Appendix

A. Building the Ontology

A1. Semantic Keyword Vectors For any given input image $\mathbf{x}$, we have a keyword vector $\vec{v}(\mathbf{x}) = [v_1, v_2, \ldots]$ (the value v_i is corresponding to the i^{th} keyword) obtained by pre-trained scene understanding models. In this work, we use $k = 150 + 365 = 515$ keywords (excluding foreground objects). We define the keyword vector in the following format:

- $\vec{v}_{seg} \rightarrow v_1$ to v_{150}: object predictions, e.g., wall, floor, etc. These 150 object predictions are provided by a pre-trained segmentation model on the ADE20k dataset.
- $\vec{v}_{scene} \rightarrow v_{151}$ to v_{525}: scene class predictions, e.g., pond, grassland, etc. These 365 scene class predictions are provided by a pre-trained model on Place365 dataset.

We define the k^{th} keyword as present if $v_k > 1\%$ when the k^{th} keyword is a segmentation keyword, or if $v_k = \max(\vec{v}_{scene})$ when the k^{th} keyword is a scene classification keyword. In Figure 9, we show that on average, one input image has three frequent items. However, this is not enough for background invariance testing, therefore, we build an ontology to expand the keywords.

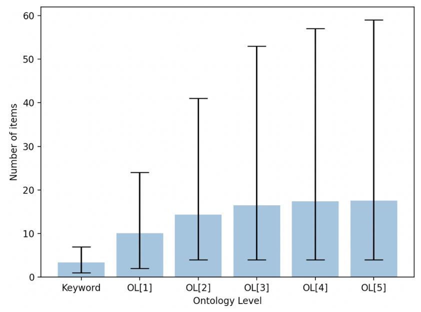

Figure 9. Number (maximum, mean, minimum) of enriched / expanded items after each ontology level.

A2. Databases **For scene understanding**: we use pre-trained models on two databases. In the ADE20k dataset [44], 150 object classes in 20,000 background scenes from the SUN and Places database are annotated. Some labelled objects can be a whole object or a part of another object, e.g., a door can be an indoor picture or a part of a car. Place365 dataset consists of 1,803,460 training images with 365 different scene classes [43]. And the number of images per class varies from 3,068 to 5,000. **For preparing models** to be tested: we use a smaller version of ImageNet, namely IN9 which consists of images from ImageNet but labelled with nine classes [40]. There are 45,405 images on the training set (5045 images per class) and 4185 images on the testing set (465 images per class). They also provide segmented foreground masks for some images on the testing set. Finally, **For background candidates**: we use BG20k which consists of 20,000 background scenes with no foreground images [21].

A3. Association Analysis and Ontology: We use association analysis to build an ontology to help background enrichment in this work. Ontology is a graph based approach to store relationships between different entities, e.g., keywords in our case. Commonly used relationships in an ontology include "α is related to β" or "α has β". In this work, the relationship in our ontology is defined: "Keyword A is related to keyword B in a manner that if A is known to be in an image, B has the conditional probability of $pr(B|A)$ to occur in the same image. This probability is computed as the confidence in association analysis (Eq. 2).

A4. Pre-steps of Building the Ontology

- Run a pre-trained scene understanding model on each of the given input images and generate a keyword vector for the image.
- Run association analysis to obtain frequent items, item-sets, association rules, and the ontology using the Apriori or FP-Growth algorithm [16].
- The ontology is re-built only when necessary, e.g., when new keywords are added into the ontology, or when the distribution of the dataset has changed.

A5. Keyword Expansion The major steps are:

1. For an input image, obtain a keywords vector V_0 using the scene understanding model. Each keyword is marked with [0] indicating level 0. Initialize the overall keywords vector V as $V \leftarrow V_0$, and $i \leftarrow 0$
2. If the overall keywords vector V has reached the minimal number of keywords desired, i.e., $\|V\| \geq$ MINKWS or the number of iterations reached the maximum number allowed, i.e., $i \geq$ MAXITS, stop the process and return the overall keywords vector V. Otherwise go to the next step.
3. We use the ontology to find all keywords that connected to the keywords in the vector V_i. For any newly-found keyword that is not already in V, add it to V_{i+1}, and mark it as $[i+1]$, where $i+1$ indicates the new extended level.
4. $V \leftarrow$ union$(V_0, V_1, \ldots, V_{i+1})$, $i \leftarrow i + 1$.

We use the keyword expansion algorithm to enrich the originally detected keywords V_0 (from the pre-trained scene understanding model). In Figure 9, we show that using the ontology with more iterations can increase the number of total keywords in V. However, due to the limited size of the ontology (the limited size of the database we use to build the ontology), the effect of using the ontology becomes less significant after the 4th iteration, which is consistent with the experiments in Section 5.

Table 3. Experiment results: the automation accuracy using random forest as the assessor is around 80% and the inter-rater reliability score with majority votes is around 0.65. Meanwhile, no significant differences are found using different settings (i.e., kernel size, sigma value) for RBF interpolation.

Interpolator / Assessor	Automation Accuracy			Inter-rater Reliability with Majority Votes		
	Size: 32, σ=2	Size: 16, σ=5	Size: 32, σ=10	Size: 32, σ=2	Size: 16, σ=5	Size: 32, σ=10
Random Forest	79.6±8.1%	78.7±7.7%	**79.7±7.5%**	**0.651±0.103**	0.635±0.116	0.649±0.091
Adaboost	54.5±7.5%	70.9±10.2%	74.8±9.1%	0.358±0.134	0.548±0.159	0.599±0.102
Worst-case accuracy	64.4%			0.387		

B. Ablation Study

B1. Number of Selected Testing Images When use the two sampling approaches (see Section 4.2.1) to search testing scenes, we obtain $N = 32$ testing images via 32 intervals (bins) or 32 keywords. Here we show that we obtain similar results when $N = 50$ or $N = 100$.

As shown in Table 4, selecting different number of testing images leads to similar neuron coverage rates and IRR scores. This is aligned with [31] where fuzzing techniques (e.g., mutation) are used to find adversarial examples to achieve a better coverage rate. Therefore in this work, we choose $N = 32$.

Table 4. Selecting different number of testing images leads to similar Recall, precision and F1 score

	N=32	N=50	N=100
Neuron Coverage (Recall)	0.652	0.657	0.660
Fleiss' Reliability (IRR, Precision)	0.649	0.645	0.645
F1	0.650	0.651	0.652

B2. Different Settings for RBF Interpolation We conduct the ablation study for some parameters of the interpolation, i.e., the kernel size and sigma values. In Figure 13, the three interpolants are (kernel size: 16, sigma: 5, K: 32), (kernel size: 32, sigma: 2, K: 32) and (kernel size: 32, sigma: 10, K: 32) respectively. The K values only affect a few points at the corners and we barely see any difference. As shown in Figure 10, kernel size and sigma value affect how blurry the interpolated scatter plots are. Based on the quality of the interpolated scatter plots, we tested these three interpolants and reported the automation results in table 3 where we show that using different parameters for the interpolation did not heavily affect the automation accuracy (the performance of assessors).

B3. Number of Testing Images To generate the scatter plots for a given model, we use the testing set of IN9 dataset [40]. However, among all the 4185 images in the testing set, only 1712 images are provided with a foreground mask. Therefore, we use the 1712 images to generate the scatter plots. In Figure 13, we show that using different percentages of the 1712 images will not significantly affect the qualities of the generated scatter plots (neither the original nor interpolated). However, when the number of images becomes too low, e.g., ≤25% (less than 430 images), the generated scatter plots are starting to be heavily affected.

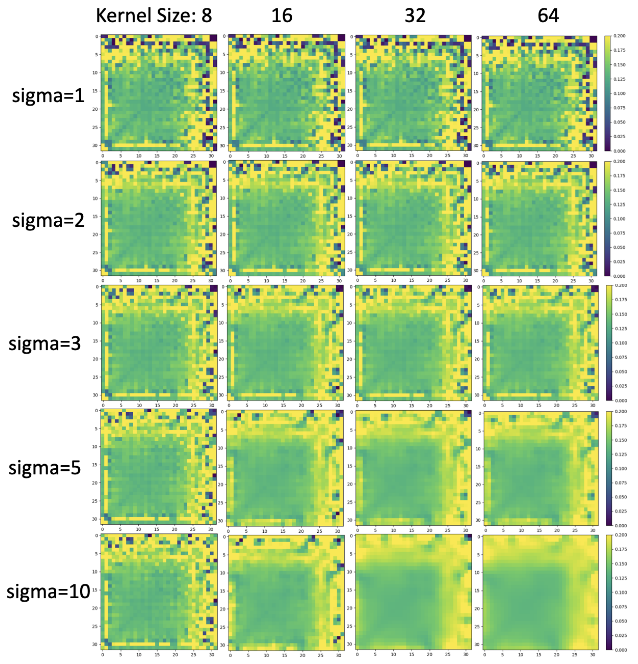

Figure 10. Different interpolant parameters for M_{61}. We use (Kernel size 16, sigma 5), (Kernel size 32, sigma 2), and (Kernel size 32, sigma 10) respectively in Figure 13.

B4. Testing Patterns Across Different Testing Runs Finally, we show that our proposed testing method leads to consistent visualized patterns across different runs (in Figure 12), especially when compared with random sampling (Figure 2). This is also reflected by the relatively higher reliability across annotators.

B5. Distribution of the distances $d_{i,j}$ As shown in Figure 11, the distances are Gaussian distributed. As expected,

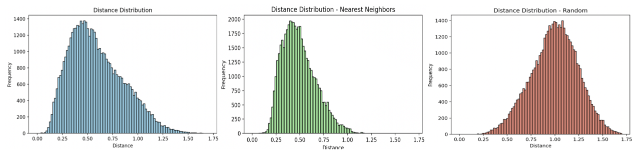

Figure 11. Distribution of distances between original images and synthesized testing images. Left (blue): Association Ontology, middle (green): nearest neighbors, right (red): random.

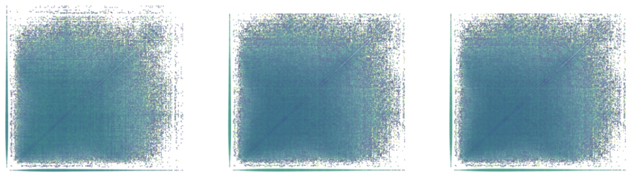

Figure 12. Our proposed methods lead to consistent visualized testing patterns across multiple different testing runs, compared with random sampling (Figure 2)

compared with nearest neighbor, the distribution (of association ontology) suggests that our sampled scenes are more diverse. Compared with random sampling, the distribution shows that the sampled scenes prioritize scenes with a stronger association with the original images. Any methods involving selecting N items from $K(\gg N)$ has a long-tail by definition. Only when the sorting of N is not good, it becomes a problem. In this work, we sample testing scenes based on semantically connected keywords. In our approach, the "semantic connections" depend on the frequency of association.

C. Automated Background Invariance Testing

We build a small model repository of 250 models for object classification. The models were trained under different settings:

- Architectures: VGG13bn, VGG13, VGG11bn, VGG11 [37], ResNet18 [17], and Vision Transformer [10]
- Hyper-parameters: learning rate, batch size, epochs
- Augmentation: rotation, brightness, scaling, using images with only foreground (black pixels as background)
- Optimizers: SGD, Adam, RMSprop
- Loss: cross-entropy loss, triplet loss, adversarial loss

We apply these models to the synthesized testing images. With the two measuring positions per model, we generate four variance matrices using the results from the ML Testing process as mentioned in Section 5. In this section, we discuss our professional annotations based on the variance matrices, statistics of our model repository, and finally analysis on the automation process of the invariance testing procedure.

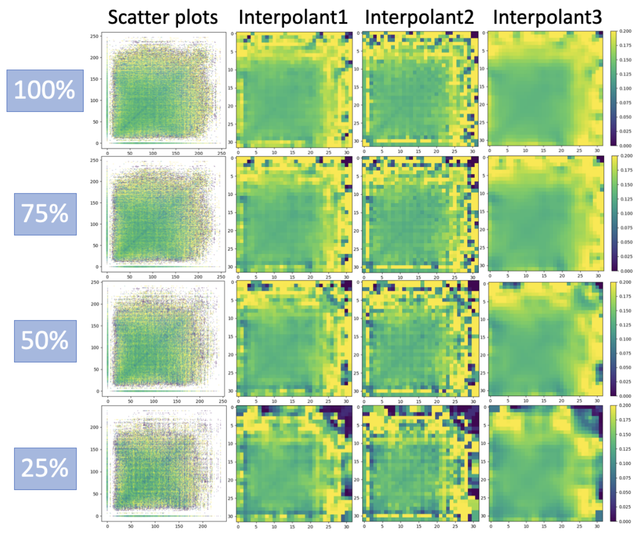

Figure 13. We use different numbers (percentages) of the testing images to generate scatter plots. The number of images will not significantly affect the quality of the generated plots unless the number becomes too small, e.g., $\leq 25\%$. The example we show here is model M_{61}

C1. Professional Annotations We survey the three professionals who provide the annotations for background invariance testing. We ask them the following questions:

- Q1: Which locations are you interested in?
- Q2: Are the interpolated scatter plots helpful? Which interpolant is the most helpful?
- Q3: Are there any common visual patterns?
- Q4: Are the annotations aligned with the worst-case accuracy?
- Q5: Would you say your decisions are consistent? Are you confident about your decisions?
- Q6: Any interesting findings about the models?

Q1: Testing locations All the three coders agree that the final predictions and outputs from the last layer before the final fully connected are important to be tested.

Q2: Helpful interpolated Coder 1 and 2 mention all of the interpolated plots are helpful and there is no particular interpolator that is superior to the others. Coder 3 mentions they check all of the interpolated plots and find the one with kernel size 32 and $\sigma = 10$ helps their judgements the most because it shows the most straightforward global shape of the plots.

Q3: Common visual patterns Coder 1 and 3 mention that the most common patterns are those with a green area (which indicates a lower error rate) at the bottom left, and some yellowish patterns (which indicates a higher error

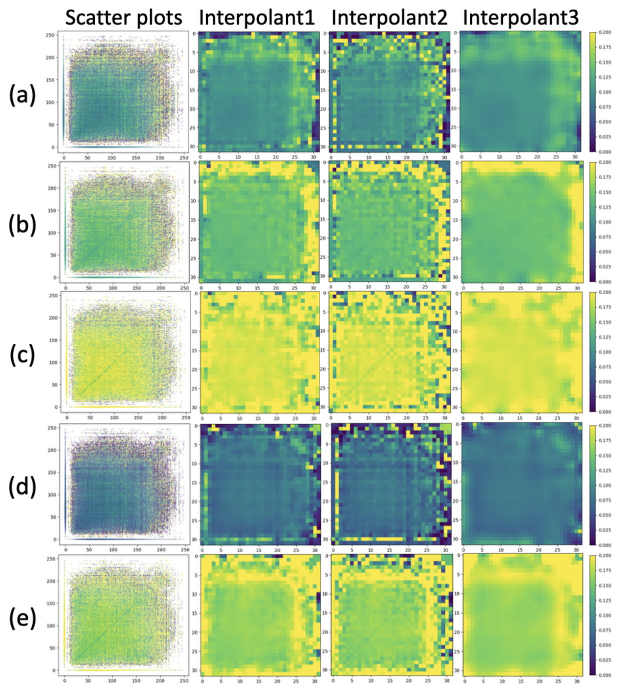

Figure 14. Examples of original scatter plots and interpolated / accumulated scatter plots. From (a) to (e) are model M_1, M_7, M_{16}, M_{68}, M_{71} respectively.

rate) at either the edges or corners at the top right. Coder 2 notices it is common that there is a green line along the primary diagonals (from bottom left to top right) of the plots. We show more different patterns in Figure 14.

Q4: Annotations versus worst-case accuracy All three coders mention their annotations are not completely aligned with worst-case accuracy. However, they all find their annotations and the traditional metric are related to each other to some extent.

Q5: Consistent annotations All three coders are confident about their decisions for the majority of the models. However, they are less confident for those labelled as "borderline".

Q6: Interesting findings Coder 1: vision transformers seem to be naturally more robust against background permutations. Even when the final predictions do not appear to be robust at all, their last layer before the head module can still be robust (dark green). Coder 2: models trained using images without background (black pixels as their background) appear to be more robust than those trained with natural images. Coder 3: metric learning, e.g., triplet loss, does not seem to be helpful even for the convolutional layers that are used to form the triplet loss function.

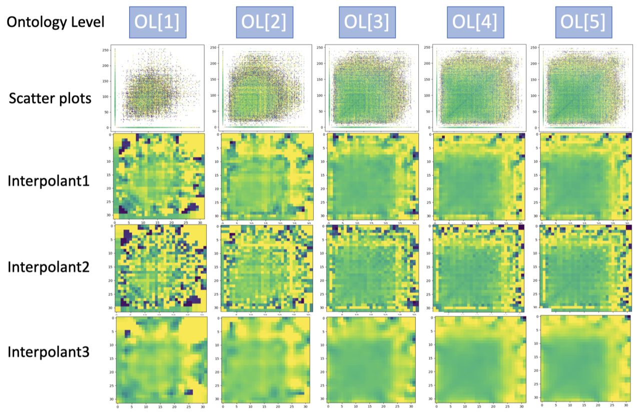

Figure 15. Original scatter plots (first column) and interpolated variance matrices (second to fourth column). The scatter plots are generated using the ontology level one to five, i.e., $OL[1], OL[2]$, $\ldots, OL[5]$ for a randomly selected model M_{61}. Interpolant 1 - 3: (kernel size 16 σ=5), (kernel size 32 σ=2), and (kernel size 32 σ=10).

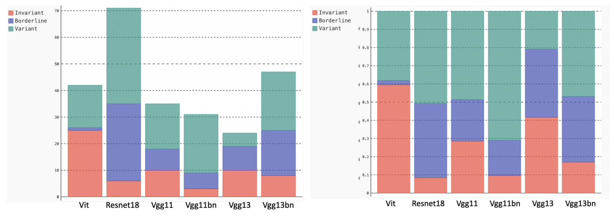

Figure 16. Statistics of model architectures in our model repository. Left: number of models with each architecture in our model repository. Right: percentage of invariant / borderline / variant models for each architecture in our model repository.

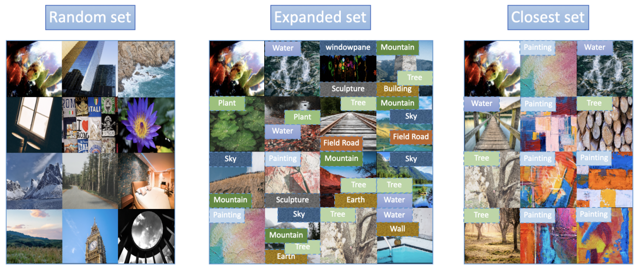

Figure 17. Three sets of example background scenes discovered for a target image of fish (the top-left of each set). The random set includes mostly unsuitable images. The closest set includes those discovered using only the original keywords K_x = {painting, water, tree}. The expanded set includes those discovered using the ontology, showing more suitable background scenes. Note that those keywords found in background scenes should not include any of the foreground objects that the original ML models were trained to classify as specified at the beginning of Section 4.

C2. Model Repository In this section, we show some statistics of our model repository (250 models). As mentioned in the main paper, we trained the models under different settings, and provide professional annotations for each of them as being background-invariant, borderline or background-variant.

Table 5. Statistics of the annotations of the model repository

Invariant	Borderline	Variant	Total
62 (24.8%)	70 (28.0%)	118 (47.2%)	250 (100%)

As shown in Table 5, the model repository is not balanced. There are more background variant models than background invariant models, which is expected as most of the models were trained using no special technique to boost their background invariance qualities; Unlike other types of transformations, e.g., rotation, background augmentation is not commonly used.

Table 6. Statistics of the accuracy of the model repository

	mean	max	min	std
Accuracy	91.88%	99.42%	43.40%	9.11%
Worst-case	28.76%	65.36%	0.18%	11.92%

In Table 6 we show that the average accuracy of the models in the collection is around 91.9%. We did not intentionally make any model not sufficiently trained. However, for some models that have not been pre-trained on ImageNet, the accuracy can drop to lower than 50%.

In Figure 16, we show that in our model repository, we have a wide range of different architectures, namely vision transformer patch16-224, resnet 18, and vgg variants. We also show the statistics for their background variance qualities. However, the size of our model collection is limited and we leave the studies of the architectures' impact on invariance qualities to future work.

C3. Analysis on Automated Assessor Shapley values (a game theoretic approach) [26] are commonly used to explain outputs or predictions of machine learning models. It isolates each input feature and averages its expected marginal contribution. In this section, we report the usage of Shapley values to explain the predictions of the random forest (assessor) we train for prediction background invariance qualities of ML models.

As shown in Figure 19, for an arbitrarily selected model M_5, the assessor's prediction is "background invariant". And the top factor contributing to the predictions is the mean value of the variance matrix generated at pos_0 (final prediction). We also show that, in general, the assessor will consider the mean value of the variance matrix and the error rate (defined as the percentage of the pixels $\geq$ $thres$ [25]). This way, we can have a testing report for the assessor for both the overview analysis and case studies for any interested model(s).

D. Variance Matrices from RBF-based Resampling.

Whilst scatter point clouds are able to display our testing results, they suffer from the problem of overlapping glyphs when the number of points per unit area becomes excessively large [27]. Therefore, analysis and judgement based purely on scatter plots might not be consistent.

To address the problem of overlapping glyphs of scatter point clouds, we use the common approach of radial basis functions (RBFs) to transform a set of point clouds into a variance matrix. For each element e in a variance matrix, an RBF defines a circular area in 2D, facilitating the identification of all data points in the circle. Let these data points be $p_1, p_2, \ldots, p_c$ and their corresponding values are $v_1, v_2, \ldots, v_c$. As discussed earlier, the coordinates of each data point are determined by the semantic distances from the target image to two testing images. A Gaussian kernel ϕ is then applied to these data points, and produces an interpolated value for element e as

$$value(e) = \frac{\sum_{i=1}^{c}(\phi(\|e - p_i\|) \cdot v_k)}{\sum_{i=1}^{c}\phi(\|e - p_i\|)}$$

However, when the RBF has a large radius, the computation can be costly. When the radius is small, there can be cases of no point in a circle. In order to apply the same radius consistently, we define a new data point at each element e and use K nearest neighbors algorithm to obtain its value $u(e)$. The above interpolation function thus becomes:

$$value(e) = \frac{\phi(0) \cdot u(e) + \sum_{i=1}^{c}(\phi(\|e - p_i\|) \cdot v_k)}{\phi(0) + \sum_{i=1}^{c}\phi(\|e - p_i\|)}$$

In Figure 15, we show the application of three different RBFs. The mixed green and yellow patterns in row OL[1] gradually become more coherent towards OL[5]. We can clearly see a green square at the centre and yellow areas towards the top and right edges.

E. Plausibility for Invariance Testing and Future Works

Consider two sets of testing images: X and Y with plausible and implausible background respectively. Model A performs 100% correct with X, and 100% incorrect with Y; Model B performs 100% incorrect with X, and 100% correct with Y. We should always prefer model A than B. For example, a case where a model fails to recognize a car just because it appears in front of a different building is a more serious issue than a case where it fails when the car appears in a bathroom. Therefore in this paper, we focus

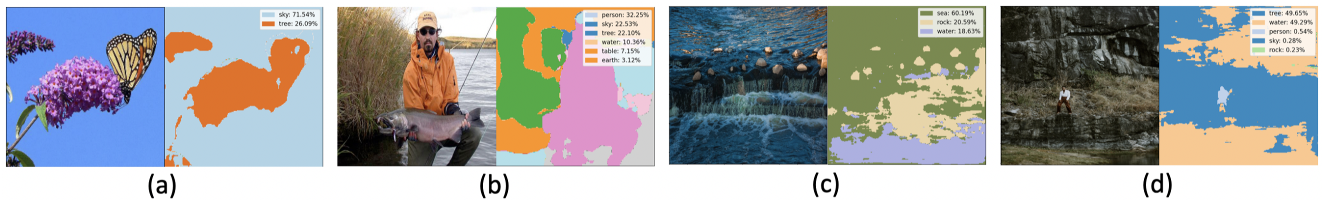

Figure 18. (a), (b) are target images. (c), (d) are background candidates. (a) and (c) have only a few detected keywords.

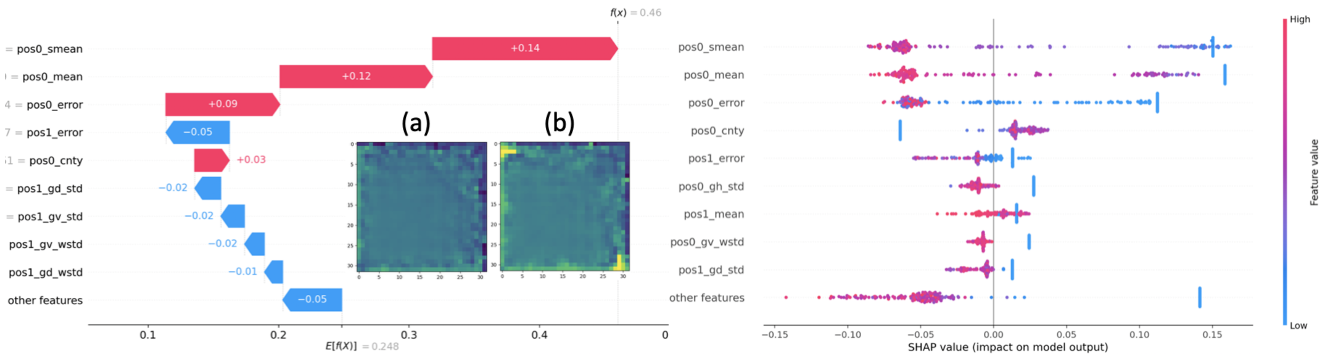

Figure 19. Shapley values of a randomly selected assessor (a random forest). Left: given a model M_5, and its variance matrices generated at (a) the final prediction (pos_0) and (b) the final convolutional layer (pos_1), the assessor predicts the model to be "background invariant". And the top reasons for this prediction is: the mean value of the variance matrix (pos_0) and the error rate (defined in [25]) of the variance matrix (pos_0). Note that there are yellowish areas on the corners of the variance matrices at pos_1, therefore the continuity score and the gradient scores at these two positions have a negative impact on the decision. Right: In general, the assessor's decisions for being "background-invariant" depend on the mean value of the variance matrix (pos_0) and the error rate of the variance matrix (pos_0), which is aligned with the analysis of M_5 on the left.

on invariance testing which takes plausibility into consideration. Figure 17 shows three sets of example background scenes discovered for a targeting image (i.e., the fish image on the top-left corner of each set). While it is not necessary for every testing image in invariance testing to be realistic, the plausibility of a testing image reflects its probability of being captured in the real world. It is unavoidable that invariance testing involves testing images of different plausibility, and therefore it is important to convey and evaluate the testing results with the information of the plausibility [29, 33]. An ideal set of background scenes should have a balanced distribution of scenes of different plausibility. Qualitatively, we can observe that in Figure 17, the random set has too many highly implausible images and the closest set has images biased towards keywords $K_x = \{$painting, water, tree$\}$, many are not quite plausible, while the expanded set has a better balance between more plausible to less plausible background scenes. When generating / synthesizing testing images, we adopted the simple background replacement in this work due to the reason stated in Section 4.2.2. In the future, this can be replaced by generative models when they become more reliable or when more advanced and accountable filtering techniques are available. Finally, in this work, for each target image, we select N items from $K(\gg N)$ keywords based on semantic connections. In our approach, semantic connections depend on the frequency of association. In future work, more advanced techniques can be explored to improve the association analysis, e.g., a human-in-the-loop association mining technique.