb{R}^D$, as well as learning the underlying data manifold (Brehmer & Cranmer, 2020; Arvanitidis et al., 2018). Approaches such as VAEs (Kingma & Welling, 2014; Rezende et al., 2014) and GANs (Goodfellow et al., 2014) makes the assumption of a d-dimensional data manifold, $\mathcal{M}$, embedded in the ambient space with $d < D$, through highly flexible generator, $g : \mathcal{Z} \to \mathcal{M}$, where $\mathbf{z} \in \mathcal{Z} = \mathbb{R}^d$ is the latent variable. The local Jacobian of the generator function, $\mathbf{J}_\mathbf{z} = \frac{\partial g}{\partial \mathbf{z}}|_{\mathbf{z}=\mathbf{z}}$, provides local basis in the input space and $\mathbf{G}_\mathbf{z} = \mathbf{J}_\mathbf{z}^T \mathbf{J}_\mathbf{z}$ is the Riemannian metric.

Heat Equation Solving. To solve the heat equation on manifold, especially a high-dimensional one, we utilize the Feynman-Kac formula and the framework proposed by (Beck et al., 2021; Berner et al., 2020) where one can simulate training data in order to learn solution $P_t f$ by means of deep learning. A supervised learning problem is constructed via the predictor variables $(\mathbf{x}, t)$ and the dependent target variables $y = f(X_t)|_{X_0=\mathbf{x}}$. The unique minimizer of the quadratic loss $\min_\phi \mathbb{E}[(\phi(\mathbf{x}, t) - y)^2]$ is the solution of the heat equation. In practice, the following empirical error is minimized:

$$\hat{\phi} = \arg\min_{\phi} \frac{1}{s} \sum_{i=1}^{s} (\phi(\mathbf{x}_i, t_i) - f(X_t)|_{X_0=\mathbf{x}_i})^2, \tag{9}$$

where $\{(\mathbf{x}_i, t_i)\}_{i=1}^s$ are realization of i.i.d. samples uniformly drawn from $\mathcal{M} \times [0, T]$. To simulate Brownian motion on a Riemannian manifold, we perform random walks over the learned manifold depending on the Laplacian adopted. In Euclidean setting, random walks are realized by adding Gaussian noise. While in curved space, one usually resort to geodesic random walks performed directly on latent variables. Detailed algorithms are presented in the Appendix.

Numerical Decomposition After heat equation solving, we take gradient of $\hat{\phi}(\mathbf{x}, t)$ with respect to $\mathbf{x}$ at different time levels, which corresponds to feature importance at different levels of localness. In addition, we need to explicitly realize the projection $\mathcal{E}_i$. When a local chart g is available, such as a generator function, we define $\mathbf{J}(\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T$ as the projection matrix. The resulting explicit forms of grad$f$ and $\mathcal{E}_i$ are $\operatorname{grad} f = \mathbf{G}^{-1}\partial f/\partial \mathbf{z}$, $\mathcal{E}_i = \mathbf{G}^{-1}\partial g_i/\partial \mathbf{z}$. By the definition of Christoffel symbol which describes the metric connection of manifold, it is easy to derive $\Gamma^i_{jk} = \sum_{m,b} \mathbf{G}^{-1}_{im} \mathbf{J}^b_m \mathbf{H}^b_{jk}$, where $\mathbf{H}^b_{jk} = \frac{\partial^2 g_b}{\partial z_j \partial z_k}$ is the Hessian matrix of the generator and $\mathbf{J}^b_m = \frac{\partial g_b}{\partial z_m}$ the Jacobian matrix. Following the definition of Hessian tensor, we have

$$\nabla^2 f(\mathcal{E}_i, \mathcal{E}_i) = \mathcal{E}_i^T \left(\frac{\partial^2 f}{\partial z_j \partial z_k} - \Gamma^i_{jk} \frac{\partial f}{\partial z_i} \right)_{jk} \mathcal{E}_i = \frac{\partial g_i}{\partial \mathbf{z}}^T \mathbf{G}^{-1} \left(\mathbf{H}_f - \mathbf{J}_f \mathbf{G}^{-1} (\sum_b \mathbf{J}^b \mathbf{H}^b) \right) \mathbf{G}^{-1} \frac{\partial g_i}{\partial \mathbf{z}},$$

where $\mathbf{H}_f$ and $\mathbf{J}_f$ are the Hessian and Jacobian matrix of the explained model f with respect to latent variables, respectively. Substitute this implementation into the third decomposition strategies in Eq. 7 for each $P_t f$ and then we obtain multi-scale attribution.

A.2 AXIOMATIC PROPERTIES IN EUCLIDEAN SPACE

Here we prove the Symmetry axiom in Proposition 1.

Proof. Let $f(x_1, ..., x_i, ..., x_j, ..., x_n)$ be a function of n variables, $f : \mathbb{R}^n \to \mathbb{R}$. Without loss of generality, let x_i and x_j be symmetric variables, i.e. $f(..., x_i, ..., x_j, ...) = f(..., x_j, ..., x_i, ...)$ for

$\forall x_i, x_j \in \mathbb{R}$. Then we have

$$
\begin{aligned}
&P_t f(..., x_i, ..., x_j, ...) \\
&= \int_y \frac{1}{(4\pi t)^{n/2}} \exp\Big(-\frac{1}{4t} \sum_{k=1}^{n} (x_k - y_k)^2 \Big) f(..., y_i, ..., y_j, ...) d\mathbf{y} \\
&= \int_y \frac{1}{(4\pi t)^{n/2}} \exp\Big(-\frac{1}{4t} \sum_{k \notin \{i,j\}}^{n} (x_k - y_k)^2 + (x_i - y_i)^2 + (x_j - y_j)^2 \Big) f(..., y_i, ..., y_j, ...) d\mathbf{y} \\
&= \int_y \frac{1}{(4\pi t)^{n/2}} \exp\Big(-\frac{1}{4t} \sum_{k \notin \{i,j\}}^{n} (x_k - y'_k)^2 + (x_i - y'_j)^2 + (x_j - y'_i)^2 \Big) f(..., y'_j, ..., y'_i, ...) d\mathbf{y}' \\
&= \int_y \frac{1}{(4\pi t)^{n/2}} \exp\Big(-\frac{1}{4t} \sum_{k \notin \{i,j\}}^{n} (x_k - y'_k)^2 + (x_j - y'_i)^2 + (x_i - y'_j)^2 \Big) f(..., y'_i, ..., y'_j, ...) d\mathbf{y}' \\
&= P_t f(..., x_j, ..., x_i, ...),
\end{aligned}
$$

where $\mathbf{y}'$ is the permutation of $\mathbf{y}$ by exchanging the i-th and j-th variables. The third equality follows from the change of variables formula and the fact that the determinant of a permutation matrix is ± 1. Hence, we draw the conclusion that the heat operator preserves the symmetry in Euclidean space. Next, further assuming that $x_i = x_j = a$ and according to the limit for second derivative, we have,

$$
\begin{aligned}
&\frac{\partial^2}{\partial x_i^2} f(..., a, ..., a, ...) \\
&= \lim_{h \to 0} \frac{f(..., a+h, ..., a, ...) - 2f(..., a, ..., a, ...) + f(..., a-h, ..., a, ...)}{h^2} \\
&= \lim_{h \to 0} \frac{f(..., a, ..., a+h, ...) - 2f(..., a, ..., a, ...) + f(..., a, ..., a-h, ...)}{h^2} \\
&= \frac{\partial^2}{\partial x_j^2} f(..., a, ..., a, ...).
\end{aligned}
$$

In other words, we have shown that the second derivatives of the symmetric function with respect to symmetric variables are equal when the values are equal. As a result, we have proved that $\partial^2 P_t f / \partial x_i^2 = \partial^2 P_t f / \partial x_j^2$ as long as $x_i = x_j$. □

A.3 Deviation from Additive Contribution and Link with Shapley Value

An interesting link exists between our method and the Hodge decomposition of a coorperative game (Stern & Tettenhorst, 2019). Given a set of D players and a value function $v : 2^{[D]} \to \mathbb{R}$, each coalition of players $S \subset 2^{[D]} = V$ can be considered as a vertex of a D-dimensional hypercube $G = (V, E)$, where each edge corresponds to addition of a single player $i \notin S$ to S. The gradient and divergence of exterior calculus operating on this graph, denoted by (d, d^*), are defined as $dv(S, S \cup \{i\}) = v(S \cup \{i\}) - v(S)$ and $(d^* f)(a) = \sum_{(b,a) \in E} f(b, a)$, respectively. $dv \in \ell^2(E)$ gives the marginal value contributed by a player joining a coalition. To specify marginal contribution of an individual player $i \in [D]$, a partial gradient $d_i : \ell^2(V) \to \ell^2(E)$ is defined as

$$
d_i v(S, S \cup \{j\}) = \begin{cases} v(S \cup \{j\}) - v(S), & i = j \\ 0, & \text{otherwise} \end{cases}.
$$

Notice the corresponding definition of partial differentiation operator in the continuous space adopted in our method, $d_i f = \mathrm{grad}_i f$. A key result in (Stern & Tettenhorst, 2019) is connecting inessentiality of games to the defined partial gradient operators in the following proposition, where in an inessential game, for all $S \subseteq V$, $v(S) = \sum_{i \in S} v(\{i\})$, meaning each player contributes a precise value $v(\{i\})$ to any coalition it participates in.

Proposition 3. *(Stern & Tettenhorst, 2019, Prop 3.3) A game is inessential if and only if $d_i v \in \mathrm{im}\, d$ for all $i \in [D]$.*

This means a game is inessential if one can find games v_i such that $d_i v = dv_i$ for each player i. In our setting, there is no notion of game but we consider in breaking down the given differentiable

model $f(\cdot)$ to additive univariate functions. This ability in separating each feature to contribute independently is the "inessentiality" of our problem.

Proposition 4. *If one can find a function f_i whose gradient matches partial gradient $\partial/\partial x_i$ of f everywhere, contribution of feature i can be represented as an univariate function, as $f(\mathbf{x}) = f_i(x_i) + f_{-i}(\mathbf{x}_{-i})$.*

Proof. Assume an arbitrary path over $\mathbb{R}^n$, $\mathbf{r}(t) = (x_1(t), \dots, x_n(t))^T$, such that the end points are $\mathbf{r}(0) = \mathbf{x}_a$ and $\mathbf{r}(1) = \mathbf{x}_b$, particularly, the i-th feature has value $(\mathbf{x}_a)_i = a$ and $(\mathbf{x}_b)_i = b$. By the fundamental theorem of calculus for line integrals, and if $\frac{\partial f}{\partial x_i} = \frac{df_i}{dx_i}$ for any $\mathbf{x}$, we have,

$$
\begin{aligned}
f(\mathbf{x}_b) - f(\mathbf{x}_a) &= \int_0^1 \frac{\partial f(\mathbf{r}(t))}{\partial \mathbf{x}} \cdot \frac{d\mathbf{r}(t)}{dt} dt \\
&= \int_0^1 \sum_{i=1}^n \frac{\partial f(\mathbf{r}(t))}{\partial x_i} \frac{dx_i(t)}{dt} dt \\
&= \sum_{j \neq i} \int_0^1 \frac{\partial f(\mathbf{r}(t))}{\partial x_j} \frac{dx_j(t)}{dt} dt + \int_0^1 \frac{\partial f(\mathbf{r}(t))}{\partial x_i} \frac{dx_i(t)}{dt} dt \\
&= \sum_{j \neq i} \int_0^1 \frac{\partial f(\mathbf{r}(t))}{\partial x_j} \frac{dx_j(t)}{dt} dt + \int_0^1 \frac{df_i(x_i(t))}{dx_i} \frac{dx_i(t)}{dt} dt \\
&= \sum_{j \neq i} \int_0^1 \frac{\partial f(\mathbf{r}(t))}{\partial x_j} \frac{dx_j(t)}{dt} dt + f_i(b) - f_i(a).
\end{aligned}
$$

With $\mathbf{x}_a$ set to $\mathbf{0}$ and assuming $f(\mathbf{0}) = 0$, we have $f(\mathbf{x}) = \sum_{j \neq i} \int_0^1 \frac{\partial f(\mathbf{r}(t))}{\partial x_j} \frac{dx_j(t)}{dt} dt + f_i(x_i)$. Since path $\mathbf{r}(t)$ is arbitrary meaning the equation holds for any path between $\mathbf{0}$ and $\mathbf{x}$, if the first term depends on x_i, then by differentiating both sides with respect to x_i, $\frac{\partial f}{\partial x_i} = \frac{df_i}{dx_i}$ would be contradicted and hence Proposition 4 proved. □

Next, we prove Proposition 2, such that our method $\mathrm{HeatFlow}_i^{\mathbb{R}^D}(\cdot, t, f)$ is solving the Poisson equation $d^* df_i = d^* d_i (P_t f - f)$.

Proof. First we emphasize that operator $d^* d_i$ and the heat operator P_t commute with each other since $d^* d_i$ commute with the Laplacian Δ as follows:

$$
\begin{aligned}
d^* d_i \Delta f &= \mathrm{div}\left(\frac{\partial}{\partial x_i} \sum_{k=1}^D \frac{\partial^2}{\partial x_k^2} f \right) \\
&= \frac{\partial^2}{\partial x_i^2} \sum_{k=1}^D \frac{\partial^2}{\partial x_k^2} f \\
&= \sum_{k=1}^D \frac{\partial^2}{\partial x_k^2} \frac{\partial^2}{\partial x_i^2} f = \Delta d^* d_i f.
\end{aligned}
$$

With this fact, we have

$$
\begin{aligned}
d^* d\Big(\mathrm{HeatFlow}_i(\mathbf{x}, t, f) \Big) &= d^* d \int_{\tau=0}^t d^* d_i (P_\tau f)(\mathbf{x}) d\tau \\
&= d^* d \int_{\tau=0}^t P_\tau (d^* d_i f)(\mathbf{x}) d\tau \\
&= \int_{\tau=0}^t \Delta P_\tau (d^* d_i f)(\mathbf{x}) d\tau \\
&= \int_{\tau=0}^t \frac{\partial}{\partial t} P_\tau (d^* d_i f)(\mathbf{x}) d\tau \\
&= P_t (d^* d_i f)(\mathbf{x}) - d^* d_i f(\mathbf{x}) = d^* d_i (P_t f - f)(\mathbf{x}).
\end{aligned}
$$

□

A.4 EXAMPLE ON MULTIVARIATE GAUSSIAN

Here, we present a simple example of our method, HeatFlow, on a d-dimensional multivariate Gaussian with mean $\boldsymbol{\mu}$ and diagonal covariance $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_d)$. Firstly, the heat kernel is calculated as follow,

$$
\begin{aligned}
P_t f(\mathbf{x}) &= \int k_t(\mathbf{x}, \mathbf{y}) \mathcal{N}(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma})\, dy \\
&= \int \frac{1}{(4\pi t)^{d/2}} \exp\big(-\frac{1}{4t}\|\mathbf{x}-\mathbf{y}\|^2\big) \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\big(-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{y}-\boldsymbol{\mu})\big)\, dy \\
&= \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma} + 2tI)
\end{aligned}
$$

Consequently, we have the contribution of the i-th variable as,

$$
\mathrm{HeatFlow}_i^{\mathbb{R}^D}(\mathbf{x}, T, f) := \int_{t=0}^{T} \frac{\partial^2}{\partial x_i^2}(P_t f)(\mathbf{x})\, dt = \int_{t=0}^{T} \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma} + 2tI|^{1/2}} \Big(-\frac{1}{\sigma_i + 2t}\Big)\, dt
$$

A.5 RANDOM WALK ON MANIFOLDS

To solve a high-dimensional heat equation, we need to resort to the formula of expectation over Brownian paths. In order to simulate Brownian motion on a Riemannian manifold, we resort to Algorithm 1, which is also adopted by (Arvanitidis et al., 2018).

Algorithm 1: Random walk on a Riemannian manifold: $\mathrm{RW}(\mathbf{z}, s, T, \mathbf{G})$

input : Latent starting point $\mathbf{z} \in \mathbb{R}^d$, step size s, number of steps T, metric tensor $\mathbf{G}$
output : Random walk path $\mathbf{Z} \in \mathbb{R}^{T \times d}$
for $t = 1$ *to* T **do**
  $\mathbf{L}, \mathbf{U} = \mathrm{eig}(\mathbf{G}_{\mathbf{z}})$, ($\mathbf{L}$: eigenvalues, $\mathbf{U}$: eigenvectors)
  $\mathbf{v} = \mathbf{U}\mathbf{L}^{-\frac{1}{2}}\boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_d)$,
  $\mathbf{z} = \mathbf{z} + s \cdot \mathbf{v}$,
  $\mathbf{Z}(t, :) = \mathbf{z}$
end

A.6 IMPLEMENTATION DETAILS

We summarize hyperparameter settings in our experiments in Table 1.

Table 1: Implementation details for the three experiments, synthetic, MNIST and UTKFace.

Hyperparameter	Value
Number of generated training samples	60000
Number of random walk steps	5000
Synthetic	
Image size	128×128
VAE latent dimension d	$6/12$
Step size s	0.1
Explained model performance (MSE)	$< 10^{-4}$
MNIST	
Image size	28×28
VAE latent dimension d	10
Step size s	0.02
Explained model performance (ACC)	$> 98\%$
UTKFace	
Image size	$200 \times 200 \times 3$
VAE latent dimension d	16
step size s	0.05
Explained model performance (MAE)	< 3.0

A.7 More Results for Synthetic Experiments

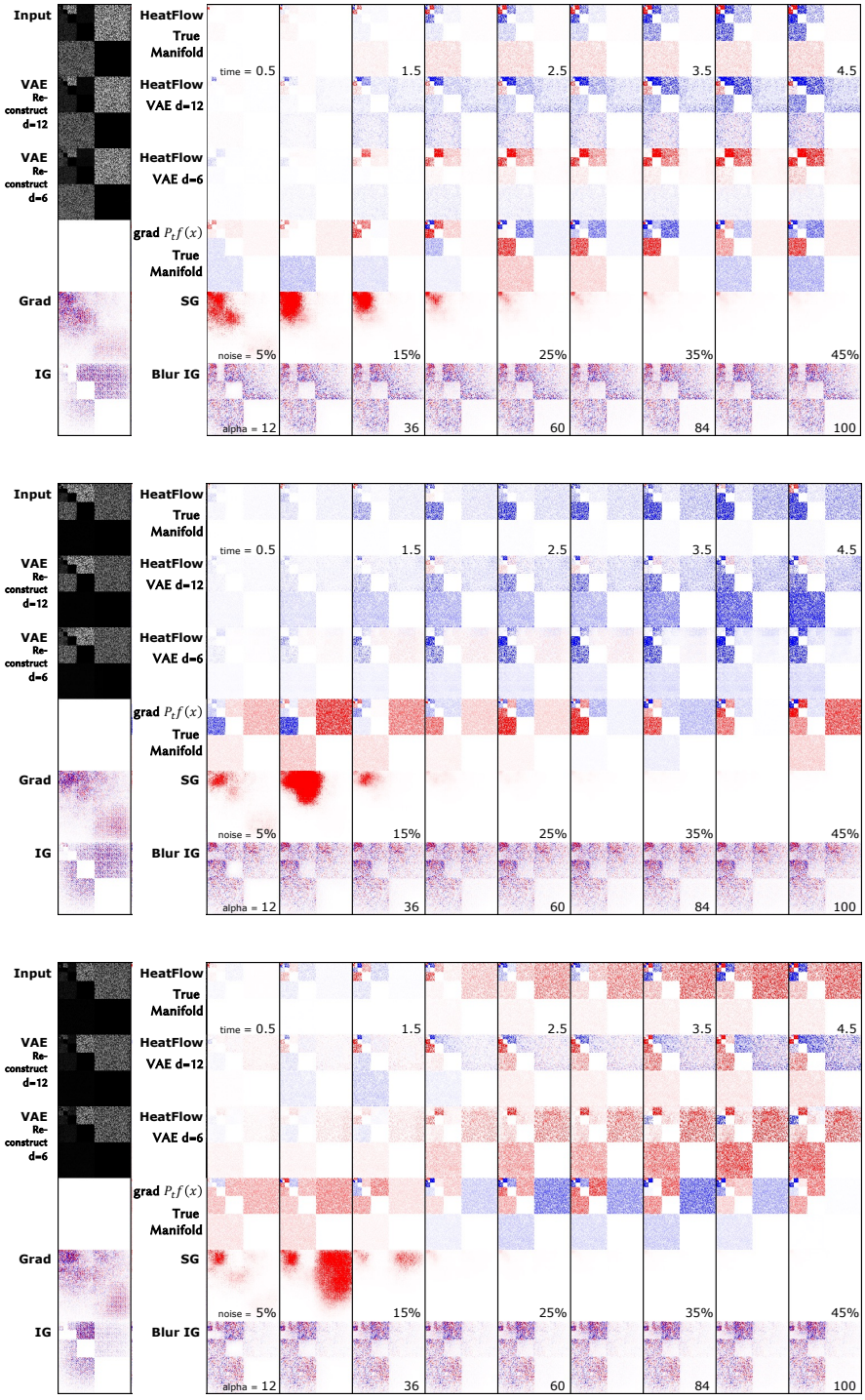


Figure 5: Three more examples for synthetic experiments. HeatFlow under true manifold, and learned manifold by VAE with latent dimension $d = 12$ and $d = 6$ are presented in the firt three rows, respectively. On the fourth row, gradients on the true manifold is collected at each time step. SG and Blur IG are presented on the last two rows.

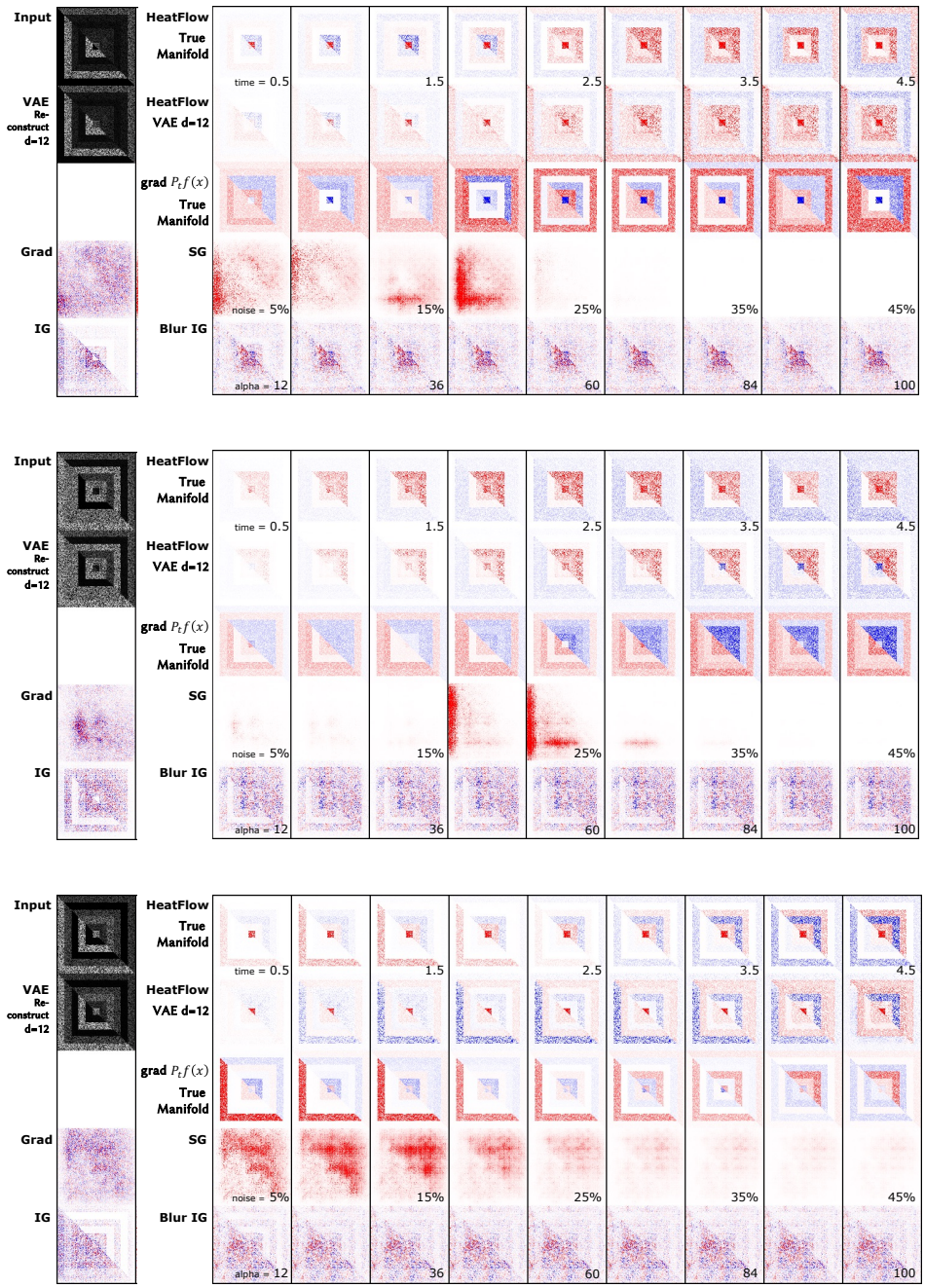


Figure 6: Three examples for another synthetic experiments. HeatFlow with true manifold, VAE with latent dimension $d = 12$ are presented in the first and second rows, respectively. On the thir row, gradients on the true manifold is collected at each time step. SG and Blur IG are presented on the last two rows.

A.8 RESULTS FOR FACIAL AGE PREDICTION

Figure 7: Five test examples for facial age prediction explanation. HeatFlow using VAE with latent dimension $d = 16$ are presented in the first row. Vanilla Grad and IG are shown below the original input. SG and Blur IG are presented on the second and thir rows.

A.9 ABLATION: COMPARING HEATFLOW ON MANIFOLD AND EUCLIDEAN SPACE

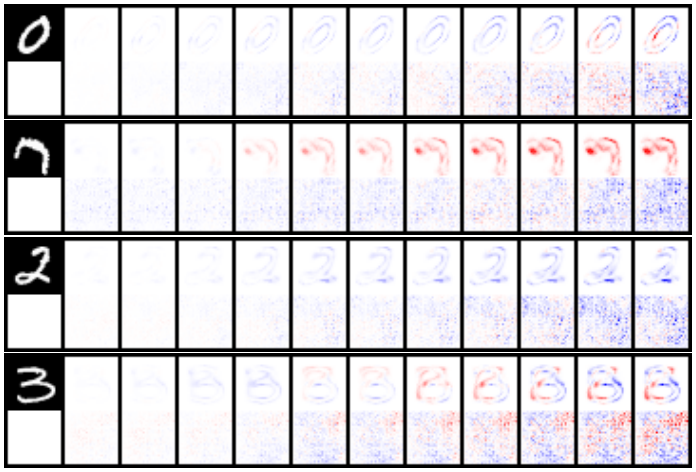


Figure 8: Four examples comparing HeatFlow on the learned manifold and Euclidean space $\mathbb{R}^{28\times 28}$. Original input is shown on the most upper-left corner. HeatFlow run on the learned manifold and Euclidean space is shown on the first and second rows, respectively.