## A. Additional Results: Explaining distilled and residual features on Plant Disease Dataset

In the experimental sections, we evaluated our proposed method to explain and quantify Knowledge Distillation (KD) on well-established datasets such as ASIRA and CIFAR10, showing its versatility and effectiveness in various scenarios. To further validate the generalisability of our method, especially for identifying distilled and residual features, we also applied it to the plant disease classification. This additional analysis confirmed the suitability of the proposed explainability technique in challenging datasets. In this section, we summarise the results, highlighting the ability of our method to detect salient features essential for diagnosing plant diseases and demonstrating its wide applicability to real-world problems.

We generate additional results to visualise distilled and residual features for plant disease images. Fig. A.1 shows that the Student model (ResNet-50) more accurately localises salient features compared to the Base model (ResNet-50). Specifically, the distilled features predominantly highlight regions relevant for accurate prediction, whereas the residual features tend to be distributed over areas irrelevant to the prediction. This implies that the Student model learns to ignore the features that are not useful for the prediction and focus on the salient parts. These saliency maps align with the findings presented in the main body of our work (in Fig. 4 and Fig. 6), explaining that the Student model consistently learns features of better relevance across different datasets compared to its equivalent Base model.

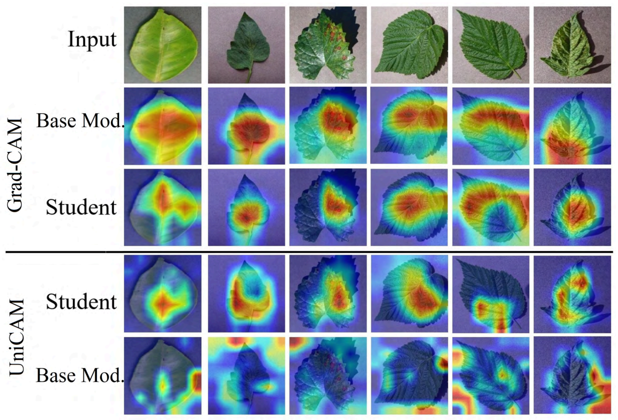


Figure A.1. Sample visualisation of unique (distilled and residual) features in Plant disease classification.

Next, we analyse a specific case of the *Strawberry Leaf Scorch* plant disease classification, and the Student shows an improved focus on the crucial signs of the disease on the leaves (Fig. A.3). The Student model can detect more relevant features for diagnosing Potato Early Blight and Strawberry Leaf Scorch.

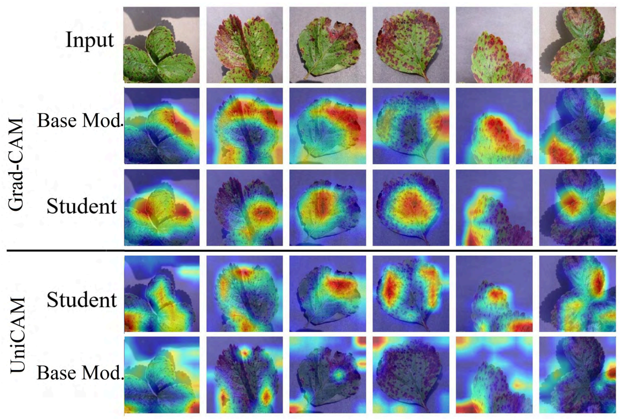


Figure A.2. Strawberry Leaf Scorch

Figure A.3. Sample visualisation of distilled and residual features on Strawberry Leaf Scorch plant disease classification.

In our further analysis, we visualised the distilled and residual features from Layer-3 and Lyer-2(Fig.A.4). The distilled features mainly localise the diseased areas of the input image, even though they are challenging to locate. Meanwhile, the residual features highlighted the areas of the leaf image that had little impact on plant disease classification. To conclude, the proposed novel visual explanation and quantitative metrics help to explain and quantify the knowledge that the Student model learned and failed to learn from the Teacher model during Knowledge Distillation.

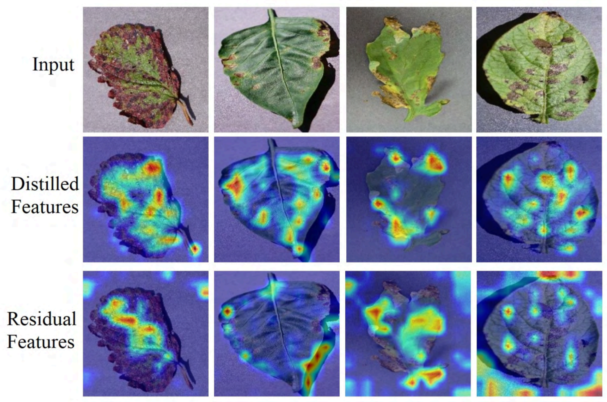
## B. Distance and Partial Distance Correlation

Here, we provide the detailed steps of distance and partial distance correlation following Szelkely et al. [39]. Distance correlation (dCor) measures linear and nonlinear associations or dependence between two random vectors. For an observed random sample $(x, y) = (X_k, Y_k) : k = 1, \dots, n$ drawn from a distribution of random vectors, the empirical distance correlation $R_n^2(x, y)$ for $n$ samples is derived from the distance covariance of the samples. To compute the distance covariance between the samples, we first compute the $n$ by $n$ distance matrices $(a_{j,k})$ and $(b_{j,k})$ containing all pairwise distances:

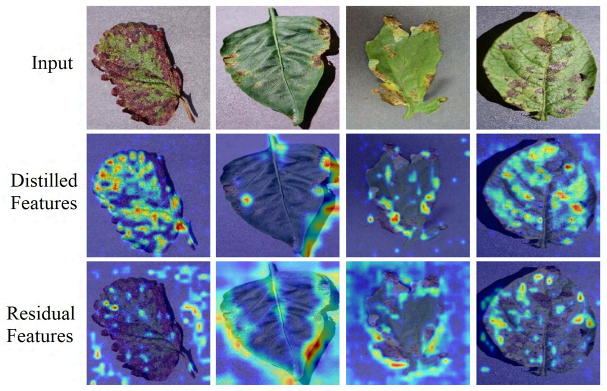
$$\begin{aligned} a_{j,k} &= \|X_j - X_k\|, j, k = 1, 2, \cdots, n, \\ b_{j,k} &= \|Y_j - Y_k\|, j, k = 1, 2, \cdots, n \end{aligned} \tag{B.1}$$

where $\|.\|$ represents the Euclidean norm. Taking all the doubly centred distances as:

$$\begin{aligned} A_{j,k} &:= a_{j,k} - \overline{a}_{j\cdot} - \overline{a}_{\cdot k} + \overline{a}_{\cdot\cdot}, \\ B_{j,k} &:= b_{j,k} - \overline{b}_{j\cdot} - \overline{b}_{\cdot k} + \overline{b}_{\cdot\cdot}, \end{aligned} \tag{B.2}$$



(a) Layer-3



(b) Layer-2

Figure A.4. Sample visualisation of Distilled and residual features on Plant disease classification from Layer-3 and Layer-2

where $\overline{a}_{j\cdot}$ is the $j^{th}$ row mean, $\overline{a}_{\cdot k}$ is the $k^{th}$ column mean, and $\overline{a}_{\cdot\cdot}$ is the grand mean of the distance matrix of the sample $x$ (the notation is similar for $b$ of sample $y$), then we compute the squared sample distance covariance $V(x, y)$ as the arithmetic average of the products $A_{j,k}B_{j,k}$:

$$V_n^2(x, y) = \frac{1}{n^2} \sum_{j,k=1}^{n} A_{j,k}B_{j,k}. \tag{B.3}$$

and the distance variance $V(x)$ and $V(y)$ as follows:

$$\text{dVar}_n^2(x) := V_n^2(x, x) = \frac{1}{n^2} \sum_{j,k=1}^{n} A_{j,k}^2. \tag{B.4}$$

$$\text{dVar}_n^2(y) := V_n^2(y, y) = \frac{1}{n^2} \sum_{j,k=1}^{n} B_{j,k}^2. \tag{B.5}$$

Distance variance is a measure of the complexity or diversity of a single random vector, while distance covariance is a measure of the dependence or similarity between two random vectors. Distance variance is a special case of distance covariance when the two random vectors are identical, meaning that they have the same information and knowledge. In our context, distance covariance and distance variance functions are used to measure the amount of knowledge transferred from Teacher to student models during knowledge distillation. They calculate the degree of dependence or similarity between the Teacher and student models and the degree of complexity or diversity within each model based on their outputs or features. The meaning of the result is as follows:

- A high distance covariance between Teacher and student models means that they have a high degree of similarity or alignment in their information and knowledge, which implies a successful knowledge transfer.
- A low distance covariance between Teacher and student models means that they have a low degree of similarity or alignment in their information and knowledge, which implies an unsuccessful knowledge transfer or a potential overfitting or underfitting problem.
- A high distance variance for either the Teacher or student model means that it has a high degree of complexity or diversity in its information and knowledge, which implies a high capacity or expressiveness of the model.
- A low distance variance for either the Teacher or student model means that it has a low degree of complexity or diversity in its information and knowledge, which implies a low capacity or expressiveness of the model.

**Definition 1** *(Distance correlation) [39]. For an observed random sample $(x, y) = (X_k, Y_k) : k = 1, \ldots, n$ drawn from a distribution of random vectors $X$ in $\mathbb{R}^p$ and $Y$ in $\mathbb{R}^q$, the empirical distance correlation $R_n^2(x, y)$ for $n$ samples is defined as:*

$$R_n^2(x, y) = \begin{cases} \frac{V_n^2(x,y)}{\sqrt{V_n^2(x,x)V_n^2(y,y)}} & , V_n^2(x, x)V_n^2(y, y) > 0 \\ 0 & , V_n^2(x, x)V_n^2(y, y) = 0 \end{cases} \tag{B.6}$$

where $V_n^2(x, y)$, $V_n^2(x, x)$ and $V_n^2(y, y)$ are the squared sample distance covariance. Distance correlation is zero when the random vectors are independent and one when they are dependent, indicating a strong correlation between each other.

Distance correlation (dCor) is a measure of the similarity of the information contained in two random variables. However, in some situations, we may want to measure the association between two random vectors after adjusting for a third random vector. This leads to the concept of partial distance correlation (pdCor), which is an extension of dCor proposed by Szekely et al. [39]. They introduced a Hilbert space where the squared distance covariance is an inner product and showed how to obtain *U-centered* matrices $\tilde{A}$ from the distance matrices $(a_{j,k})$ such that their inner product is the distance covariance.

**Definition 2** *($U-centered$ matrix) [39]: Let $A = (a_{j,k})$ be a symmetric, real valued $n \times n$ matrix with zero diagonal, $n > 2$. The U-centred matrix of $A$ at the $(j,k)^{th}$ entry is defined by:*

$$\tilde{A}_{j,k} = \begin{cases} a_{j,k} - \frac{1}{n-2}\sum_{l=1}^{n} a_{l,k} - \frac{1}{n-2}\sum_{i=1}^{n} a_{j,i} \\ + \frac{1}{(n-1)(n-2)}\sum_{l,i=1}^{n} a_{l,i}, & j \neq k, \\ 0, & j = k \end{cases} \tag{B.7}$$

*and the inner product between $\tilde{A}, \tilde{B}$ is defined as $(\tilde{A} \cdot \tilde{B}) := \frac{1}{n(n-3)}\sum_{j \neq k} \tilde{A}_{j,k}\tilde{B}_{j,k}$,*

**Definition 3** *(Partial distance correlation) [39]: Let $(x, y, z)$ be random variables observed from the joint distribution of $(X, Y, Z)$, then the partial distance correlation between $x$ and $y$ controlling for $z$ (assuming it as a confounding variable) is given by:*

$$R^{*2}(x, y; z) := \begin{cases} \frac{(P_z^{\perp}(x) \cdot P_z^{\perp}(y))}{\|P_z^{\perp}(x)\| \|P_z^{\perp}(y)\|} & , \|P_z^{\perp}(x)\| \cdot \|P_z^{\perp}(y)\| \neq 0, \\ 0 & , Otherwise \end{cases} \tag{B.8}$$

*where, $P_z^{\perp}(x) = \tilde{A} - \frac{(\tilde{A} \cdot \tilde{C})}{(\tilde{C} \cdot \tilde{C})}\tilde{C}$, $P_z^{\perp}(y) = \tilde{B} - \frac{(\tilde{B} \cdot \tilde{C})}{(\tilde{C} \cdot \tilde{C})}\tilde{C}$ denotes the orthogonal projection of $\tilde{A}(x)$ and $\tilde{B}(y)$ onto $\tilde{C}(z)^{\perp}$ respectively, and $(P_z^{\perp}(x) \cdot P_z^{\perp}(y)) = \frac{1}{n(n-3)}\sum_{j \neq k}(P_z^{\perp}(x)_{j,k})P_z^{\perp}(y))_{j,k})$ is sample partial distance covariance.*

To summarise, we utilised distance correlation (dCor), a robust statistical method, to measure the degree of association between the Base model and the Student during Knowledge Distillation (KD). Furthermore, we used partial distance correlation (pdCor) to extract the distilled and residual features. This approach, coupled with gradient-based visual explainability techniques, helped to propose *UniCAM*, which explains the KD process and provides a better understanding of the knowledge it acquired and overlooked during KD.

### B.1. Theoretical Basis of Feature Subtraction in UniCAM

The subtraction operation in the formulation of *UniCAM* is used to remove the shared feature representations between the Student and the Base model or vice versa, identifying features unique to each model. This approach is conceptually similar to orthogonal projection in linear algebra, where a vector is decomposed into components: one that lies along a reference direction and another orthogonal to it. In this case, consider the features $x_s$ (Student features) and $x_b$ (Base model features). The shared features between $x_s$ and $x_b$ are represented by their projection:

$$\text{proj}_{x_b}(x_s) = \frac{\langle x_s, x_b \rangle}{\langle x_b, x_b \rangle} x_b, \tag{B.9}$$

where $\langle \cdot, \cdot \rangle$ represents the inner product, and this term quantifies the component of $x_s$ aligned with $x_b$.

Now, $x_s$ can be decomposed into two orthogonal components: 1. The component is aligned with $x_b$ (shared features): $\text{proj}_{x_b}(x_s)$. 2. The component orthogonal to $x_b$ (unique features): $x_s|unique = x_s - \text{proj}_{x_b}(x_s)$.

Thus, the subtraction is valid and justified because:

$$x_s = \text{proj}_{x_b}(x_s) + (x_s - \text{proj}_{x_b}(x_s)),$$

where $\text{proj}_{x_b}(x_s)$ identifies the shared features, and $x_s - \text{proj}_{x_b}(x_s)$ gives the unique features.

In *UniCAM*, we work in the transformed space of pairwise distance matrices and the features are adjusted for mutual influence using a U-centered distance matrix, $P^{(s)}$ and $P^{(b)}$. This provides a robust mechanism to capture the relational structure of the features rather than their absolute values. This approach is invariant to shifts or rotations in the feature space, and the analysis focuses on the geometric relationships between features. The shared features are calculated as follows:

$$\text{Shared}_{x_s} = \frac{\langle P^{(s)}, P^{(b)} \rangle}{\langle P^{(b)}, P^{(b)} \rangle} P^{(b)}. \tag{B.10}$$

The unique features of the Student, after removing the shared features, are:

$$x_{s|unique} = P^{(s)} - \text{Shared}_{x_s}. \tag{B.11}$$

This subtraction extracts the component of $P^{(s)}$ orthogonal to $P^{(b)}$, preserving only the unique features of $x_s$ that do not exist in $x_b$. The operation is mathematically valid due to the properties of vector spaces, where such decomposition is meaningful in terms of orthogonal projections.

## C. Steps on Feature Extraction

In this section, we explain the step-by-step feature extraction from the relevant regions (which was briefly introduced in Eq. 7, Section 3.2 in the main paper). Given a Base model or a Student, then we can extract the features as follows:

$$\hat{x} = f(I \odot \mathcal{H}) \tag{C.1}$$

where $I$ is the input image, $\mathcal{H}$ is the saliency map generated using *UniCAM*, $\odot$ is the element-wise multiplication operator, and $f$ is a feature extraction function. We extract the features by applying perturbation technique [34] that modifies the input image $I$ by replacing each pixel $I_{ij}$ with the weighted average of its neighbouring pixels in the highlighted region as follows:

$$I'_{ij} = \sum_{k,l} w_{kl} I_{kl} \tag{C.2}$$

where $w_{kl}$ is a weight that depends on the relevance of pixel $I_{kl}$ for prediction. The relevance of each pixel is determined by the saliency map generated using *UniCAM*, which produces a heatmap $\mathcal{H}$ that assigns a value to each pixel based on its contribution to the relevant features learned by one model. The higher the value, the more relevant the pixel is. The weight $w_{kl}$ is proportional to $\mathcal{H}$, such that:

$$w_{kl} = \frac{\mathcal{H}_{kl}}{\sum_{k,l} \mathcal{H}_{kl}} \tag{C.3}$$

Therefore, we can write Eq. C.2 as:

$$I'_{ij} = \sum_{k,l} \frac{\mathcal{H}_{kl}}{\sum_{k,l} \mathcal{H}_{kl}} I_{kl}. \tag{C.4}$$

We can simplify this equation by using element-wise multiplication and division operators and rewrite Eq. C.4 as follows:

$$I' = \frac{I \odot \mathcal{H}}{\sum \mathcal{H}}, \tag{C.5}$$

where $\sum \mathcal{H}$ is a scalar that represents the sum of all elements in $\mathcal{H}$. This equation shows how we obtain a modified image $I'$ that contains only the features of interest for each model. To extract these features into a vector representation, we apply a feature extraction function $f$ to $I'$:

$$\hat{x} = f(I'). \tag{C.6}$$

We substitute Eq. C.5 into Eq. C.6 to obtain:

$$\hat{x} = f(\frac{I \odot \mathcal{H}}{\sum \mathcal{H}}). \tag{C.7}$$

Since $\sum \mathcal{H}$ is a scalar, we can ignore it for feature extraction purposes, as it does not affect the relative values of the pixels. Therefore, we simplify Eq. C.7 to:

$$\hat{x} = f(I \odot \mathcal{H}). \tag{C.8}$$

This is the step-by-step formulation to extract features from the regions identified as relevant using *UniCAM* or other gradient-based visual explainability.

Fig. C.1 illustrates the results of the perturbed images $I' = I \odot \mathcal{H}$ for plant disease classification. First, we use *UniCAM* to extract and explain the distilled and residual features from the original images. The distilled features focus on the diseased areas of the leaves, which are crucial for diagnosis, while the residual features are spread across the background or non-essential parts of the leaves. Next, we apply a perturbation technique proposed by Rong et al. [34] to modify the images based on pixel relevance. We then feed the modified images to the Student or Base model and obtain the corresponding feature vectors, $\hat{x}_s$ or $\hat{x}_b$. Finally, we use *FSS* and *RS* to measure the feature similarity and relevance of the feature vectors. The heatmap intensity for distilled features indicates a higher contribution to the classification decision, resulting in more distinct and less perturbed images of the critical areas for diagnosis after perturbation.

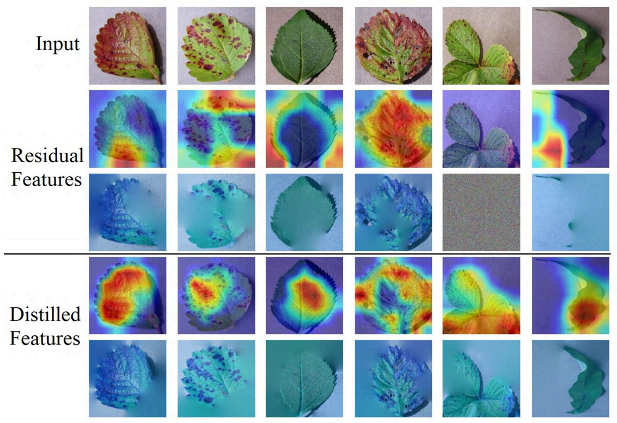


Figure C.1. The residual and distilled features.

## D. Experimental Details

### D.1. Training Setup

For our experiments, we employed three widely-used KD techniques: Response-based KD, Overhaul feature-based KD, and Attention-based KD. The training of the Student model followed an offline KD setup, where the Teacher model was pre-trained before being used to guide the Student. For the Teacher model, we used the Cross Entropy (CE) loss to optimise its performance based solely on the training data.

The Student model's training incorporated both the CE loss for the training data and the additional loss function specific to the KD technique applied, as recommended in their respective literature. For Response-based KD, the distillation loss minimised the divergence between the Teacher and Student outputs. Overhaul feature-based KD introduced intermediate feature-level supervision, and Attention-based KD utilised attention maps from the Teacher to align the Student's attention patterns.

In cases where the Student and Teacher shared the same architecture, we used the Teacher model as the Base model for comparison. When the Teacher and Student had different architectures, the Base model was trained using the same experimental settings as the Student, except without the Teacher's guidance, and optimised using only the CE loss.

### D.2. Selecting Layers

We mainly employ ResNet family models, ResNet-18, ResNet-50, and ResNet-101, as the main convolutional neural network architectures for our implementation. We generate Grad-CAM and *UniCAM* outputs from the last residual blocks in each of the four layers of the ResNet models. We denote these blocks as L1, L2, L3 and L4 and the details of these blocks are as follows:

**ResNet-18:** This network has four layers with 2 residual blocks each. Each residual block has two convolutional layers, batch normalisation and ReLU activation. The first convolutional layer has a kernel size of $3 \times 3$ and preserves the number of channels. The second convolutional layer has a kernel size of $3 \times 3$ and also preserves the number of channels. The skip connection may have a convolutional layer to match the dimensions of the input and output. The last blocks in each layer have 64, 128, 256 and 512 output channels, respectively. Hence, each layer $L_i$ represents the following:

- **L1**: This block is the second and last block in the first layer. It has two convolutional layers with 64 output channels each.
- **L2**: This block is the second and last block in the second layer. It has two convolutional layers with 128 output channels each.
- **L3**: This block is the second and last block in the third layer. It has two convolutional layers with 256 output channels each.
- **L4**: This block is the second and last block in the fourth layer. It has two convolutional layers with 512 output channels each.

**ResNet-50:** This network has four layers with 3, 4, 6 and 3 residual blocks, respectively. Each residual block has three convolutional layers with batch normalisation and ReLU activation. The first convolutional layer has a kernel size of $1 \times 1$ and reduces the number of channels by a factor of 4. The second convolutional layer has a kernel size of $3 \times 3$ and preserves the number of channels. The third convolutional layer has a kernel size of $1 \times 1$ and increases the number of channels by a factor of 4. The skip connection may also have a convolutional layer to match the dimensions of the input and output. The last blocks in each layer have 256, 512, 1024 and 2048 output channels, respectively. Hence, each layer $L_i$ represents the following:

- **L1**: This block is the third and last block in the first layer. It has three convolutional layers with 64, 64 and 256 output channels, respectively. The skip connection does not have a convolutional layer.
- **L2**: This block is the fourth and last block in the second layer. It has three convolutional layers with 128, 128 and 512 output channels, respectively.
- **L3**: This block is the sixth and last block in the third layer. It has three convolutional layers with 256, 256 and 1024 output channels, respectively.
- **L4**: This block is the third and last block in the fourth layer. It has three convolutional layers with 512, 512 and 2048 output channels, respectively.

**ResNet-101:** This network has four layers with 3, 4, 23 and 3 residual blocks, respectively. Each residual block has three convolutional layers with batch normalisation and ReLU activation. The first convolutional layer has a kernel size of $1 \times 1$ and reduces the number of channels by a factor of 4. The second convolutional layer has a kernel size of $3 \times 3$ and preserves the number of channels. The third convolutional layer has a kernel size of $1 \times 1$ and increases the number of channels by a factor of 4. The skip connection may have a convolutional layer to match the dimensions of the input and output. The last blocks in each layer have 256, 512, 1024 and 2048 output channels, respectively. Hence, each layer $L_i$ represents the following:

- **L1**: This block is the third and last block in the first layer. It has three convolutional layers with 64, 64 and 256 output channels, respectively.
- **L2**: This block is the fourth and last block in the second layer. It has three convolutional layers with 128, 128 and 512 output channels, respectively.
- **L3**: This block is the twenty-third and last block in the third layer. It has three convolutional layers with 256, 256 and 1024 output channels, respectively.
- **L4**: This block is the third and last block in the fourth layer. It has three convolutional layers with 512, 512 and 2048 output channels, respectively.