# PEXP: A Scalable Parallel Tree-Based Framework for Interpreting Models on Big Data

Wen-Dong Jiang, *Graduate Student Member, IEEE*, Chih-Yung Chang, *Member, IEEE*, Tzu-Chia Huang, Yu-Ting Chin, and Diptendu Sinha Roy, *Senior Member, IEEE*

***Abstract*—The proliferation of Big Data has fueled the success of deep learning; however, its inherent opaque decision-making nature poses significant challenges for its adoption in safety-critical domains. Existing interpretable machine learning methods offer partial solutions but often struggle with model fidelity, inconsistent explanations, a lack of holistic model understanding, and critically, computational inefficiency, especially when applied to models trained on large-scale datasets. To overcome these hurdles, this paper introduces Parallel Explainer (PEXP), an innovative parallel tree-based interpretation framework designed for scalability and comprehensive understanding. PEXP initiates by generating a localized sample set around a target instance through data distribution-aware perturbations. It then computes similarity scores and employs a kernel function to weight these samples effectively. Leveraging concepts from Bagging and Boosting, PEXP efficiently constructs Parallel Ensemble Trees as its core interpretable model. This model provides feature importance-based explanations and aggregates insights across all samples to achieve a global understanding of the model's behavior on the entire dataset. Experimental results demonstrate PEXP's significant advantages over mainstream interpretable methods in both runtime efficiency and the quality of explanations, particularly crucial for Big Data analytics. Furthermore, a case study illustrates PEXP's application in enhancing the interpretability of video anomaly detection systems within smart cities, a domain characterized by large volumes of data and offers insights for improving Transformer-based architectures.**

***Index Terms*—Big Data, explanation, interpretable machine learning, systems management.**

## I. Introduction

IN RECENT years, deep learning has achieved remarkable success in Big Data analysis [1], [2], [3], [4], [5]. Unlike traditional machine learning methods [6], [7], [8], [9], [10], which typically convert unstructured data into structured data and attempt convex function convergence within an assumed data distribution, deep learning employs stacked hidden layers. This end-to-end process offers significant advantages over early expert systems [11], [12], [13], [14] and traditional machine learning, enabling better fitting to unknown data distributions and achieving superior performance [15], [16]. However, deep learning models suffer from the opaque decision-making problem [17], [18], [19], [20]. Specifically, as shown in Fig. 1, the three columns represent the original test images, the segments generated using Superpixel [21], and the top 10 regions identified by the model as critical for the "dog" category. The results reveal that the deep learning model is influenced to varying degrees by the background in all three test images. Notably, in the second image, the model identifies the snowy background as the most important region for a husky. Although the model exhibits excellent performance on evaluation metrics during training, its results deviate from human cognition, highlighting the importance of interpretable machine learning. In safety-sensitive domains, the absence of interpretability may lead to numerous unnecessary issues.

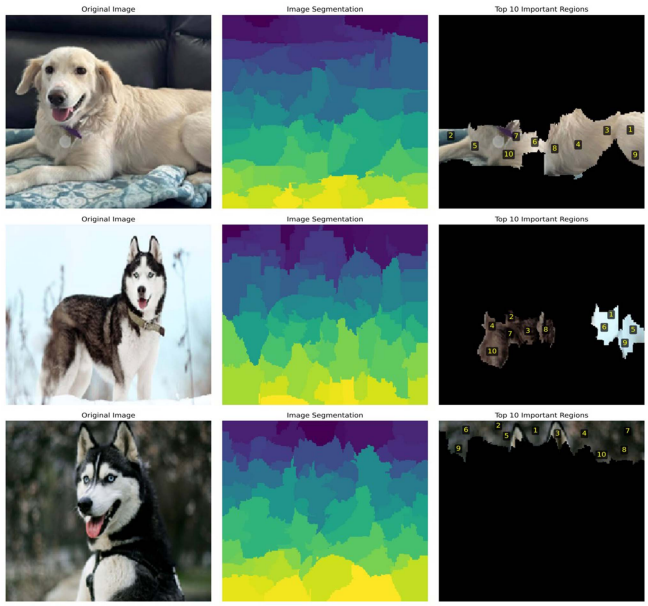



Fig. 1. The opaque decision-making problems in deeplearning models.

Received 14 May 2025; revised 26 September 2025; accepted 19 February 2026. Date of publication 27 February 2026; date of current version 9 July 2026. This work was supported by the National Science and Technology Council of Taiwan, Republic of China under Grant NSTC 112-2622-E-032-005. Recommended for acceptance by A. Bouguettaya. *(Corresponding author: Chih-Yung Chang.)*

Wen-Dong Jiang, Chih-Yung Chang, and Tzu-Chia Huang are with the Department of Computer Science and Information Engineering, Tamkang University, New Taipei 25137, Taiwan (e-mail: wendongjiang@ieee.org; cychang@mail.tku.edu.tw; 142870@o365.tku.edu.tw).

Yu-Ting Chin is with the Department of the Aritificial Intelligence, Tamkang University, New Taipei 25137, Taiwan (e-mail: chin.yuting.c@gmail.com).

Diptendu Sinha Roy is with the Department of Computer Science and Engineering, National Institute of Technology, Shillong 793003, India (e-mail: diptendu.sr@nitm.ac.in).

This article has supplementary downloadable material available at https://doi.org/10.1109/TBDATA.2026.3668673, provided by the authors.

Digital Object Identifier 10.1109/TBDATA.2026.3668673


Fig. 2. Explanation of the instability of results.

Interpretable machine learning [22], [23], [24] aims to elucidate opaque decision-making models by analyzing their decision-making processes, revealing their internal mechanisms and the basis for their output results, thereby enhancing model transparency and trustworthiness. Existing interpretable machine learning methods can be broadly categorized into three types: perturbation-based methods, gradient-based methods, and embedding-based methods. However, these methods generally suffer from the following issues: model limitations, instability of interpretation results, lack of global interpretability, and low computational efficiency.

The first issue is model limitations. Specifically, some methods [25], [26] are typically designed or modified for particular types of network, such as convolutional neural networks (CNNs) or Bayesian networks, to achieve transparency within those specific architectures. However, these approaches often struggle to generalize to arbitrary deep learning architectures.

The second issue is the instability of explanation results. As illustrated in Fig. 2, for different input images, the explanations generated by some interpretable algorithms can vary dramatically, exhibiting significant inconsistency. Consequently, this makes it difficult for users to trust these explanations, greatly diminishing their practical value in guiding model debugging, verifying the reliability of model behavior, and building user trust in AI systems. Misleading interpretations stemming from such instability can even lead to erroneous decisions.

The third issue is a lack of global interpretability. Specifically, many existing interpretability methods, such as LIME (Local Interpretable Model-agnostic Explanations) [27] or certain versions of SHAP (SHapley Additive exPlanations) [28], are primarily designed to provide "local" explanations, that is, to elucidate why a model makes a specific prediction for an individual input sample. While these local insights are highly valuable for understanding the decision logic in individual cases, they are often difficult to directly aggregate or effectively generalize to depict the model's overall behavior patterns, key decision boundaries, or generalizable feature importance across an entire dataset or a class of inputs. Consequently, users find it challenging to gain a macroscopic understanding of what the model has learned, and to systematically assess its overall bias, fairness, or consistency in performance across different data subsets.

The fourth issue is low computational efficiency. Specifically, many interpretability methods, particularly those reliant on extensive perturbation sampling or complex iterative optimization processes [27], [28], [29], [30], consume substantial computational resources and processing time when applied to high-dimensional data, large complex models, or when needing to explain numerous sample instances. For instance, generating an explanation for a single sample might necessitate hundreds or even thousands of repetitive evaluations of the original model. The high computational cost limits the practical use of these methods, especially in large-scale data scenarios or applications requiring real-time or near real-time responses. It also delays model iteration, debugging, and validation, ultimately reducing the overall efficiency of research and development.

To address the aforementioned limitations, this study introduces a novel parallel tree interpretation framework called Parallel Explainer (PEXP). The proposed PEXP's core operational workflow commences by applying data distribution-based perturbations to individual samples, thereby generating a local neighborhood of perturbed instances. Subsequently, the framework assesses the similarity between each perturbed instance within this neighborhood and the original sample, utilizing a kernel function to weight their relative importance. Following this, the proposed PEXP integrates Bagging and Boosting ensemble learning strategies to efficiently construct the Parallel Ensemble Tree structure, which serves as the core of the interpretation model. These ensemble trees then output feature importance scores, forming the basis for generating local explanations. Ultimately, by performing a weighted aggregation of the explanatory insights derived from all samples, the proposed PEXP can establish a global understanding of the entire dataset's behavioral patterns. Further details will be elaborated upon in Sections III and IV.

In practice, we have validated the efficacy of the proposed PEXP method through experiments on multiple benchmark datasets, spanning both image and video data. Furthermore, through case studies, this paper also demonstrates how the proposed PEXP method can assist in diagnosing models on large-scale datasets. In summary, the main contributions of this study can be outlined as the following three points:

1) The proposed PEXP introduces a perturbation sample generation technique based on the data distribution. Compared to methods of random perturbation generation [27]–[28] or fixedproportion generation [29]–[30], this technique offers higher stability.
2) The proposed PEXP introduces a method for constructing boosting and bagging trees based on parallel matrix aggregation. This method is more efficient compared to traditional methods for training surrogate models.

3) Extensive experiments demonstrate that PEXP outperforms current stateoftheart methods in stability, interpretability, and efficiency. Furthermore, through case studies, the potential of this method for explaining largescale video data in smart cities has also been explored.

The rest of this paper is organized as follows. Section II discusses and compares previous related work. Section III presents the proposed PEXP framework. Section IV describes experiments and performance evaluation. Section V discusses limitations and future directions, and Section VI concludes the paper.

## II. Related Work

This chapter introduces related work on interpretable machine learning, divided into three parts: perturbation-based methods, gradient backpropagation-based methods, embedding-based methods.

### A. Perturbation-Based Methods

The fundamental principle underlying these methods involved the systematic occlusion or modification of specific portions of individual test instances, followed by re-feeding the perturbed data into the model to obtain new predictions. The resultant changes in the model's predictions were then utilized to quantify the relative importance of individual features within the original test data.

For instance, Zeiler and Fergus [31] identified image regions deemed most salient by the model by systematically sliding a grey occluding patch across images and monitoring the corresponding shifts in model predictions. Ribeiro et al. [27] employed random occlusion of portions of images or text to generate new predictive outputs, and subsequently leveraged the coefficients of linear regression models, constructed based on these perturbed outputs, to elucidate the model's decision-making rationale. Petsiuk et al. [32] similarly employed a strategy of randomly masking input images to evaluate feature contributions. In the realm of game theory–based approaches, Ancona et al. [33] introduced a method that utilized Shapley values to approximate explanations for deep learning model predictions. Lundberg & Lee [28] proposed using the probabilistic derivative of Shapley values as a metric to quantify the discrepancy between perturbed samples and the original instance, thereby probing the model's internal mechanisms.

Furthermore, other strategies for generating perturbed samples were explored. Zafar and Khan [34] utilized clustering methods to generate perturbed samples for explanatory purposes. Liu et al. [35] introduced a framework combining interpretable dual trees with SHAP values for interpretability analysis. Lee et al. [29] developed EnEXP, which achieved interpretability through the construction of Random Forest and XGBOOST models. Jiang et al. [30] proposed RealExp, which enhanced interpretability by decomposing Shapley Values into individual and interaction importance components, and utilized tree-based structures for explanation.

*Gupte and Paparrizos* [50] carried out a deep empirical comparison of Shapley value approximation methods on tabular datasets, highlighting how different sampling and regression based estimators balance attribution accuracy against computational cost. *Oreti and Visani* [55] developed parallel implementations for a decisiontree explainability algorithm, achieving significant speedups on large datasets by distributing both tree induction and explanation phases across compute nodes. Earlier, *Sagi and Rokach* [56] proposed approximating XGBoost models with single, interpretable decision trees—transforming gradient-boosted forests into compact trees that retain most of the original predictive power. *Weinberg and Last* [57] presented an interpretable decisiontree induction method within a bigdata parallel framework, SySM, which selects a single representative tree based on syntactic similarity metrics to balance interpretability and efficiency at scale.

Although these methods were, in principle, applicable to any opaque decision-making model, they faced two primary challenges. Firstly, the generation of perturbed samples often relied on random or fixed-ratio sampling techniques, which could lead to a lack of stability in the interpretability outcomes. Secondly, when these methods depended on surrogate (or proxy) models, the construction process for such models was frequently inefficient. This inefficiency, in turn, constrained their capability to explain deep learning models when applied to large-scale datasets.

### B. Gradients-Based Methods

The primary objective of such techniques was to ascertain, via one or more iterations of backpropagation, which specific input elements exerted a substantial influence on the model's final output. Academic literature documents various such implementations: for instance, the work by Zhang et al. [36] utilized backpropagation to delineate the input pixels responsible for activating particular nodes or categories within the neural architecture, thereby illuminating the model's decision-making rationale. Selvaraju et al. [37] leveraged gradient information derived from convolutional layers to emphasize regions pivotal for the model's predictions, enabling a visual understanding of the model's attentional focus. Binder et al. [38] identified features bearing the most significant predictive power by retroactively distributing the output error proportionally to each input attribute. In a related vein, Shrikumar et al. [39] assigned "significance values" to input features by contrasting their individual contributions against a baseline input, thereby deconstructing the model's predictive behavior. Smilkov et al. [40] refined gradient visualizations by introducing random perturbations to input imagery and averaging the resulting gradients, aiming to bolster the robustness and perspicuity of these visual accounts. More recently, Bakchy et al. [41] developed a streamlined CNN architecture and employed the Grad-CAM methodology to facilitate model interpretability.

Wang et al. [51] proposed Feature Similarity Group Class Activation Mapping (FSG CAM), which clusters similar features before computing class activation maps to produce more coherent and human interpretable visual explanations. Mitra et al. [53] advanced ScoreCAM by incorporating a learnable gating mechanism ("ScoreCAM++") to weight neuron scores

dynamically, yielding sharper and more discriminative activation maps for CNNs.

Notwithstanding their provision of intuitive visual pathways for discerning model decision-making, these approaches were beset by four principal drawbacks. Initially, they encountered difficulties in furnishing lucid explanations for models characterized by a vast number of parameters. Furthermore, their applicability was predominantly confined to CNN and their derivatives, and largely restricted to image-based data. Additionally, such techniques were unable to deliver a holistic interpretation across an entire dataset. Lastly, the assessment of these explanatory techniques' efficacy was challenging due to the visual nature of their outputs; this stemmed less from the visual presentation itself and more from the fact that the rudimentary pixel agglomerations serving as explanations often lacked inherent semantic value within the specific application context.

### C. Embedding-Based Methods

Such approaches leveraged non-linear dimensionality contraction methodologies to augment the lucidity of data comprehension. Fundamentally, these methodologies projected samples onto adjacent regions within a lower-dimensional manifold, contingent upon their affinity within the high-dimensional distribution.

The research by Mnih et al. [42] served as an illustration, showcasing the application of t-SNE, a tool for dimensional compression. This instrument was designed to transform intricate high-dimensional information into scatter plots on a two-dimensional surface, where the arrangement of markers unveiled the inherent architecture of the source data. To impart supplementary insights, the hue of each data marker in the scatter plot was calibrated based on its attributes, whether acquired automatically or designated by human intervention. In contrast, *Engel and Mannor* [43] introduced a strategy employing embedded graph structures, which appraised the proximity between samples by computing their transitional likelihoods. More recently, Adrien et al. [44] developed DT-SNE, a scheme that refined t-SNE using decision tree models, intended to elevate the explanatory efficacy of conventional SNE approaches.

While these methodologies adeptly employed statistical instruments for explanatory ends, they encountered impediments when addressing voluminous datasets, primarily attributable to the intricacies of feature engineering.

Rawal et al. [52] tackled the vexing problem of assessing explanation fidelity in the absence of ground truth by introducing a suite of proxy metrics and humanintheloop evaluation procedures, demonstrating that existing benchmarks can misrank explanation methods. Most recently, Yang et al. [54] framed CNN interpretability as a coalitional game, enabling interactive exploration of feature coalitions and their marginal contributions, and thus offering users a principled way to probe model behavior.

Moreover, embedding techniques during the dimensionality contraction process frequently failed to comprehensively retain all attributes of the source data from its original n-dimensional space, thereby complicating subsequent analytical interpretation. Furthermore, as an unsupervised learning algorithm, t-SNE inherently could not guarantee that instances pertaining to identical classes would consistently maintain their agglomerative structure or analogous characteristics in the resultant visual representation.

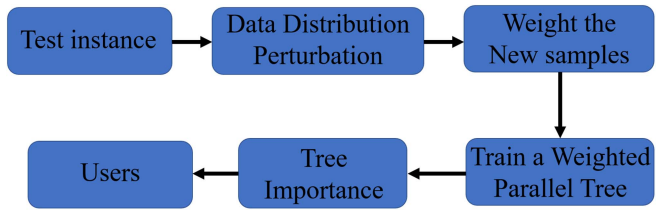



Fig. 3. The proposed PEXP workflow.

## III. The Proposed Parallel Explainer

This section presents the assumptions and problem statements of this study, divided into two parts: Detection and This section will provide a detailed exposition of our proposed Parallel Explainer (PEXP). Unlike existing perturbation-based methods [27], [28], [29], [30], the proposed PEXP not only significantly enhances the stability of explanation results but also optimizes computational efficiency. Compared to gradient-based methods [36], [37], [38], [39], [40], [41], the proposed PEXP is theoretically designed to generalize to any type of deep learning model, rather than being solely confined to CNN-based architectures. Furthermore, in contrast to embedding-based methods [42], [43], [44], the proposed PEXP can effectively perform interpretability analysis on large-scale datasets. Fig. 3 clearly illustrates the operational workflow of the proposed PEXP.

As shown in Fig. 3, the proposed PEXP method initially processes an input 'Test instance'. This instance then undergoes a 'Data Distribution Perturbation' phase, during which the system generates a series of new, perturbed samples around the original data point. Subsequently, these newly generated perturbed samples are 'Weighted' (Weigh the New samples), meaning they are assigned varying importance weights. These weighted perturbed samples are then used to train a core 'Weighted Parallel Tree' (Train a Weighted Parallel Tree) model. Local explanations, feature importance scores, or similar information extracted from this model are further integrated and refined through a 'Tree importance' step. Finally, the explanation results formed through to 'Users', The specific implementation steps and details are elaborated below.

### A. Data Distribution Perturbation

Firstly, the perturbation is based on the data distribution. Unlike previous works [27], [28] that use random perturbation generation or adversarial attacks [29], [30], the proposed method for generating perturbation samples is based on the original data distribution. The advantage of this is that the generated perturbation samples are more consistent with the intrinsic structure and statistical characteristics of the real data. This allows the model to exhibit better generalization ability and

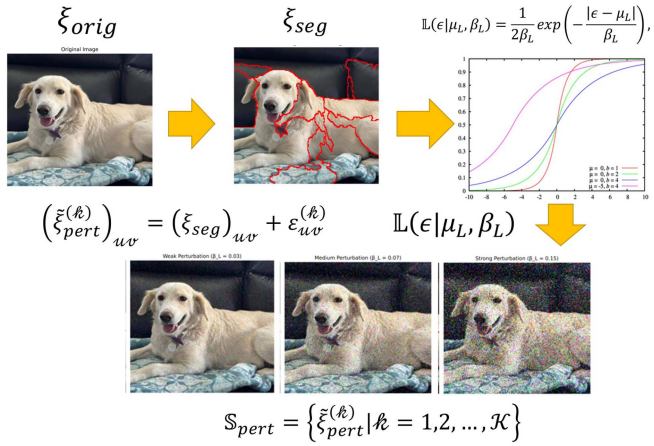



Fig. 4. The data distribution perturbation workflow.

stability when facing these perturbations, reducing the risk of a model's predictions fluctuating drastically due to encountering unknown perturbations that significantly differ from the training data distribution.

Let $\widetilde{\xi}_{\text{LNS}}$ denote the proposed perturbation based on the data distribution (DD), as illustrated in Fig. 4. Specifically, let $\xi_{\text{orig}}$ denote the high-dimensional image data to be analyzed. Let $\mathcal{W}_\Theta$ represent the Watershed image segmentation operator with a specific parameter set $\Theta$. Initially, the segmentation operator $\mathcal{W}_\Theta$ is applied to $\xi_{\text{orig}}$ to perform structural segmentation, yielding the segmented image $\xi_{\text{seg}}$. This transformation process can be expressed as $\xi_{\text{seg}} = \mathcal{W}_\Theta(\xi_{\text{orig}})$.

The segmented image $\xi_{\text{seg}}$ is defined over the pixel coordinate space $\Omega_{\text{seg}}$ as a pixel intensity function $\xi_{\text{seg}} : \Omega_{\text{seg}} \to \mathbb{R}^{\mathcal{C}}$, where $\mathcal{C}$ is the number of image channels. Its pixels are partitioned into a series of disjoint segmented regions $\mathcal{R}_j$, satisfying $\bigcup_{j=1}^{N_\mathcal{R}} \mathcal{R}_j = \Omega_{\text{seg}}$ and $\mathcal{R}_i \cap \mathcal{R}_j = \emptyset$ for $i \neq j$, where $N_\mathcal{R}$ is the total number of segmented regions.

Subsequently, to generate perturbed samples in a statistical sense, a Laplacian random process is employed to introduce perturbations to the intensity values of each pixel in the image $\xi_{\text{seg}}$. For this purpose, let $\mathbb{L}(\epsilon|\mu_L, \beta_L)$ denote the Laplace Probability Density Function, which is calculated in Exp. (1):

$$\mathbb{L}\left(\epsilon|\mu_L,\beta_L\right) = \frac{1}{2\beta_L}\exp\left(-\frac{|\epsilon-\mu_L|}{\beta_L}\right), \tag{1}$$

where $\mu_L \in \mathbb{R}$ is the location parameter of the distribution, and $\beta_L \in \mathbb{R}^+$ is the scale parameter, which directly controls the statistical dispersion of the injected noise.

To construct a sample set $\mathbb{S}_{\text{pert}} = \{\widetilde{\xi}^{(k)}_{\text{pert}} | k = 1,2,\dots,\mathcal{K}\}$ containing $\mathcal{K}$ independent and identically distributed perturbed images, a noise realization value $\varepsilon^{(k)}_{uv}$, independently sampled from the aforementioned Laplace distribution $\mathbb{L}(\bullet|\mu_L,\beta_L)$, is added to each pixel coordinate $(u,v)$ within the image domain $\Omega_{\text{seg}}$ of $\xi_{\text{seg}}$. Consequently, the pixel value $(\widetilde{\xi}^{(k)}_{\text{pert}})_{uv}$ of the $k$-th perturbed image $\widetilde{\xi}^{(k)}_{\text{pert}}$ at coordinate $(u,v)$ can be expressed in Exp. (2):

$$\left(\widetilde{\xi}^{(k)}_{\text{pert}}\right)_{uv} = (\xi_{\text{seg}})_{uv} + \varepsilon^{(k)}_{uv}, \tag{2}$$

where $\varepsilon^{(k)}$ is a random noise tensor possessing the same dimensions and structure as $\xi_{\text{seg}}$. Each of its elements $\varepsilon^{(k)}_{uv}$ is an independent random variable generated according to the probability density function $\mathbb{L}(\epsilon|\mu_L,\beta_L)$, denoted as $\varepsilon^{(k)}_{uv} \sim \mathbb{L}(\mu_L,\beta_L)$.

In summary, by applying a Laplacian noise injection mechanism to the pixel intensities of the segmented image $\xi_{\text{seg}}$, a series of statistically perturbed image samples $\mathbb{S}_{\text{pert}} = \{\widetilde{\xi}^{(k)}_{\text{pert}}\}_{k=1}^{\mathcal{K}}$ can be generated. Each instantiated sample $\widetilde{\xi}^{(k)}_{\text{pert}}$ in this set represents a specific random perturbation realization under this Laplacian noise model.

For comparison, we consider existing Random Masking (RM) [27], [28] and Proportional Masking (PM) [29], [30] methods. These methods typically operate directly on the original image $\xi_{\text{orig}}$. For RM/PM methods, each pixel $(\xi_{\text{orig}})_{uv}$ of the original image is replaced by a predefined constant $c$ with probability $\mathcal{P}_M$, and retains its original value with probability $1-\mathcal{P}_M$. Thus, a pixel value $(\hat{\xi}_M)_{uv}$ of the perturbed sample can be defined in Exp. (3):

$$\left(\hat{\xi}_M\right)_{uv} = \begin{cases} c, & \text{with probability } \mathcal{P}_M \\ (\xi_{\text{orig}})_{uv}, & \text{with probability } 1-\mathcal{P}_M. \end{cases} \tag{3}$$

A key difference between DD and RM/PM lies in the perturbation mechanism: The DD perturbs by adjusting the pixel values of the segmented image, whereas RM/PM perturbs by replacing pixel values of the original image, the latter potentially leading to information loss during replacement.

The Theorem 1 compares the key statistical properties of samples generated by the DD perturbs with those generated by RM/PM methods.

*Theorem 1:* Under the expected mean squared error, the DD method produces perturbations closer to the original data distribution, satisfying: $\mathbb{E}\|\xi_{\text{pert}} - \xi_{\text{seg}}\|_2^2 < \mathbb{E}\|\xi_M - \xi_{\text{orig}}\|_2^2$.

*Proof:* The DD method adds Laplacian noise to $\xi_{\text{seg}}$: $(\widetilde{\xi}_{\text{pert}})_{uv} = (\xi_{\text{seg}})_{uv} + \epsilon_{uv}$, where $\epsilon_{uv} \sim L(0,\beta_L)$ and $L(\epsilon|0,\beta_L) = \frac{1}{2\beta_L}\exp(-\frac{|\epsilon|}{\beta_L})$. The variance is $\text{Var}(\epsilon_{uv}) = 2\beta_L^2$, the expected error is $\mathbb{E}\|\xi_{\text{pert}} - \xi_{\text{seg}}\|_2^2 = \mathbb{E}[\sum_{u,v}\epsilon_{uv}^2] = |\Omega_{\text{seg}}| \cdot 2\beta_L^2$. The RM/PM method perturbs $\xi_{\text{orig}}$ as

$$\left(\widetilde{\xi}_M\right)_{uv} = \begin{cases} c, & \text{with probability } \mathcal{P}_M \\ (\xi_{\text{orig}})_{uv}, & \text{with probability } 1-\mathcal{P}_M \end{cases}$$

The expected error per pixel is $\mathbb{E}\|\xi_M - \xi_{\text{orig}}\|_2^2 = \mathcal{P}_M|\Omega_{\text{seg}}|\sigma^2$, where $\sigma^2 = [c - (\xi_{\text{orig}})_{uv}]^2$. For inequality comparison, $|\Omega_{\text{seg}}| \cdot 2\beta_L^2 < \mathcal{P}_M|\Omega_{\text{seg}}|\sigma^2$, simplifying, $2\beta_L^2 < \mathcal{P}_M\sigma^2$. Since $\beta_L$ is small and $\sigma^2$ is large, with $\mathcal{P}_M = 0.1$, the inequality holds. □

This indicates that the DD method is more stable in preserving the statistical properties of the data. After Data Distribution Perturbation, next part is weighting the new samples.

### B. Weighting the New Samples

The second step involves calculating the similarity between the perturbed samples and the original image, and then assigning weights based on this similarity for the subsequent training of the surrogate model. A two-stage method is used for weight assignment: initially, cosine similarity is calculated, which is then non-linearly adjusted using an exponential kernel function.

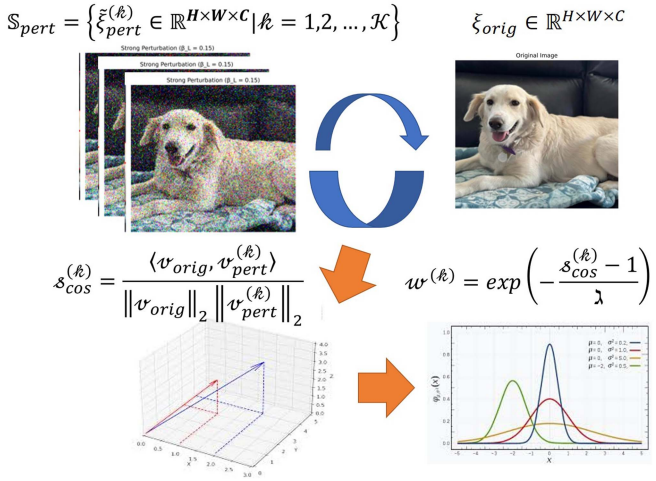



Fig. 5. The working flow of the weighting the new samples.

Specifically, as shown in Fig. 5, the perturbed sample set $\mathbb{S}_{\text{pert}} = \{\tilde{\xi}_{\text{pert}}^{(k)} \in \mathbb{R}^{H\times W\times C} | k = 1, 2, \dots, \mathcal{K}\}$, a critical procedure is the quantification of fidelity between each perturbed instance $\tilde{\xi}_{\text{pert}}^{(k)}$ and the original high-dimensional image $\xi_{\text{orig}} \in \mathbb{R}^{H\times W\times C}$. This fidelity measure underpins the derivation of a sample-specific weight, $w^{(k)} \in \mathbb{R}^{+}$, which is integral to the empirical risk minimization objective employed for training a surrogate model, denoted $\mathcal{F}_{\text{surrogate}}$. The weighting schema, $\mathcal{W} : (\mathbb{R}^{H\times W\times C}, \mathbb{R}^{H\times W\times C}) \to \mathbb{R}^{+}$, is operationalized via a two-stage methodology:

*Stage 1. Cosine Similarity Metric:* The initial stage quantifies the angular concordance between the latent vector representation of $\tilde{\xi}_{\text{pert}}^{(k)}$. Both multi-channel images are transformed into $\mathcal{D}$ dimensional vectors, where $\mathcal{D} = H \cdot W \cdot C$, through a vectorization mapping $\text{vec} : \mathbb{R}^{H\times W\times C} \to \mathbb{R}^{\mathcal{D}}$. Let $v_{\text{orig}} = \text{vec}(\xi_{\text{orig}})$ and $v_{\text{pert}}^{(k)} = \text{vec}(\tilde{\xi}_{\text{pert}}^{(k)})$. Let $s_{\cos}^{(k)}$ denote the cosine similarity, which resides in the interval $[-1, 1]$, and is mathematically defined as per Exp. (4):

$$s_{\cos}^{(k)} = \frac{\left\langle v_{\text{orig}}, v_{\text{pert}}^{(k)} \right\rangle}{\|v_{\text{orig}}\|_2 \left\|v_{\text{pert}}^{(k)}\right\|_2}, \tag{4}$$

where $\langle \cdot, \cdot \rangle$ signifies the canonical inner product in $\mathbb{R}^{\mathcal{D}}$, and $\| \cdot \|_2$ denotes the Euclidean $(L_2)$ norm. This metric isolates the orientation alignment, abstracting from magnitude discrepancies between the image vectors.

*Stage 2. Exponential Kernel Weighting Mechanism:* The set of scalar similarities $\{s_{\cos}^{(k)}\}_{k=1}^{\mathcal{K}}$ are subsequently transformed into a corresponding set of positive weights $\{w^{(k)}\}_{k=1}^{\mathcal{K}}$ by means of an exponential kernel function. This non-linear mapping serves to accentuate the contribution of high-fidelity perturbed samples within the surrogate model's learning process.

We first define a cosine distance $d_{\cos}^{(k)} = 1 - s_{\cos}^{(k)}$. Given $s_{\cos}^{(k)} \in [-1, 1]$, the corresponding $d_{\cos}^{(k)} \in [0, 2]$. For typical image data with non-negative pixel intensities, $s_{\cos}^{(k)}$ often lies in $[0, 1]$, rendering $d_{\cos}^{(k)} \in [0, 1]$.

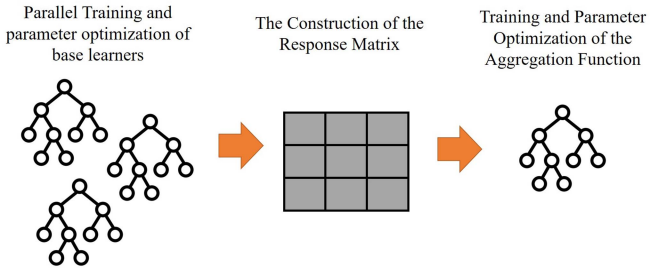



Fig. 6. The weighted parallel tree working flow.

The weight $w^{(k)}$ for the $k$-th sample $\tilde{\xi}_{\text{pert}}^{(k)}$ is then formulated as specified in Exp. (5):

$$w^{(k)} = \exp\left(-\frac{d_{\cos}^{(k)}}{\lambda}\right) = \exp\left(-\frac{1 - s_{\cos}^{(k)}}{\lambda}\right), \tag{5}$$

In Exp. (5), $\lambda \in \mathbb{R}^{+}$ represents the kernel bandwidth which governs the decay profile of the exponential weighting function. A diminutive $\lambda$ imposes a stringent locality, thereby assigning substantial weights almost exclusively to those samples $\tilde{\xi}_{\text{pert}}^{(k)}$ exhibiting $s_{\cos}^{(k)} \approx 1$. Conversely, a larger $\lambda$ effectuates a more lenient weighting distribution.

Theorem 2 aims to characterize the sensitivity of the kernel bandwidth to the weights, in order to demonstrate the robustness of the method. The specific performance analysis and proof are provided below.

*Theorem 2 (Sensitivity of the weight to the kernel bandwidth $\lambda$): For any fixed $d_{\cos}^{(k)} > 0$, the weight $w^{(k)}(\lambda)$ has first derivative $\frac{\partial w^{(k)}}{\partial \lambda} = w^{(k)}(\lambda)\frac{d_{\cos}^{(k)}}{\lambda^2} > 0$, and second derivative $\frac{\partial^{2\tilde{}} w^{(k)}}{\partial \lambda^2} = w^{(k)}(\lambda)\frac{d_{\cos}^{(k)}(d_{\cos}^{(k)} - 2\lambda)}{\lambda^4}$. The second derivative is negative when $\lambda < \frac{1}{2}d_{\cos}^{(k)}$ and positive when $\lambda > \frac{1}{2}d_{\cos}^{(k)}$.*

*Proof:* Differentiating the weight definition with respect to $\lambda$ yields $\frac{\partial w^{(k)}}{\partial \lambda} = \exp(-\frac{d}{\lambda}) \cdot (-\frac{d}{\lambda^2}) \cdot (-1) = w^{(k)}(\lambda)\frac{d}{\lambda^2}$, where $d = d_{\cos}^{(k)} > 0$. Hence the first derivative is always positive, indicating that increasing $\lambda$ raises all weights at a rate proportional to $d/\lambda^2$. A second differentiation gives the stated expression for $\partial^{2\tilde{}} w^{(k)}/\partial \lambda^2$, from which its concavity for $\lambda < d/2$ and convexity for $\lambda > d/2$ follow directly. □

After completing the weight calculation, the training of the parallel trees follows immediately.

### C. Train a Weighted Parallel Tree

As shown in Fig. 6, this process includes three stages: the first stage is parallel training and parameter optimization of base learners; the second stage is the construction of the response matrix; and the third stage is the training and parameter optimization of the aggregation function. Specifically, the training of the weighted parallel tree surrogate model, denoted as $F_{\text{surrogate}} = g_{\text{agg}} \circ \Phi_{\text{transform}}$, primarily involves the following three core stages. Let $\zeta_{\text{pert}} = \{\tilde{\xi}_{\text{pert}}^{(k)} \in X, y^{(k)} \in Y, w^{(k)} \in \mathbb{R}^{+}\}_{k=1}^{\mathcal{K}}$ denote the sample set whose objective is to learn the model parameters from the weighted perturbed data.

*Stage 1. Parallel Training and Parameter Optimization of Base Learners:* The goal is to train $M$ base learners $h_m : X \to Z_m$ (for $m = 1, \ldots, M$), each with parameters $\varphi_m \in \Theta_m$. Training begins with a data allocation strategy. Each $h_m$ can use the full weighted dataset $\zeta_{\text{pert}}$. Alternatively, to enhance efficiency and diversity, $D_w$ may be divided into $M$ subsets $D_w^{(m)} \subseteq D_w$, which may overlap or be disjoint. Another approach assigns each $h_m$ to a subspace $X^{(m)} \subseteq X$. This ensures parallelizable training. Parameter optimization determines $\varphi_m^*$ by solving the optimization problem in Exp. (6):

$$\begin{aligned}\varphi_m^* &= \arg\min_{\varphi_m \in \Theta_m} J_m\left(\varphi_m; D_w^{(m)}\right) \\ &\equiv \arg\min_{\varphi_m \in \Theta_m} \left[\mathbb{E}_{(\widetilde{\xi},y,w)\sim P_{D_w^{(m)}}}\left[w \cdot L_{\text{base}}\left(y, h_m(\widetilde{\xi};\varphi_m)\right)\right]\right. \\ &\left. +\lambda_b \cdot \Omega_{\text{reg}}\left(y, h_m(\varphi_m; D_w^{(m)})\right)\right]. \end{aligned} \tag{6}$$

Here, $\mathbb{E}_{(\widetilde{\xi},y,w)\sim P_{D_w^{(m)}}}$ represents the expectation under the empirical probability distribution $P_{D_w^{(m)}}$ defined by the weighted training subset $D_w^{(m)}$. The loss function $L_{\text{base}} : Y \times Z_m \to \mathbb{R}^+$ is chosen for the base learner, typically the mean squared error $(y - h_m(\widetilde{\xi};\varphi_m))^2$. The regularization functional $\Omega_{\text{reg}} : \Theta_m \times (X \times Y \times W)^{K_m} \to \mathbb{R}^+$ is applied to the parameters $\varphi_m$, with $\lambda_b \geq 0$ as the regularization strength coefficient. This optimization can be solved using standard algorithms tailored to the type of base learner $h_m$.

Upon completion of this stage, a set of trained base learner parameters $\{\varphi_m^*\}_{m=1}^M$ is obtained.

*Stage 2. Construction of the Response Matrix:* This stage leverages the set of trained base learners $\{h_m(\cdot, \varphi_m^*)\}_{m=1}^M$ to transform perturbed input samples from the input space $X$ into a joint response space $Z_L \subseteq \mathbb{R}^{D_L}$.

Initially, each perturbed sample $\widetilde{\xi}_{\text{pert}}^{(k)}$ $(k = 1, \ldots, K)$ in the weighted dataset $\zeta_{\text{pert}}$ is processed through all $M$ base learners to compute their responses, denoted as $h_{m,\text{output}}^{(k)} = h_m(\widetilde{\xi}_{\text{pert}}^{(k)}; \varphi_m^*)$.

Next, these responses are concatenated to form the $k$-th row vector $\ell_k$ of the response matrix $\ell \in \mathbb{R}^{K \times D_L}$, where $\ell_k = \bigoplus_{m=1}^M h_{m,\text{output}}^{(k)}$. Specifically, this concatenation operation results in $\ell_k = [(h_1(\widetilde{\xi}_{\text{pert}}^{(k)}; \varphi_1^*))^T, (h_2(\widetilde{\xi}_{\text{pert}}^{(k)}; \varphi_2^*))^T, \ldots, (h_M(\widetilde{\xi}_{\text{pert}}^{(k)}; \varphi_M^*))^T]^T$.

The total dimensionality of the response vector is $D_\ell = \sum_{m=1}^M \dim(Z_m)$. Upon completion, the response matrix $\ell$ is fully assembled, with each row representing a transformed sample in the feature space defined by the base learners.

*Stage 3: Training and Parameter Optimization of the Aggregation Function:* The objective of this stage is to train the second-layer aggregation function $g_{\text{agg}} : \mathbb{R}^{D_L} \to Y$, which is responsible for mapping the joint responses of the base learner ensemble to the final prediction output. The aggregation function $g_{\text{agg}}$ has its own set of parameters $\theta_{\text{agg}} \in \Theta_{\text{agg}}$.

The first step is construction of meta-dataset. Let $D_L = \{L_k, y^{(k)}, w^{(k)}\}_{k=1}^K$ denote the new training dataset, which is constructed using the response matrix $\ell$ generated in Stage 2 as input features, the target outputs $Y_{\text{pert}} = \{y^{(k)}\}_{k=1}^K$ from the original perturbed samples as target values, and the original sample weights $\Omega = \{w^{(k)}\}_{k=1}^K$.

The second step is parameter optimization. The optimal parameters $\theta_{\text{agg}}^*$ for the aggregation function are obtained by solving the weighted empirical risk minimization problem defined in Exp. (7):

$$\begin{aligned}\theta_{\text{agg}}^* &= \arg\min_{\theta_{\text{agg}} \in \Theta_{\text{agg}}} J_{\text{agg}}(\theta_{\text{agg}}; D_L) \\ &\equiv \arg\min_{\theta_{\text{agg}} \in \Theta_{\text{agg}}} \left[\mathbb{E}_{(z,y,w)\sim \hat{P}_{D_L}}\left[w \cdot L_{\text{agg}}\left(y, g_{\text{agg}}(z;\theta_{\text{agg}})\right)\right]\right. \\ &\left. +\lambda_g \cdot \Omega_{\text{agg}}(\theta_{\text{agg}})\right]. \end{aligned} \tag{7}$$

Here, $\mathbb{E}_{(z,y,w)\sim \hat{P}_{D_L}}$ denotes the expectation under the empirical probability distribution $\hat{P}_{D_L}$ defined by the meta-dataset $D_L$. $L_{\text{agg}} : Y \times Y \to \mathbb{R}^+$ is the loss function selected for the aggregation layer, $\Omega_{\text{agg}} : \Theta_{\text{agg}} \to \mathbb{R}^+$ is a regularization functional imposed on the parameters $\theta_{\text{agg}}$, and $\lambda_g \geq 0$ is the regularization strength coefficient. This optimization problem can be solved using standard algorithms suitable for the chosen type of aggregation function $g_{\text{agg}}$.

The trained aggregation function parameters $\theta_{\text{agg}}^*$ are obtained. The final Weighted Parallel Tree surrogate model $F_{\text{surrogate}}$ is jointly defined by the trained base learner parameters $\{\varphi_m^*\}_{m=1}^M$ and the aggregation function parameters $\theta_{\text{agg}}^*$. Its prediction for a new input $\xi$ is given by Exp. (8):

$$F_{\text{surrogate}}(\xi) = g_{\text{agg}}\left(\Phi_{\text{transform}}(\xi; \{\varphi_m^*\}_{m=1}^M); \theta_{\text{agg}}^*\right). \tag{8}$$

where $\Phi_{\text{transform}}(\xi; \{\varphi_m^*\}_{m=1}^M) = \bigoplus_{m=1}^M h_m(\xi; \varphi_m^*)$.

The proposed approach significantly enhances the efficiency of surrogate model construction, offering substantial computational advantages over traditional methods. Consider a dataset with $N$ samples and $d$ features, processed using identical parallel computing resources supporting at least $M$-way parallelism. Let $T_{PT}$, $T_{RF}$, and $T_{XGB}$ represent the training times for PTM, Random Forest, and XGBoost, respectively. Define $C_h$ as the computational cost of training one PTM base learner, $C_{rf}$ as the cost of training one Random Forest decision tree, and $C_x$ as the cost of training one XGBoost decision tree.

Additionally, let $C_\Sigma = C_{\text{matrix}} + C_{\text{aggregator}}$ denote PTM's total overhead for constructing the response matrix $C_{\text{matrix}}$ and training the aggregator $C_{\text{aggregator}}$. The efficiency of the proposed PTM depends on the following conditions:

1) For XGBoost, PTM achieves a lower training time if $C_h + C_\Sigma < M \cdot C_x$. This typically holds when $M > 1$, particularly when $C_h \approx C_x$ and the overhead $C_\Sigma < (M-1) \cdot C_x$.

2) For Random Forest, PTM achieves a lower training time if $C_h + C_\Sigma < C_{rf}$. This is satisfied when PTM's base learners are significantly less complex than Random Forest trees ($C_h \ll C_{rf}$) and $C_\Sigma$ remains modest.

Theorem 3 establishes that the Parallel Tree Model (PTM) outperforms Random Forest and XGBoost in training speed under specific conditions, surpassing the performance of prior works [29], [30].

*Theorem 3:* Under the specified conditions, PTM exhibits lower training time than XGBoost when $C_h + C_\Sigma < M \cdot C_x$,

and lower training time than Random Forest when $C_h + C_\Sigma < C_{rf}$.

*Proof:* For PTM, the $M$ base learners are trained in parallel, resulting in a training time dominated by $T_{PT} \approx C_h + C_\Sigma$. For XGBoost, the $M$ trees are constructed sequentially, with each tree's training dependent on prior trees. The total training time is $T_{XGB} = \sum_{i=1}^{M} C_{x_i} \approx M \cdot \bar{C}_x$, where $\bar{C}_x$ is the average training cost per XGBoost tree. By Condition 1, PTM has a lower training time if $C_h + C_\Sigma < M \cdot \bar{C}_x$. If $C_h \approx \bar{C}_x$, this reduces to $C_\Sigma < (M-1)\bar{C}_x$. For $M > 1$, PTM's parallel training yields an efficiency advantage when the overhead $C_\Sigma$ is less than the cost of training $M-1$ XGBoost trees, a condition that becomes increasingly achievable as $M$ increases.

For Random Forest, the $M$ trees are trained independently in parallel, so the training time is approximately the cost of a single decision tree $T_{RF} \approx C_{rf}$.

By Condition 2, PTM achieves a lower training time if $C_h + C_\Sigma < C_{rf}$. This holds when PTM's base learners are computationally lightweight, using shallower trees, fewer features, or simpler models, such that $(C_h \ll C_{rf})$ and the overhead $C_\Sigma$ is small. PTM's efficient design and parallelization thus provide a significant reduction in training time compared to Random Forest.

Compared to previous work [29], [30], Theorem 4 provides clear boundary conditions for achieving a computational advantage.

*Theorem 4:* PTM is faster than XGBoost if $C_h + C_\Sigma < MC_x \iff M > \frac{C_h + C_\Sigma}{C_x}$. PTM is faster than Random Forest if $C_h + C_\Sigma < C_{rf} \iff C_\Sigma < C_{rf} - C_h$.

*Proof:* In contrast to XGBoost, the $M$ base learners in PTM can be trained in full parallel. The total time is determined by the slowest base learner plus an additional aggregation overhead, given by:

$$T_{PTM} = \max_{1 \le m \le M}(C_h) + C_\Sigma$$

Conversely, the trees in XGBoost must be built serially, as each tree depends on the residuals of the preceding one. Its total time is therefore $T_{XGB} = \sum_{i=1}^{M} C_{x_i} \approx MC_x$. Thus, for PTM to have a computational advantage $T_{PTM} < T_{XGB}$, the condition $C_h + C_\Sigma < MC_x$ must be met.

Compared to Random Forest, although its $M$ trees can also be trained in parallel, the total time is similarly bottlenecked by the slowest single tree, yielding $T_{RF} \approx C_{rf}$. For PTM to be faster $T_{PTM} < T_{RF}$, the condition is $C_h + C_\Sigma < C_{rf}$. □

Therefore, under the stated conditions, PTM demonstrates superior computational efficiency, completing the proof. After training the Weighted Parallel Trees, the next step is the tree importance analysis.

### D. Tree Importance

This process helps in understanding which original input features contribute most significantly to the model's predictions, offering valuable insights for interpretability. This guide outlines a simplified approach using Gini importance, assuming that both the base learners and the aggregator in your Weighted Parallel Trees model are tree-based. The calculation of feature importance proceeds in the following steps:

*Step 1: Calculate Feature Importance within Each Base Learner $h_m$:* For every base learner $h_m$ in the model: Determine the Gini importance of each original feature $X_j$ within that specific tree $h_m$. This value, $\text{Gini}(X_j|h_m)$, is typically a standard output from decision tree implementations. It quantifies the total reduction in Gini impurity brought about by splits on feature $X_j$ within tree $h_m$.

Normalize these feature importances for each base learner $h_m$ so that they sum to 1, assuming at least one feature is used by $h_m$, as shown in Exp. (9):

$$I_{\text{norm}}(X_j|h_m) = \frac{\text{Gini}(X_j|h_m)}{\sum_{k=1}^{d} \text{Gini}(X_k|h_m)}, \tag{9}$$

where $d$ is the total number of original input features. If a base learner $h_m$ does not use any features, all $I_{\text{norm}}(X_j|h_m)$ will be zero for that learner.

*Step 2. Calculate Importance of Base Learner Outputs for the Aggregator $(g_{agg})$:* The aggregator $g_{\text{agg}}$ takes the outputs $\{z_1, z_2, \ldots, z_M\}$ from the $M$ base learners as its input features.

Calculate the Gini importance of each input $z_m$ for the aggregator tree $g_{\text{agg}}$. This is denoted as $\text{Gini}(z_m|g_{\text{agg}})$ and can be obtained from $g_{\text{agg}}$'s own feature importance scores if it's a tree-based model. Normalize these importances so they sum to 1, as shown in Exp. (10):

$$I_{\text{norm}}(z_m|g_{\text{agg}}) = \frac{\text{Gini}(z_m|g_{\text{agg}})}{\sum_{k=1}^{M} \text{Gini}(z_k|g_{\text{agg}})}, \tag{10}$$

*Step 3. Combine Importances for Global Feature Importance:* The global Gini-based importance of an original feature $X_j$, $\text{FI}(X_j)$, is derived by combining the normalized importances from the previous two steps, as shown in Exp. (11):

$$\text{FI}(X_j) = \sum_{m=1}^{M} \left( I_{\text{norm}}(z_m|g_{\text{agg}}) \cdot I_{\text{norm}}(X_j|h_m) \right). \tag{11}$$

Intuitively, this formula suggests that the overall importance of an original feature $X_j$ is a weighted sum. It considers how important $X_j$ is to each base learner $h_m$ ($I_{\text{norm}}(X_j|h_m)$), and then weights this by how important that base learner $h_m$'s output is to the final decision made by the aggregator $g_{\text{agg}}$ ($I_{\text{norm}}(z_m|g_{\text{agg}})$).

The above outlines the complete details of the proposed PEXP model. In practical applications, users can utilize visualization techniques to identify the Top-K features contributing most significantly to model predictions. We have also provided the pseudocode algorithm and relevant explanations in the Appendix. Please refer to the Appendix version. The experimental section follows.

## IV. Experimental

This chapter will introduce the performance of the proposed PEXP. The experimental section is divided into five parts: dataset, hyperparameter settings, evaluation metrics introduction, comparison with state-of-the-art methods, ablation studies. The specific presentations of the case studies, along with

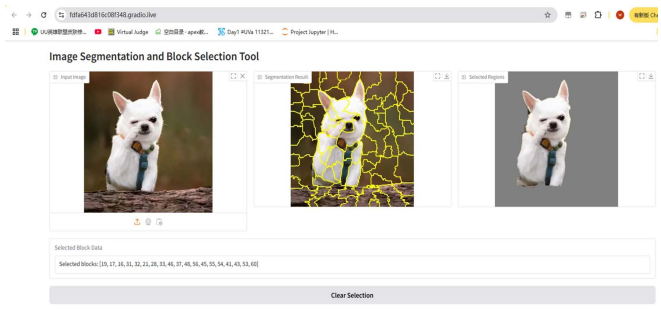



Fig. 7. The expert annotation process.

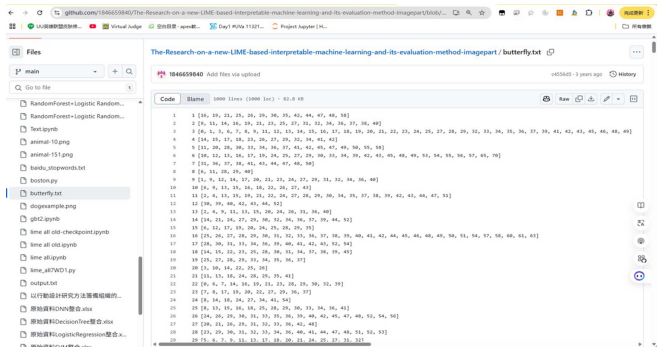


Fig. 8. The annotators final labels.

additional experiments, can be found in the supplementary file provided in the Appendix.

### A. Dataset

For benchmark performance evaluation, the Animal-10 and Animal-151 datasets were adopted. The Animal-10 dataset contains approximately 28,000 medium-quality animal images, which are categorized into 10 classes: dog, cat, horse, spider, butterfly, chicken, sheep, cow, squirrel, and elephant. The Animal-151 dataset is characterized by a small number of images per class: each class has a maximum of 60 images, while some classes have only 30. To save storage space, all images were resized to 224 × 224 pixels. The selection of these datasets and the image preprocessing methods are consistent with previous works [27]–[31].

In case-based experiments, the XD-Violence dataset [45] was selected for the video anomaly detection task. This dataset has a total duration of 217 hours and includes 4754 untrimmed videos with audio signals and weak labels.

In addition, a manual evaluation benchmark was established for the interpretability experiments. Specifically, 1,000 images were selected from each of the Animal-10 and Animal-151 datasets, yielding a total of 2,000 images for expert annotation. A labeling platform was built using the Gradio front-end application, and public access links were distributed to three researchers. As shown in Fig. 7, the right panel displays the images to be annotated, while the central panel updates segmentation results in real time. Experts selected regions deemed to contain dogs using the mouse, then ranked those regions in order of importance.

To ensure objectivity, experts were not shown any model outputs during the annotation process. All annotators followed a standardized protocol that defined region granularity, semantic boundaries, and importance criteria, which was provided before annotation. To further ensure rigor, the annotations from all three experts were cross-compared, and the majority-consensus combinations illustrated in Fig. 8 were adopted as the final labels.

### B. Parameter Settings and Model Selection

This section introduces the hyperparameter settings and model selection. First, for the perturbation samples $S_{\text{pert}} = \{\widetilde{\xi}_{\text{pert}}^{(k)} \in \mathbb{R}^{H\times W\times C} | k = 1, 2, \ldots, K\}$, the number $K$ is set to 1000. In Exp. (1), $\mu_L$ and $\beta_L$ were uniformly set to 0.2, and in Exp. (4), $\lambda$ was set to 0.25. The number of parallel trees $h_m$, $m$ was set to 50.

Regarding model selection, for the benchmark datasets, we explored four models using ImageNet pre-trained weights: MobileNet [46], ResNet [47], DenseNet [48], and Vision Transformer (ViT) [49].

During training of both the base learners and the aggregator, we employed the following configurations: for optimizer settings, we used the Adam optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$) with an initial learning rate of $1 \times 10^{-3}$, a CosineAnnealingLR schedule ($T_{\max} = 50$), and a weight decay of $1 \times 10^{-4}$ ; for training configurations, we built both the base learners and the aggregator as small multilayer perceptrons (MLPs); for random seed and reproducibility, we fixed the seed at 759 and enabled deterministic mode in PyTorch (`torch.backends.cudnn.deterministic=True`, `torch.backends.cudnn.benchmark=False`); and for hardware specifications, we ran all experiments on Ubuntu 20.04 LTS using two NVIDIA Tesla V100 GPUs (16 GB each), an Intel Xeon Gold 6248 CPU (20 cores, 2.5 GHz), and 128 GB of DDR4 RAM, with Python 3.8, PyTorch 1.10.0, CUDA 11.3, and cuDNN 8.2.

### C. Evaluation Metrics

We analyze the model performance of the proposed PEXP from five aspects: stability, fidelity, coarse-grained human expert matching, fine-grained human matching and influence time.

*Stability:* To quantitatively assess the robustness of interpretable models, particularly considering the congruence between their attributed regions (explained regions) and the original salient parts of an image, and referencing the methodologies in existing literature [27]–[30], this study introduces the Jaccard coefficient as one of the core evaluation criteria. The experimental design involves subjecting the same experiment to five-fold independent replications under conditions of slight variations.

For any $k$-th replication $k = 1, 2, \ldots, 5$, the explanation method outputs a set composed of identified key features. This set (whose elements may be salient regions in an image or core feature indices in structured data) is denoted as $A_k$. For any two distinct replication instances $k$ and $\ell$ (where $k \neq \ell$), the degree of overlap between their corresponding feature sets $A_k$ and $A_\ell$ is precisely quantified using the Jaccard coefficient as defined

in Exp. (12):

$$J(A_k, A_\ell) = \frac{|A_k \cap A_\ell|}{|A_k \cup A_\ell|}, \tag{12}$$

where $|A_k \cap A_\ell|$ represents the cardinality of the intersection of feature sets $A_k$ and $A_\ell$, the number of features common to both sets; correspondingly, $|A_k \cup A_\ell|$ is the cardinality of their union, representing the total count of unique features across both sets. A higher value of the Jaccard coefficient indicates a higher level of intrinsic consistency between the attribution results generated from the two replications.

Finally, an overall stability metric is constructed by aggregating the mean Jaccard coefficient from all possible pairs (amounting to 10 pairs) across all five replications. If an explanation method consistently maps to the same or highly similar feature subsets in all replications, this average Jaccard coefficient will tend towards a higher value, thereby characterizing the method as possessing superior explanatory robustness.

*Fidelity:* To provide a rigorous quantitative characterization of the fidelity of the proposed methods—this being a core attribute concerning their precise reproduction of a benchmark model's behavior—and to ensure this assessment deeply aligns with the evaluation paradigms established in existing literature [27]–[30], this study employs the Coefficient of Determination $R^2$, also frequently regarded as a goodness-of-fit index, as the key quantitative metric for gauging this fidelity. The mathematical construct of this metric is specifically elucidated as follows in Exp. (13):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \tag{13}$$

where $y_i$ serves to characterize the true response, or alternatively, the observed ground truth $F(x_i)$ produced by the benchmark model $F(\cdot)$ for the $i$-th specific input instance $x_i$.

Correspondingly, $\hat{y}_i$ represents the predicted response or approximate output $g(x_i)$ generated by an explanation mechanism $g(\cdot)$, which aims to simulate the behavior of the benchmark model $F(\cdot)$ or provide local intelligible approximations—for the same input instance $x_i$.

Furthermore, $\bar{y}$ is defined as the sample mean of the true response sequence $\{y_j\}_{j=1}^{n}$, where $j$ is the sample index.

*Coarse-Grained Human Expert Matching:* To evaluate explanation methods for image classification that are based on superpixels, let's first define some basic concepts. Suppose $M$ is the image classification model we want to explain. For the input image $\xi \in \mathbb{R}^{H\times W\times C}$, we use image segmentation algorithm $Q$ to divide the image $\xi$ into $n$ small regions called superpixels, denoted as $\{s_1, s_2, \ldots, s_n\}$. These superpixel regions do not overlap and together they form the entire image, where $s_i \cap s_j = \emptyset$ for $i \neq j$, and $\bigcup_{i=1}^{n} s_i = \xi$.

Next, we ask domain experts to identify the $m$ superpixels they believe contribute most significantly to the model's classification result. We denote this set of expert-selected superpixels as $U = \{u_1, u_2, \ldots, u_m\}$, and we assume the experts also provide an importance ranking for these $m$ superpixels. At the same time, the interpretable algorithm we are evaluating calculates an importance score $F(s_i)$ for every superpixel $s_i$. Based on these scores, we select the top-$m$ superpixels that the algorithm considers most important, denoting this set as $K = \{k_1, k_2, \ldots, k_m\}$, also sorted from most to least important.

TABLE I
SUMMARY OF NOTATION IN SECTION III

| Symbol | Definition |
|---|---|
| $\tilde{\xi}$ | Perturbed instance |
| $\xi_{\text{seg}}$ | Segmented image |
| $\xi_{\text{orig}}$ | Original instance |
| $\xi_{\text{pert}}$ | Perturbation set |

To quantify the consistency between the model's explanations and expert judgments, we introduce two evaluation metrics, the ideas for which are drawn from the work of [29], [30].

First, we find the superpixels that are considered important by both the experts and the XAI algorithm. We denote this set of commonly selected superpixels as $H_{\text{common}} = U \cap K$. Let $H_{\text{score}}$ denote the coarse-grained human expert matching, which is calculated in Exp. (14):

$$H_{\text{score}} = \frac{|H_{\text{common}}|}{m}, \tag{14}$$

where $|H_{\text{common}}|$ is the number of commonly identified superpixels, and $m$ is the total number of superpixels marked by experts. A higher $H_{\text{score}}$ indicates better accuracy by the algorithm in identifying these superpixels.

*Fine-Grained Human Matching:* Let $M_{\text{score}}$ denote the fine-grained human matching score. This score is used to assess whether the importance ranking given by the XAI algorithm is consistent with the experts' ranking for the commonly identified important features. To do this, we need to define 'concordant pairs' and 'discordant pairs'. For any distinct pair of superpixels $(S_a, S_b)$ from $H_{\text{common}}$. Let $N_C$ and $N_D$ denote the count of concordant pairs and discordant pairs, respectively.

The total number of pairs in $H_{\text{common}}$ is $\frac{|H_{\text{common}}|(|H_{\text{common}}|-1)}{2}$. The $M_{\text{score}}$ uses a calculation method similar to Kendall's $\tau$ coefficient as shown in Exp. (15):

$$\begin{aligned} M_{\text{score}} &= \frac{N_C - N_D}{|H_{\text{common}}|(|H_{\text{common}}| - 1)/2} \\ &= \frac{2(N_C - N_D)}{|H_{\text{common}}|(|H_{\text{common}}| - 1)} \end{aligned} \tag{15}$$

*Influence Time:* Inference time refers to the total duration required for the proposed explainable method to generate explanations for the entire dataset. To ensure fairness of comparison, all relevant time evaluations are executed on the same hardware device.

### D. Compare With the State of the Arts Method

Tables I and II aim to compare the performance of the proposed PEXP method with that of previous research efforts. This evaluation spans multiple backbone neural network architectures, including MobileNet, ResNet, DenseNet, Vision Transformer (ViT), and Vision GNN (VIG), and utilizes two

TABLE II
COMPARISON WITH STATE-OF-THE-ART METHODS ON ANIMAL-10 DATASET

| Model | Zeiler & Fergus [31] (Baseline, 2014) | | | | Ribeiro et al. [27] (LIME, 2016) | | | |
|---|---|---|---|---|---|---|---|---|
| | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ |
| MobileNet | 0.46 | 0.53 | 0.54 | 0.52 | 0.73 | 0.76 | 0.68 | 0.65 |
| ResNet | 0.46 | 0.51 | 0.46 | 0.50 | 0.73 | 0.77 | 0.71 | 0.60 |
| DenseNet | 0.46 | 0.52 | 0.50 | 0.51 | 0.73 | 0.76 | 0.76 | 0.58 |
| ViT | 0.46 | 0.50 | 0.50 | 0.50 | 0.73 | 0.83 | 0.81 | 0.52 |
| ViG | 0.46 | 0.48 | 0.46 | 0.48 | 0.73 | 0.78 | 0.79 | 0.54 |
| | Lundberg & Lee [28] (KernelSHAP, 2017) | | | | Liu et al. [35] (HIDTWM, 2024) | | | |
| | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ |
| MobileNet | 0.71 | 0.77 | 0.69 | 0.66 | 0.75 | 0.78 | 0.70 | 0.67 |
| ResNet | 0.71 | 0.78 | 0.72 | 0.61 | 0.75 | 0.79 | 0.73 | 0.62 |
| DenseNet | 0.71 | 0.77 | 0.76 | 0.59 | 0.75 | 0.78 | 0.78 | 0.60 |
| ViT | 0.71 | 0.84 | 0.82 | 0.53 | 0.75 | 0.85 | 0.83 | 0.54 |
| ViG | 0.71 | 0.77 | 0.70 | 0.55 | 0.75 | 0.81 | 0.72 | 0.58 |
| | Jiang et al. [30] (RealEXP, 2025) | | | | The proposed PEXP | | | |
| | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ |
| MobileNet | 0.83 | 0.85 | 0.78 | 0.71 | **0.88** | **0.87** | **0.82** | **0.73** |
| ResNet | 0.83 | 0.86 | 0.81 | 0.66 | **0.88** | **0.88** | **0.84** | **0.77** |
| DenseNet | 0.83 | 0.85 | 0.86 | 0.64 | **0.88** | **0.91** | **0.90** | **0.71** |
| ViT | 0.83 | 0.94 | 0.92 | 0.58 | **0.88** | **0.96** | **0.94** | **0.61** |
| ViG | 0.83 | 0.81 | 0.83 | 0.62 | **0.88** | **0.86** | **0.91** | **0.64** |

TABLE III
COMPARISON WITH STATE-OF-THE-ART METHODS ON ANIMAL-151 DATASET

| Model | Zeiler & Fergus [31] (Baseline, 2014) | | | | Ribeiro et al. [27] (LIME, 2016) | | | |
|---|---|---|---|---|---|---|---|---|
| | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ |
| MobileNet | 0.46 | 0.43 | 0.42 | 0.41 | 0.73 | 0.64 | 0.53 | 0.55 |
| ResNet | 0.46 | 0.44 | 0.30 | 0.41 | 0.73 | 0.68 | 0.53 | 0.53 |
| DenseNet | 0.46 | 0.41 | 0.36 | 0.45 | 0.73 | 0.62 | 0.71 | 0.36 |
| ViT | 0.46 | 0.44 | 0.30 | 0.45 | 0.73 | 0.67 | 0.61 | 0.23 |
| ViG | 0.46 | 0.42 | 0.33 | 0.42 | 0.73 | 0.65 | 0.65 | 0.42 |
| | Lundberg & Lee [28] (KernelSHAP, 2017) | | | | Liu et al. [35] (HIDTWM, 2024) | | | |
| | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ |
| MobileNet | 0.71 | 0.63 | 0.51 | 0.54 | 0.75 | 0.66 | 0.55 | 0.58 |
| ResNet | 0.71 | 0.69 | 0.52 | 0.54 | 0.75 | 0.71 | 0.55 | 0.53 |
| DenseNet | 0.71 | 0.62 | 0.64 | 0.37 | 0.75 | 0.58 | 0.73 | 0.38 |
| ViT | 0.71 | 0.68 | 0.63 | 0.42 | 0.75 | 0.69 | 0.64 | 0.43 |
| ViG | 0.71 | 0.66 | 0.62 | 0.44 | 0.75 | 0.63 | 0.59 | 0.34 |
| | Jiang et al. [30] (RealEXP, 2025) | | | | The proposed PEXP | | | |
| | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ | $J\uparrow$ | $R^2\uparrow$ | $H_{\text{score}}\uparrow$ | $M_{\text{score}}\uparrow$ |
| MobileNet | 0.83 | 0.73 | 0.56 | 0.51 | **0.88** | **0.76** | **0.59** | **0.52** |
| ResNet | 0.83 | 0.66 | 0.65 | 0.51 | **0.88** | **0.69** | **0.66** | **0.53** |
| DenseNet | 0.83 | 0.70 | 0.67 | 0.52 | **0.88** | **0.72** | **0.67** | **0.54** |
| ViT | 0.83 | 0.84 | 0.82 | 0.48 | **0.88** | **0.86** | **0.88** | **0.52** |
| ViG | 0.83 | 0.82 | 0.77 | 0.51 | **0.88** | **0.83** | **0.84** | **0.51** |

image datasets: Animal-10 and the more challenging Animal-151. Performance is measured using four core metrics: Jaccard index ($J$), Coefficient of Determination ($R^2$), Hit Rate ($H_{\text{score}}$), and Ranking Consistency ($M_{\text{score}}$), with higher values being desirable for all.

On the relatively simpler Animal-10 dataset, the PEXP method comprehensively surpassed all compared methods, including baseline approaches [31], LIME [27], KernelSHAP [28], HIDTWM [35], and the advanced RealEXP [30], achieving top scores across all metrics and backbone models. Specifically, the proposed PEXP achieved a Jaccard index of 0.88 across all models, its $R^2$ score reached as high as 0.96 on ViT, with VIG also showing strong performance at 0.81. The $H_{\text{score}}$ was outstanding across all architectures (e.g., 0.94 on ViT and 0.83 on VIG). The $M_{\text{score}}$ was also leading, with a notable improvement on ResNet (0.77 compared to RealEXP's 0.66), and VIG achieving 0.64, although ViT's $M_{\text{score}}$ (0.61) was slightly less prominent than its advantages in other metrics.

When faced with the more complex Animal-151 dataset, the performance metrics of all methods generally declined, further

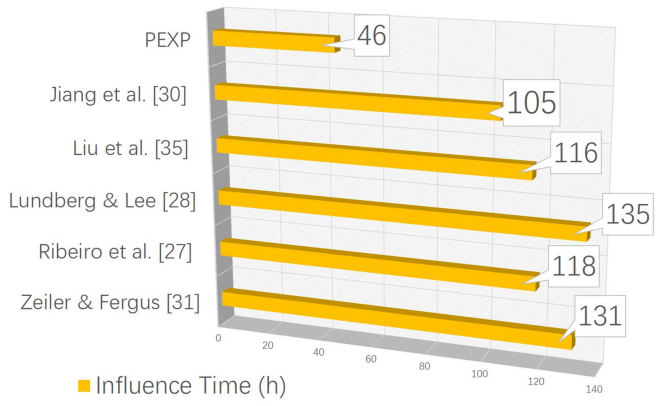



Fig. 9. Compare with influence time in Animal-10 dataset.

highlighting the difficulty of generating high-quality explanations on this dataset. Even so, the proposed PEXP method maintained its leading position. Its $J$ remained stable at 0.88 across all models, and while its $R^2$ scores decreased, they remained the best (e.g., 0.86 on both ViT and VIG). The $H_{\text{score}}$ also maintained strong performance (e.g., 0.88 on ViT and 0.84 on VIG). The decline in $M_{\text{score}}$ was more widespread and significant (the proposed PEXP generally scored between 0.52–0.54 on this dataset, with VIG achieving 0.54), reflecting that feature ranking consistency is a major challenge in the current XAI field.

Overall, the proposed PEXP method not only demonstrated optimal performance on both datasets, continuously surpassing previous SOTA methods (like RealEXP [30]), but also exhibited good robustness to dataset complexity, high fidelity to model behavior (high $R^2$ values), the ability to accurately identify features considered important by experts (high $H_{\text{score}}$), and improvements in feature ranking consistency (relatively high $M_{\text{score}}$). The proposed PEXP method can efficiently adapt to different types of network architectures, from various CNNs to transformer-based models (ViT) and graph-based models (VIG), and its effectiveness in terms of fidelity and feature identification is particularly prominent when combined with ViT and VIG architectures.

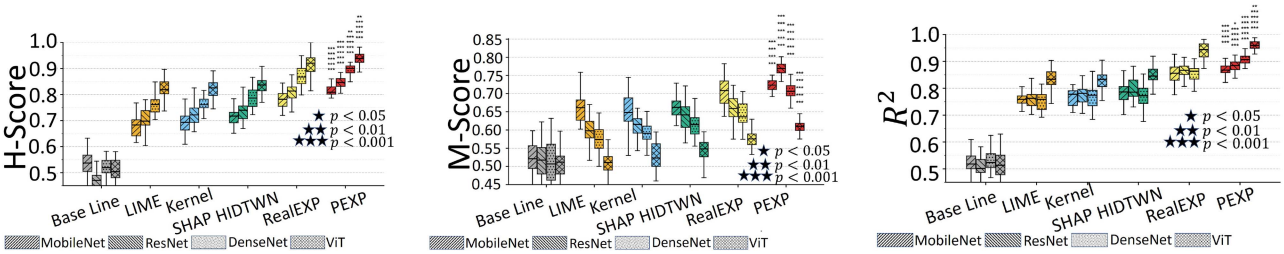
The experimental results have validated the correctness of Theorem 1. Moreover, regarding fidelity and human expert evaluations, the results indicate that the proposed Parallel Tree method, when constructing a proxy model to explain the original model, is more effective than methods employing traditional ensemble tree models, ridge regression, or linear regression as proxy models.

Fig. 9 clearly illustrates a comparison of the inference time required by different XAI methods to generate explanations on the Animal-10 dataset. Most strikingly, the proposed PEXP method demonstrates significant superiority in computational efficiency, completing the explanation task for the entire dataset in only 46 hours. In comparison, several other mainstream methods, including Jiang et al.'s [30] RealEXP method (105 hours), Liu et al.'s [35] HIDTWM method (116 hours), Ribeiro et al.'s [27] LIME method (118 hours), Zeiler & Fergus's [31] baseline method (131 hours), and Lundberg & Lee's [28] KernelSHAP (135 hours), all require considerably longer times, with their processing durations generally ranging from 105 to 135 hours. Specifically, the proposed PEXP's inference time is substantially lower than all other compared methods; for instance, its time consumption is less than fifty percent of that of the second-fastest method by Jiang et al. and only approximately one-third of that required by the slowest method, KernelSHAP [28] by Lundberg & Lee. This result strongly demonstrates that the proposed PEXP method not only excels in explanation effectiveness but also possesses outstanding computational efficiency. This high efficiency further corroborates the conclusion of Theorem 3, which states that employing parallel processing can effectively reduce inference time.

Fig. 10 aims to compare the effectiveness of the proposed explainable method through statistical experiments. A consolidated analysis of these statistical experimental results for $H_{\text{score}}$ (subplot a), $M_{\text{score}}$ (subplot b), and $R^2$ (subplot c) reveals that the proposed PEXP method consistently and significantly surpasses all compared methods, including Baseline [31], LIME [27], KernelSHAP [28], HIDTWM [35], and RealEXP [30], across all three core evaluation metrics and for various backbone network models (MobileNet, ResNet, DenseNet and ViT). Furthermore, this performance advantage is highly statistically significant ($P < 0.001$). Overall, the performance on these metrics generally improves progressively with the evolution of explanation methods from earlier stages to the proposed PEXP.

Specifically, for the proposed PEXP method, it achieves exceptionally high and stable performance levels in $H_{\text{score}}$ (medians between 0.8 and 0.95) and $R^2$ score (medians between 0.85 and 0.97), with particularly outstanding results when combined with advanced backbone networks like ViT and DenseNet; its compact box plots also indicate consistency. In terms of $M_{\text{score}}$, the proposed PEXP also demonstrates the best performance, especially when applied to CNN models (such as MobileNet, ResNet, DenseNet), where medians can reach a high level of 0.7 to 0.78. However, it is noteworthy that the proposed PEXP's $M_{\text{score}}$ when combined with the ViT model (median 0.6 to 0.62), while still leading other methods' performance on ViT, is slightly lower compared to the proposed PEXP's $M_{\text{score}}$ on CNN models and its own excellent $H_{\text{score}}$ and $R^2$ scores. This might reflect specific challenges in the ranking consistency of feature importance for ViT model explanations.

In summary, the proposed PEXP method achieves comprehensive and statistically significant improvements in key dimensions of explainability, including feature identification accuracy, model behavior fidelity, and feature ranking consistency. Furthermore, this once again substantiates the assertion of Theorem 1, that employing a strategy of generating perturbed samples based on data distribution helps to improve the stability of the explanation method's performance. Fig. 10 aims to compare the effectiveness of the proposed explainable method through statistical experiments. A consolidated analysis of these statistical experimental results for $H_{\text{score}}$ (subplot a), $M_{\text{score}}$ (subplot b), and $R^2$ (subplot c) reveals that the proposed PEXP method consistently and significantly surpasses all compared methods, including Baseline [31], LIME [27], KernelSHAP [28], HIDTWM [35], and RealEXP [30], across all three core evaluation metrics and for various backbone network models (MobileNet, ResNet,




(a) $H_{\text{score}}$ (b) $M_{\text{score}}$ (c) $R^2$

Fig. 10. Statistical experiments on Animal-10 dataset

DenseNet and ViT). Furthermore, this performance advantage is highly statistically significant ($P < 0.001$). Overall, the performance on these metrics generally improves progressively with the evolution of explanation methods from earlier stages to the proposed PEXP.

Specifically, for the proposed PEXP method, it achieves exceptionally high and stable performance levels in $H_{\text{score}}$ (medians between 0.8 and 0.95) and $R^2$ score (medians between 0.85 and 0.97), with particularly outstanding results when combined with advanced backbone networks like ViT and DenseNet; its compact box plots also indicate consistency. In terms of $M_{\text{score}}$, the proposed PEXP also demonstrates the best performance, especially when applied to CNN models (such as MobileNet, ResNet, DenseNet), where medians can reach a high level of 0.7 to 0.78. However, it is noteworthy that the proposed PEXP's $M_{\text{score}}$ when combined with the ViT model (median 0.6 to 0.62), while still leading other methods' performance on ViT, is slightly lower compared to the proposed PEXP's $M_{\text{score}}$ on CNN models and its own excellent $H_{\text{score}}$ and $R^2$ scores. This might reflect specific challenges in the ranking consistency of feature importance for ViT model explanations.

In summary, the proposed PEXP method achieves comprehensive and statistically significant improvements in key dimensions of explainability, including feature identification accuracy, model behavior fidelity, and feature ranking consistency. Furthermore, this once again substantiates the assertion of Theorem 1, that employing a strategy of generating perturbed samples based on data distribution helps to improve the stability of the explanation method's performance.

Fig. 10 aims to compare the effectiveness of the proposed explainable method through statistical experiments. A consolidated analysis of these statistical experimental results for $H_{\text{score}}$ (subplot a), $M_{\text{score}}$ (subplot b), and $R^2$ (subplot c) reveals that the proposed PEXP method consistently and significantly surpasses all compared methods, including Baseline [31], LIME [27], KernelSHAP [28], HIDTWM [35], and RealEXP [30], across all three core evaluation metrics and for various backbone network models (MobileNet, ResNet, DenseNet and ViT). Furthermore, this performance advantage is highly statistically significant ($P < 0.001$). Overall, the performance on these metrics generally improves progressively with the evolution of explanation methods from earlier stages to the proposed PEXP.

Specifically, for the proposed PEXP method, it achieves exceptionally high and stable performance levels in $H_{\text{score}}$

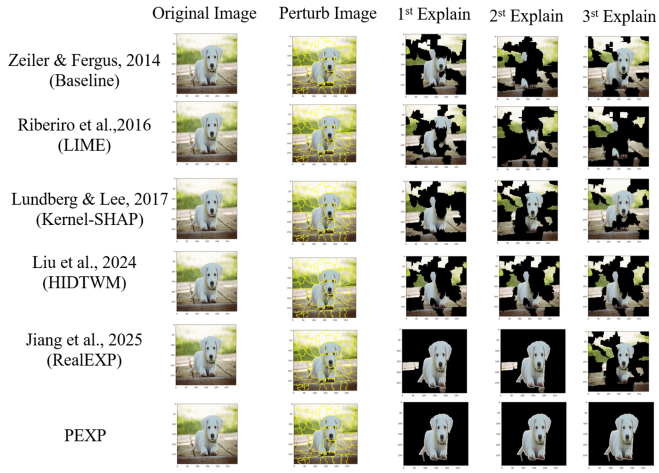



Fig. 11. A visual example.

(medians between 0.8 and 0.95) and $R^2$ score (medians between 0.85 and 0.97), with particularly outstanding results when combined with advanced backbone networks like ViT and DenseNet; its compact box plots also indicate consistency. In terms of $M_{\text{score}}$, the proposed PEXP also demonstrates the best performance, especially when applied to CNN models (such as MobileNet, ResNet, DenseNet), where medians can reach a high level of 0.7 to 0.78. However, it is noteworthy that the proposed PEXP's $M_{\text{score}}$ when combined with the ViT model (median 0.6 to 0.62), while still leading other methods' performance on ViT, is slightly lower compared to the proposed PEXP's $M_{\text{score}}$ on CNN models and its own excellent $H_{\text{score}}$ and $R^2$ scores. This might reflect specific challenges in the ranking consistency of feature importance for ViT model explanations.

In summary, the proposed PEXP method achieves comprehensive and statistically significant improvements in key dimensions of explainability, including feature identification accuracy, model behavior fidelity, and feature ranking consistency. Furthermore, this once again substantiates the assertion of Theorem 1, that employing a strategy of generating perturbed samples based on data distribution helps to improve the stability of the explanation method's performance.

Fig. 11 aims to, through a concise visual example, intuitively compare the quality of explanations generated by different interpretable methods and emphatically highlight the superiority of the proposed PEXP method. Observing the visual

TABLE IV
COMPARISON OF DIFFERENT PERTURBATION DATA GENERATION STRATEGIES ON ANIMAL-10 AND ANIMAL-151 DATASETS

| **Animal-10** | | | | |
|---|---|---|---|---|
| Strategy | $J \uparrow$ | $R^2 \uparrow$ | $H_{\text{score}} \uparrow$ | $M_{\text{score}} \uparrow$ |
| Random | 0.72 | 0.74 | 0.66 | 0.63 |
| Fixed | 0.82 | 0.83 | 0.77 | 0.69 |
| Data Distribution | 0.88 | 0.87 | 0.82 | 0.73 |
| **Animal-151** | | | | |
| Strategy | $J \uparrow$ | $R^2 \uparrow$ | $H_{\text{score}} \uparrow$ | $M_{\text{score}} \uparrow$ |
| Random | 0.72 | 0.64 | 0.53 | 0.55 |
| Fixed | 0.80 | 0.77 | 0.55 | 0.51 |
| Data Distribution | 0.88 | 0.76 | 0.59 | 0.52 |

TABLE V
COMPARISON OF DIFFERENT PTOXY MODELS ON ANIMAL-10 AND ANIMAL-151 DATASETS

| **Animal-10** | | | | |
|---|---|---|---|---|
| Proxy Model | $J \uparrow$ | $R^2 \uparrow$ | $H_{\text{score}} \uparrow$ | $M_{\text{score}} \uparrow$ |
| Ridge Regression | 0.88 | 0.72 | 0.66 | 0.63 |
| Decision Tree | 0.88 | 0.74 | 0.68 | 0.66 |
| Random Forest | 0.88 | 0.83 | 0.71 | 0.69 |
| XGBoost | 0.88 | 0.83 | 0.79 | 0.71 |
| Parallel Tree | 0.88 | 0.87 | 0.82 | 0.73 |
| **Animal-151** | | | | |
| Proxy Model | $J \uparrow$ | $R^2 \uparrow$ | $H_{\text{score}} \uparrow$ | $M_{\text{score}} \uparrow$ |
| Ridge Regression | 0.88 | 0.62 | 0.46 | 0.48 |
| Decision Tree | 0.88 | 0.72 | 0.55 | 0.44 |
| Random Forest | 0.88 | 0.73 | 0.56 | 0.35 |
| XGBoost | 0.88 | 0.74 | 0.55 | 0.51 |
| Parallel Tree | 0.88 | 0.76 | 0.59 | 0.52 |

explanations generated by the proposed PEXP for the puppy in the image, it is clear that its understanding and presentation of the image subject far surpass those of other compared methods. The proposed PEXP's explanations accurately and completely focus on the puppy itself; its highlighted key regions precisely delineate the puppy's contours, containing almost no irrelevant background interference and omitting no important parts of the subject. Crucially, in the three repeated explanations, the proposed PEXP consistently provides highly uniform results, which fully demonstrates the stability and reliability of its explanations. The above provides a comparison of our proposed method with the latest relevant research. In the next section, we will introduce the ablation studies.

### *E. Ablation Experiments*

This section will introduce the ablation experiment. It is worth noting that considering the cost of inference, the explanation model of the experiment is set to MobileNet.

Table IV clearly demonstrates the significant performance improvements achieved by employing the "Data Distribution" (perturbation based on data distribution) strategy compared to "Random" and "Fixed" strategies on both the Animal-10 and Animal-151 datasets. The evaluation is based on four metrics: $J$, $R^2$, $H_{\text{score}}$ and $M_{\text{score}}$.

Particularly noteworthy is that on the Animal-10 dataset, the "Data Distribution" strategy exhibited a comprehensive leading advantage, achieving the highest scores across all evaluation metrics ($J = 0.88$, $R^2 = 0.87$, $H_{\text{score}} = 0.82$, $M_{\text{score}} = 0.73$). This significantly surpassed the other two compared methods, fully demonstrating its exceptional capabilities in this scenario. Even on the more challenging Animal-151 dataset, where scores for all metrics were generally lower, the "Data Distribution" strategy still showcased its strong competitiveness and core advantages: it continued to maintain a leading position in the key metric of $J$, as high as 0.88, far exceeding other methods. Although its $R^2$ score (0.76) was very close to and only slightly lower than the top score from the "Fixed" strategy (0.77), and its $M_{\text{score}}$ (0.52) was also slightly below the specific performance of the "Random" strategy (0.55), considering "Data Distribution's" significant lead in the other two important metrics and its perfect performance on the Animal-10 dataset, its overall performance remains highly outstanding and robust.

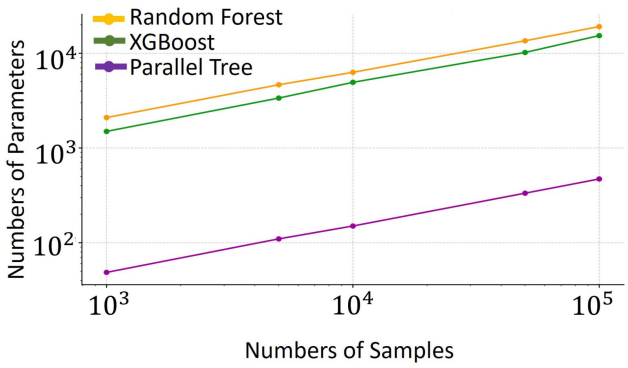
These results strongly demonstrate that the perturbation strategy based on data distribution is an efficient and, in most cases, more superior method, capable of providing a more reliable and powerful benchmark for interpretability analysis. The results of the ablation studies also once again validated the theoretical advantages proposed in Theorem 1 and Theorem 2.

Table V aims to clearly demonstrate the significant advantages of Parallel Tree as a proxy model by comparing the performance of different proxy models, including Ridge Regression, Decision Tree, Random Forest, XGBoost, and our focus, Parallel Tree on the Animal-10 and Animal-151 datasets. The evaluation is based on four key metrics: $J$, $R^2$, $H_{\text{score}}$, and $M_{\text{score}}$. A noteworthy phenomenon is that the instability of explanation results is related to the sampling method and not the model, which also validates Theorem 1 and Theorem 2.

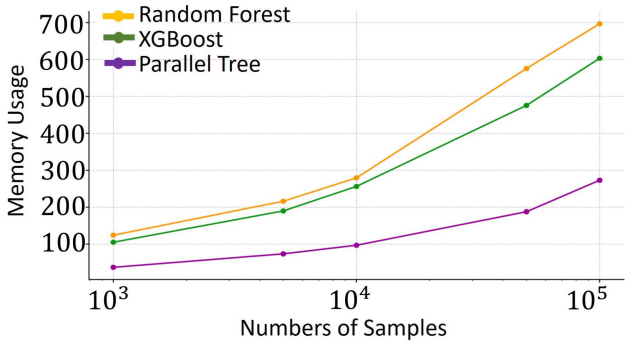
When we examine the other three metrics that can better differentiate the performance of proxy models, the superiority of Parallel Tree becomes unequivocally apparent. On the Animal-10 dataset, Parallel Tree achieved the highest scores in all three metrics of $R^2$ (0.87), $H_{\text{score}}$ (0.82), and $M_{\text{score}}$ (0.73), significantly surpassing all other compared models, including XGBoost. Similarly, on the more challenging Animal-151 dataset, where overall scores were generally lower, Parallel Tree once again demonstrated a comprehensive lead, with its $R^2$ score at 0.76, $H_{\text{score}}$ at 0.59, and $M_{\text{score}}$ at 0.52, each higher than all other proxy models.

In summary, Parallel Tree consistently achieves optimal performance in the three key explanation quality assessment dimensions of $R^2$, $H_{\text{score}}$, and $M_{\text{score}}$ across two datasets of varying complexity.

Fig. 12 aims to compare the complexity of Parallel Tree with commonly used ensemble proxy models such as Random Forest and XGBoost. In terms of parameter count (subplot (a)), Parallel Tree demonstrates an overwhelming advantage: its parameter count is approximately 500 at a sample size of $10^6$, far lower than Random Forest (over $2 \times 10^4$) and XGBoost (approximately $1.5 \times 10^4$), differing by almost one to two orders of magnitude.




(a) Parameter count comparison.




(b) Memory usage comparison.

Fig. 12. Comparison of model complexity for Parallel Tree, Random Forest, and XGBoost. The plots show (a) the number of parameters and (b) memory usage as the sample size increases

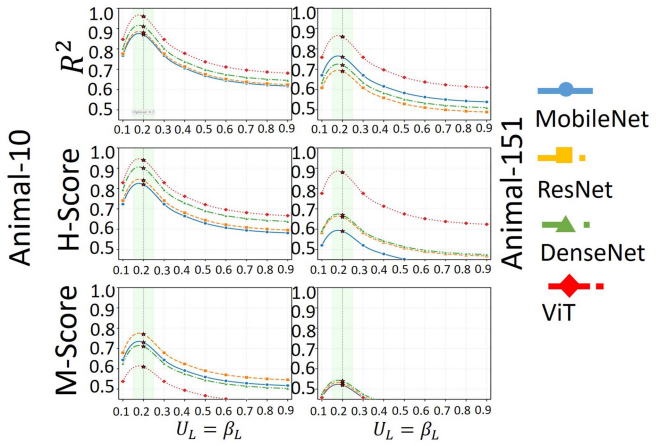



Fig. 13. Impact of hyperparameters $\mu_L$ and $\beta_L$ on the performance of the explanation method across different models and datasets.

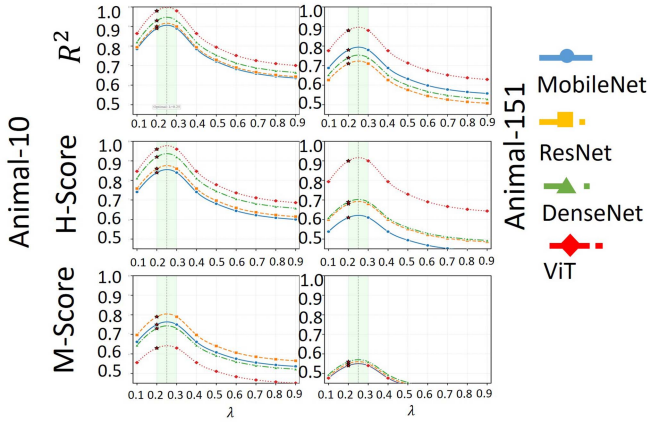



Fig. 14. Impact of hyperparameter λ on the performance of the explanation method across different models and datasets.

Similarly, in terms of memory usage (subplot (b)), Parallel Tree's consumption is also significantly lower: at a sample size of $10^6$, its memory consumption is approximately 270 MB, whereas Random Forest and XGBoost consume as much as approximately 700 MB and 600 MB, respectively. These results clearly indicate that the Parallel Tree model is more concise and lightweight. This implies that it is not only easier to understand itself, faster to train, and has a lower risk of overfitting, but also exhibits higher operational efficiency and better scalability in practical deployment, especially in resource-constrained environments and when handling large-scale datasets. Therefore, this intuitive and compelling experimental data strongly re-validates the theoretical advantages of the Parallel Tree model regarding low complexity and high efficiency, as proposed in Theorem 2.

Fig. 13 illustrates the impact of different values of hyperparameters $\mu_L$ and $\beta_L$ (where both are always set to the same value in the experiments) on the performance of the explanation method on the Animal-10 and Animal-151 datasets. The figure includes four different backbone network models and presents the results for three core performance metrics: $R^2$, $H_{\text{score}}$, and $M_{\text{score}}$, respectively. A consistent trend can be observed across all six subplots: regardless of the dataset, backbone model, or performance metric being evaluated, the performance of the explanation method typically exhibits a parabolic-like shape, initially increasing and then decreasing as the parameter value ($\mu_L = \beta_L$) increases. This implies the existence of an optimal parameter value range where the explanation method achieves its best performance. The light green shaded area and the star markers on each curve (representing peak performance points) in Fig. 13 clearly indicate that when the parameter value $\mu_L = \beta_L = 0.2$, all models generally reach or approach their peak performance across all metrics. This suggests that setting both $\mu_L$ and $\beta_L$ to 0.2 is a relatively robust and optimal hyperparameter choice across different datasets, backbone models, and evaluation dimensions. When the parameter value is less than 0.2 or greater than 0.2 and gradually moves further away, all performance metrics show varying degrees of decline.

Fig. 14 aims to investigate how these performance metrics change with the parameter λ. A common pattern can be observed from all six subplots: regardless of the dataset, the specific backbone model, or the performance metric being evaluated, the performance of the explanation method typically shows a trend of first increasing and then gradually decreasing as the value of parameter λ increases. This clearly indicates that selecting an optimal value or an optimal range for λ is crucial for achieving the best explanation results. The light green shaded area and the star markers on each curve in the figure further clarify

these optimal values. When the parameter $\lambda = 0.25$, all models generally reach or are very close to their peak performance across the three metrics of $R^2$, $H_{\text{score}}$, and $M_{\text{score}}$; therefore, $\lambda = 0.25$ is considered a key optimal hyperparameter setting for this dataset. For the Animal-151 dataset, the data indicate that the optimal $\lambda$ value tends more towards the vicinity of 0.2. When the value of parameter $\lambda$ deviates from these respective optimal values for each dataset, the performance metrics typically show varying degrees of decline.This also validates Theorem 2 and Theorem 7 in the Appendix regarding the analysis of kernel bandwidth.

## V. Discussion and Limitation

Currently, the distribution-aware perturbation strategy based on superpixel segmentation and Laplace noise is only applicable to continuous, spatially structured image data and cannot be directly transferred to discrete text tokens or heterogeneous tabular features. However, at a conceptual level, our proposed "distribution-aware perturbation and parallel ensemble trees" framework is inherently data-agnostic: as long as "perturbations" are defined by reasonable sampling at the word or column level, and sample weights are measured using word-embedding similarity or columns' conditional distributions, one can train tree models on the weighted samples to achieve equally stable and efficient explanations. In future work, we plan to incorporate language-model-based word-level replacements and feature-distribution-based column-level perturbations, and to build corresponding tree-based surrogate models for text classification and structured-data tasks in order to validate and refine this approach.

On the Animal-10 and Animal-151 datasets, PEXP, like other leading explanation methods, exhibits relatively low fine-grained matching scores (see Tables I and II). This is primarily because the superpixels are too coarse to match the exact boundaries marked by experts. The proposed distribution-aware perturbations and tree ensembles are tuned for overall explanation stability. As a result, they focus on large, salient regions and may miss subtle semantic differences. Nevertheless, the proposed PEXP still achieves the highest fine-grained matching score compared to other methods.

In future work, we plan to adopt multi-scale, edge-aware superpixel segmentation to better delineate detailed regions and to develop a hybrid attention-tree surrogate model by integrating attention weights from pretrained vision models with ensemble tree importances, thereby improving alignment accuracy for small but critical areas in complex images.

Moreover, while the proposed PEXP shows advantages in visual explainability for CNN-based and Transformer-based architectures, its performance and interpretability under highly multimodal or language-dominant tasks (e.g., video–text retrieval or document classification) remain unexplored. In such contexts, alternative approaches like SHAP or LIME may offer better flexibility due to their model-agnostic nature and compatibility with structured or sequential symbolic data. Additionally, methods such as Grad-CAM++ or RISE may outperform PEXP when the target model produces sparse or highly localized saliency signals, such as in fine-grained object detection or medical imaging tasks with low inter-region variance.

We acknowledge that these limitations highlight the importance of selecting appropriate explanation strategies based on task characteristics, model architecture, and input modality. A dedicated exploration of method selection trade-offs will be part of our future research agenda.

The proposed framework relies on sample-based perturbation generation and interprets model importance by observing the resulting variations. Although such methods can be applied to various deep learning models, they inevitably incur per-sample computational overhead, hindering strict real-time operation in large-scale scenarios. Despite implementing a series of significant optimizations that have yielded a substantial speed-up over previous studies, this class of methods fundamentally cannot meet hard real-time latency requirements. In future work, we will explore adaptive or streaming perturbation strategies to further reduce per-frame computation and close the gap toward true real-time deployment.

Practical Resource Constraints and Overheads. While PEXP's parallel tree–based architecture delivers substantial speed-ups in surrogate model training and explanation generation, its real-world performance is inherently bound by the underlying compute infrastructure. Factors such as the number and type of available GPUs/CPUs, memory capacity and bandwidth, and interconnect latency can limit the degree of parallelism achievable. In distributed or multi-node deployments, communication and synchronization overhead among workers may increasingly offset speed-up as the ensemble size grows. Moreover, the data-distribution perturbation sampling and kernel-weighting stages introduce additional memory and compute costs that must be factored into deployment planning. In our two-GPU experiments, for example, sample generation and inter-device data transfers accounted for roughly 20% of the total runtime. To mitigate these constraints, PEXP can leverage asynchronous execution, model sharding, and optimized data pipelines. Future work will quantitatively evaluate these overheads under varying resource budgets and explore advanced scheduling strategies to maximize performance under realistic hardware limits.

*Memory–Performance Trade-off:* As shown in Fig.30 (see Appendix), the efficiency is demonstrated under controlled experimental settings with moderate model and dataset scales; when scaling to multimodal tasks or very large ensemble sizes, memory bottlenecks from response-matrix construction and inter-device communication may still arise, requiring further optimization such as streaming perturbations or asynchronous pipelines.

*Boundary Conditions:* As shown in Fig. 14, As shown in Fig. 14, although the experimental results on the two benchmark datasets are consistent with the conclusions of Theorems 2 and 7, whether the method can remain stable under more complex distributional scenarios remains an open question. Future work may investigate the use of automated optimization techniques to address this challenge.

## VI. Conclusion

This paper introduces a Parallel Explainer (PEXP) that effectively enhances interpretation stability and computational efficiency through data distribution perturbation and parallel tree construction techniques. Its primary advantage lies in outperforming other mainstream methods on the Animal-10 and Animal-151 image benchmark datasets, particularly in terms of runtime speed and explanation accuracy. The model-agnostic design of PEXP allows its application across various deep learning architectures. This technology demonstrates significant application potential in fields such as smart cities and security monitoring, providing crucial support for building trustworthy AI systems. Notably, the proposed PEXP enables enhanced interpretability of a modified Transformer module. Furthermore, the proposed PEXP not only achieves breakthroughs in performance but also provides a theoretical basis for its methodology through mathematical theorems, reflecting a combination of theory and practice.

## References


[1] J. Si, J. Yang, Y. Xiang, L. Li, B. Tu, and R. Zhang, "ConDTC: Contrastive deep trajectory clustering for fine-grained mobility pattern mining," *IEEE Trans. Big Data*, vol. 11, no. 2, pp. 333–344, Apr. 2025.

[2] L. Bai, X. Song, and L. Zhu, "Joint multi-feature information entity alignment for cross-lingual temporal knowledge graph with BERT," *IEEE Trans. Big Data*, vol. 11, no. 2, pp. 345–358, Apr. 2025.

[3] J. Ji, J. He, M. Lei, M. Wang, and W. Tang, "Spatio-temporal transformer network for weather forecasting," *IEEE Trans. Big Data*, vol. 11, no. 2, pp. 372–387, Apr. 2025.

[4] Z. Wang, N. Dong, J. Sun, W. Knottenbelt, and Y. Guo, "zkFLzkFL: Zero-knowledge proof-based gradient aggregation for federated learning," *IEEE Trans. Big Data*, vol. 11, no. 2, pp. 447–460, Apr. 2025.

[5] P. Du, Y. Gao, L. Li, and X. Li, "SGAMF: Sparse gated attention-based multimodal fusion method for fake news detection," *IEEE Trans. Big Data*, vol. 11, no. 2, pp. 540–552, Apr. 2025.

[6] A. Rahmani, A. J. Olejniczak, and G. R. Weckman, "AGE: Age-gender effect on faculty career progression in American universities," *IEEE Trans. Big Data*, vol. 11, no. 2, pp. 606–619, Apr. 2025.

[7] G. Sun, Y. Cong, C. Gu, X. Tang, Z. Ding, and H. Yu, "Hierarchical lifelong machine learning with 'watchdog'," *IEEE Trans. Big Data*, vol. 9, no. 1, pp. 63–74, Feb. 2023.

[8] X. Ding, X.-H. Hu, and V. Gudivada, "A machine learning based framework for verification and validation of massive scale image data," *IEEE Trans. Big Data*, vol. 7, no. 2, pp. 451–467, Jun. 2021.

[9] H. Wang, Y. Qi, L. Yao, Y. Wang, D. Farina, and G. Pan, "A human–machine joint learning framework to boost endogenous BCI training," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 12, pp. 17534–17548, Dec. 2024.

[10] L. Chen, W. Zhang, Y. Song, and J. Chen, "Machine learning for human–machine systems with advanced persistent threats," *IEEE Trans. Human-Mach. Syst.*, vol. 54, no. 6, pp. 753–761, Dec. 2024.

[11] X. Yang and C. Zhu, "Industrial expert systems review: A comprehensive analysis of typical applications," *IEEE Access*, vol. 12, pp. 88558–88584, 2024.

[12] W. Shang, T. Gong, J. Hou, J. Lu, and Z. Cao, "Quantitative evaluation method for industrial control system vulnerability based on improved expert elicitation and fuzzy set method," *IEEE Access*, vol. 11, pp. 101007–101019, 2023.

[13] G. A. Tsihrintzis, E. Sarmas, V. Marinakis, D. P. Panagoulias, E.-A. Tsichrintzi, and M. Virvou, "Energy urban domain: Personalized evaluation of expert and non-expert stakeholder interaction with artificial intelligence through ChatGPT using the VIRTSI model," *IEEE Access*, vol. 13, pp. 62506–62526, 2025.

[14] Y. Cao, Z. Zhou, S. Tang, P. Ning, and M. Chen, "On the robustness of belief-rule-based expert systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 10, pp. 6043–6055, Oct. 2023.

[15] J. Wu, J. Wang, H. Shen, and J. H. Park, "Imitation-based reinforcement learning for Markov jump systems and its application," *IEEE Trans. Circuits Syst. I., Reg. Papers*, vol. 71, no. 8, pp. 3810–3819, Aug. 2024.

[16] R. A. Alnanih and L. A. Elrefaei, "Interactive learning tool for system of linear equations: Bridging the gap between novice and expert through MATLAB-Based solutions," *IEEE Access*, vol. 12, pp. 185307–185327, 2024.

[17] L. Wen, J. Liang, K. Yao, and Z. Wang, "Black-box adversarial attack on graph neural networks with node voting mechanism," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 10, pp. 5025–5038, Oct. 2024.

[18] G. Liu, A. Khreishah, F. Sharadgah, and I. Khalil, "An adaptive black-box defense against Trojan attacks (TrojDef)," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 4, pp. 5367–5381, Apr. 2024.

[19] S. Raponi, N. C. Rakotonirina, J. Rapin, C. Doerr, and O. Teytaud, "Optimizing with low budgets: A comparison on the black-box optimization benchmarking suite and OpenAI gym," *IEEE Trans. Evol. Comput.*, vol. 29, no. 1, pp. 91–101, Feb. 2025.

[20] Z. Xu and J. Zhai, "DCVAE-adv: A universal adversarial example generation method for white and black box attacks," *Tsinghua Sci. Technol.*, vol. 29, no. 2, pp. 430–446, Apr. 2024.

[21] Z. Wang, Z. Fu, and J. Xu, "Efficient superpixel-based seamline detection for large-scale image stitching," *IEEE Geosci. Remote Sens. Lett.*, vol. 22, 2025, Art. no. 6005305.

[22] A. Ahmad, A. Teredesai, and C. Eckert, "Interpretable machine learning in healthcare," in *Proc. IEEE Int. Conf. Healthcare Inf.*, New York, NY, USA, 2018, pp. 559–560.

[23] A. Thampi, *Interpretable AI: Building Explainable Machine Learning Systems. Shelter Island*, NY, USA: Manning, 2022.

[24] J. Niu, "Research on loan prediction based on interpretable machine learning," in *Proc. 4th Int. Conf. Mach. Learn., Big Data Bus. Intell.*, Shanghai, China, 2022, pp. 356–360.

[25] T. Ahad, H. B. Kibria, and M. Y. Mehemud, "MultiClass classification of chest diseases using CXR images with DenseNet201 CNN and grad CAM visualization," in *Proc. IEEE Int. Conf. Power, Electr., Electron. Ind. Appl.*, Rajshahi, Bangladesh, 2024, pp. 368–372.

[26] Q. Wang, X. Gao, and X. Li, "An interpretable deep Bayesian model for facial micro-expression recognition," in *Proc. 8th Int. Conf. Control Robot. Eng.*, Niigata, Japan, 2023, pp. 91–94.

[27] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, San Francisco, CA, USA, Aug. 2016, pp. 1135–1144.

[28] S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," in *Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, vol. 30, pp. 4765–4774.

[29] J. Lee et al., "Illuminating the black box: An interpretable machine learning based on ensemble trees," *Expert Syst. Appl.*, vol. 272, 2025, Art. no. 126720.

[30] Z. Jiang et al., "RealExp: Decoupling correlation bias in shapley values for faithful model interpretations," *Inf. Process. Manage.*, vol. 62, no. 4, Jul. 2025, Art. no. 104153.

[31] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, Zurich, Switzerland, Sep. 2014, pp. 818–833.

[32] V. Petsiuk, A. Das, and K. Saenko, "RISE: Randomized input sampling for explanation of black-box models," in *Proc. Brit. Mach. Vis. Conf.*, Newcastle, U.K., Sep. 2018, Art. no. 151.

[33] M. Ancona, C. Oztireli, and M. Gross, "Explaining deep neural networks with a polynomial time algorithm for Shapley values approximation," in *Proc. Int. Conf. Mach. Learn.*, Long Beach, CA, USA, Jun. 2019, pp. 272–281.

[34] M. R. Zafar and N. Khan, "Deterministic local interpretable model-agnostic explanations for stable explainability," *Mach. Learn. Knowl. Extr.*, vol. 3, no. 3, pp. 525–541, Jul. 2021.

[35] Y. Liu, T. Wang, and F. Chu, "Hybrid machine condition monitoring based on interpretable dual tree methods using Wasserstein metrics," *Expert Syst. Appl.*, vol. 235, Jan. 2024, Art. no. 121104.

[36] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff, "Top-down neural attention by excitation backprop," *Int. J. Comput. Vis.*, vol. 126, no. 10, pp. 1084–1102, Oct. 2018.

[37] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis.*, Venice, Italy, Oct. 2017, pp. 618–626.

[38] A. Binder, G. Montavon, S. Lapuschkin, K.-R. Müller, and W. Samek, "Layer-wise relevance propagation for neural networks with local renormalization layers," in *Proc. Int. Conf. Artif. Neural Netw.*, Barcelona, Spain, Sep. 2016, pp. 63–71.
[39] A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. Int. Conf. Mach. Learn.*, Sydney, NSW, Australia, Aug. 2017, pp. 3145–3153.
[40] D. Smilkov, N. Thorat, B. Kim, F. Viegas, and M. Wattenberg, "SmoothGrad: Removing noise by adding noise," Jun. 2017, *arXiv:1706.03825*.
[41] S. C. Bakchy et al., "Colon cancer detection using a Lightweight-CNN with Grad-CAM visualization," in *Proc. 3rd Int. Conf. Adv. Electr. Electron. Eng.*, Dhaka, Bangladesh, Feb. 2024, pp. 1–6.
[42] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[43] Y. Engel and S. Mannor, "Learning embedded maps of Markov processes," in *Proc. Int. Conf. Mach. Learn.*, Williamstown, MA, USA, Jun. 2001, pp. 138–145.
[44] A. Bibal, V. Delchevalerie, and B. Frénay, "DT-SNE: T-SNE discrete visualizations as decision tree structures," *Neurocomputing*, vol. 529, pp. 101–112, Mar. 2023.
[45] M. Wu, X. Liu, and J. Liu, "Weakly supervised audio-visual violence detection," *IEEE Trans. Multimedia*, vol. 25, pp. 1674–1685, 2023.
[46] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," Apr. 2017, *arXiv:1704.04861*.
[47] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[48] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, Jul. 2017, pp. 2261–2269.
[49] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent.*, May 2021, pp. 1–18.
[50] S. Gupte and J. Paparrizos, "Understanding the black box: A deep empirical dive into Shapley value approximations for tabular data," *Proc. ACM Manag. Data*, vol. 3, no. 3, pp. 1–31, Sep. 2025.
[51] D. Wang, Y. Xia, W. Pedrycz, Z. Li, and Z. Yu, "Feature similarity group-class activation mapping (FSG-CAM): Clarity in deep learning models and enhancement of visual explanations," *Expert Syst. Appl.*, vol. 282, Jun. 2025, Art. no. 127553.
[52] K. Rawal, Z. Fu, E. Delaney, and C. Russell, "Evaluating model explanations without ground truth," in *Proc. ACM Conf. Fairness, Accountability, Transparency*, Mar. 2025, pp. 3400–3411.
[53] S. Mitra, A. Sukul, S. K. Roy, P. Singh, and V. K. Verma, "ScoreCAM : Gated score-weighted visual explanations for CNNs," in *Proc. IEEE/CVF Conf. Comput. Vis.*, Pattern Recognit., Jun. 2025, pp. 2700–2709.
[54] L. Yang et al., "Interactive exploration of CNN interpretability via coalitional game theory," *Sci. Rep.*, vol. 15, no. 1, Jan. 2025, Art. no. 9261.
[55] D. Loreti and G. Visani, "Parallel approaches for a decision tree-based explainability algorithm," *Future Gener. Comput. Syst.*, vol. 158, pp. 308–322, Sep. 2024.
[56] O. Sagi and L. Rokach, "Approximating XGBoost with an interpretable decision tree," *Inf. Sci.*, vol. 572, pp. 522–542, Sep. 2021.
[57] A. I. Weinberg and M. Last, "Interpretable decision-tree induction in a Big Data parallel framework," *Int. J. Appl. Math. Comput. Sci.*, vol. 27, no. 4, pp. 737–748, Dec. 2017.

**Wen-Dong Jiang** (Graduate Student Member, IEEE) received the BS degree from the Department of Multimedia and Game Science Development, Lunghwa University of Science and Technology, Taoyuan, Taiwan, in 2021, and the MS degree in computer science and information engineering from Ming Chuan University, Taoyuan, Taiwan, in 2023. Since 2022, he has been working toward the second BS degree in software engineering with the Fayette Institute of Technology, USA, and the PhD degree with the Department of Computer Science and Information Engineering, Tamkang University, New Taipei, Taiwan. His research focuses on interpretable machine learning and its applications in smart cities.

**Chih-Yung Chang** (Member, IEEE) received the PhD degree in computer science and information engineering from the National Central University, Taoyuan, Taiwan, in 1995. He is currently a distinguished professor with the Department of Computer Science and Information Engineering and the Department of Artificial Intelligence, Tamkang University, New Taipei, Taiwan. His research interests include artificial intelligence, deep learning, machine learning, Internet of Things, and wireless sensor networks.

**Tzu-Chia Huang** is currently working toward the PhD degree with the Department of Computer Science and Information Engineering, Tamkang University, New Taipei, Taiwan. Her research focuses on machine learning.

**Yu-Ting Chin** received the MS and PhD degrees in computer science and information engineering from Tamkang University, New Taipei, Taiwan, in 2010 and 2017, respectively. He is currently an assistant professor with the Department of Artificial Intelligence, Tamkang University. His research interests include Internet of Things, artificial intelligence, ad hoc wireless networks, and software-defined networking.

**Diptendu Sinha Roy** (Senior Member, IEEE) received the PhD degree in engineering from the Birla Institute of Technology, Mesra, India, in 2010. In 2016, he joined as an associate professor with the Department of Computer Science and Engineering, National Institute of Technology Meghalaya, Shillong, India, where he has been the chair since 2017. His research interests include software reliability, distributed and cloud computing, and Internet of Things, with an emphasis on applications of artificial intelligence and machine learning for smart integrated systems.