Keep CALM and Improve Visual Feature Attribution

Jae Myung Kim1*  Junsuk Choe2*  Zeynep Akata1,3,4  Seong Joon Oh5†

1University of Tübingen  2Department of Computer Science and Engineering, Sogang University
3Max Planck Institute for Intelligent Systems  4Max Planck Institute for Informatics  5NAVER AI Lab

Abstract

The class activation mapping, or CAM, has been the cornerstone of feature attribution methods for multiple vision tasks. Its simplicity and effectiveness have led to wide applications in the explanation of visual predictions and weakly-supervised localization tasks. However, CAM has its own shortcomings. The computation of attribution maps relies on ad-hoc calibration steps that are not part of the training computational graph, making it difficult for us to understand the real meaning of the attribution values. In this paper, we improve CAM by explicitly incorporating a latent variable encoding the location of the cue for recognition in the formulation, thereby subsuming the attribution map into the training computational graph. The resulting model, class activation latent mapping, or CALM, is trained with the expectation-maximization algorithm. Our experiments show that CALM identifies discriminative attributes for image classifiers more accurately than CAM and other visual attribution baselines. CALM also shows performance improvements over prior arts on the weakly-supervised object localization benchmarks. Our code is available at https://github.com/naver-ai/calm.

1. Introduction

Interpretable AI [25, 40, 24, 52] is becoming an absolute necessity in safety-critical and high-stakes applications of machine learning. Along with good recognition and prediction accuracies, we require models to be able to transparently communicate the inner mechanisms with human users. In visual recognition tasks, researchers have developed various feature attribution methods to inspect contributions of individual pixels or visual features towards the final model prediction. Input gradients [55, 56, 58, 5, 63, 39, 28] and input perturbation methods [59, 70, 20, 43, 23, 47] have been actively researched.

In this paper, we focus on the class activation mapping (CAM) [68] method, which has been the cornerstone of the feature attribution research. CAM starts from the observation that many CNN classifiers make predictions by aggregating location-wise signals. For example, \( p(y|x) = \text{softmax} \left( \frac{1}{\pi \text{HW}} \sum_{yhw} f_{yhw} \right) \) where \( f = f(x) \) is the extracted feature map in \( \mathbb{R}^{C \times H \times W} \) where \( C, H, W \) are the number of classes, height, and width of the feature map, respectively. CAM considers the pre-GAP feature map \( f_{yhw} \) as the attribution, after scaling it to the \([0, 1]\) range by dropping the negative values and dividing through by the maximum value: \( s := (\max_{yhw} f_{yhw})^{-1} f^+ \in [0, 1]^{H \times W} \). Thanks to the algorithmic simplicity and reasonable effectiveness, CAM has been a popular choice as an attribution method with many follow-up variants [53, 9, 6, 69, 61, 45, 22].

Despite its popularity and contributions to the interpretability community, CAM still has its own limitations. What does the attribution map \( s \) really mean? We fail to find a reasonable linguistic description because \( s \) hardly encodes anything essential in the recognition process. \( s \) also violates key minimal requirements, or “axioms” [40, 59, 22], for an attribution method. For example, its dependence on the pre-softmax values \( f \) make it ill-defined: translating \( f \mapsto f + c \) yields an identical model because of the translation invariance of softmax, but it changes the attribution map \( s \).

We thus introduce a novel attribution method, class ac-
tivation latent mapping (CALM). It builds a probabilistic graphical model on the last layers of CNNs, involving three variables: input image $X$, class label $Y \in \{1, \cdots, C\}$, and the location of the cue for recognition $Z \in \{1, \cdots, HW\}$. Since there is no observation for $Z$, we consider latent-variable training algorithms like marginal likelihood (ML) and expectation-maximization (EM). After learning the dependencies, we define the attribution map for image $\hat{x}$ as $p(y, z|\hat{x}) \in [0, 1]^{H \times W}$, the joint likelihood of the recognition cue being at $z$ and the class being $y$. CALM has many advantages over CAM. (1) It has a human-understandable probabilistic definition; (2) it satisfies the axiomatic requirements for attribution methods; (3) it is empirically more accurate and useful than CAM.

In our experimental analysis, we study how well CALM localizes the “correct cues” for recognizing the given class of interest. The “correct cues” for recognition are ill-defined in general, making the evaluation of attribution methods difficult. We build a novel evaluation benchmark on pairs of bird classes in CUB-200-2011 [62] where the true cue locations are given by the parts where the attributes for the class pair differ (Figure 1). Under this benchmark and a widely-used remove-and-classify type of benchmark, CALM shows better attribution performances than CAM and other baseline methods. We also show that CALM advances the state of the art in the weakly-supervised object localization (WSOL) task, where CAM has previously been one of the best [13, 12].

In summary, our paper contributes (1) analysis on the lack of interpretability for CAM, (2) a new attribution method CALM that is more interpretable and communicable than CAM, and (3) experimental results on real-world datasets where CALM outperforms CAM in multiple tasks. Our code is available at https://github.com/naver-ai/calm.

2. Related Work

Interpretable AI is a big field. The general aim is to enhance the transparency and trustworthiness of AI systems, but different sub-fields are concerned with different parts of the system and application domains. In this paper, we develop a visual feature attribution method for image classifiers based on deep neural networks. It is the task of answering the question: “how much does each pixel or visual feature contribute towards the model prediction?”

Gradient-based attribution. Feature attribution with gradients dates back to the pioneering works by Sung [60] and Baehrend et al. [7]. The first explicit application to CNNs is the work by Simonyan et al. [55]. Input gradients consider local linearization of the model, but it is often not suitable for CNNs because the local behavior hardly encodes the complex mechanisms in CNNs for e.g. more global perturbations on the input. Follow-up works have customized the backpropagation algorithm to improve the attribution performances: Guided Backprop [57], LRP [5], Deep Taylor Decomposition [39], SmoothGrad [56], Full-Gradient [58], and others [65, 30, 54, 63, 3, 28]. We make an empirical comparison against key prior methods in this domain.

Perturbation-based attribution. Researchers have developed methods for measuring the model response to non-local perturbations. Integrated Gradients [59] measure the path integral of model responses to global input shifts. Another set of methods consider model responses to redacted input parts: sliding windows of an occlusion mask [70] and random-pixel occlusion masks [43]. Since the occluding patterns introduce artefacts that may mislead attributions, different options for redaction have been considered: “meaningful perturbations” like image blurring [20, 19], inpainting [70], and cutting-and-pasting a crop from another image of a different class [23]. Some of the key methods above are included as baselines for our experiments.

CAM-based attribution. Gradients and perturbations analyze the model by establishing the input-output relationships. Class activation mapping (CAM) [68] takes a different approach. Many CNNs have a global average pooling (GAP) layer towards the end. CAM argues that the pre-GAP features represent the discriminativeness in the image. Related works have considered variants of the last-layer modifications like max pooling [41] and various thresholding strategies [17, 16, 6]. GradCAM [53] and GradCAM++ [9] have later expanded the usability of CAM to networks of any last-layer modules by combining the widely-applicable gradient method with CAM. In this work, we identify issues with CAM and suggest an improvement.

Self-explainable models. Above attribution methods provide interpretations of a complex, black-box model in a post-hoc manner. Another paradigm is to design models that are interpretable by design in the first place [18]. There is a trade-off between interpretability and performance [34]; researchers have sought ways to push the boundary on both fronts. One line of work distills the complex, performant model into an interpretable surrogate model such as decision trees [21], sparse linear models [44, 46, 4]. Other works pursue a hybrid approach, where a small interpretable module of a neural network is exposed to humans, while the complex, less interpretable modelling is performed in the rest of the network. ProtoPNet [10] trains an interpretable linear map over prototype neural activations. Concept or semantic bottleneck models [36, 31] enforces an intermediate layer to explicitly encode semantic concepts. Our work is a hybrid self-explainable model based on the interpretable probabilistic treatment of the last layers of CNNs.

Evaluating attribution is challenging because of the lack of ground truths. Early works have resorted to qualitative [23] or human-in-the-loop evaluations [46, 53, 48, 33] with limited reproducibility. Wojciech et al. [51] and subsequent works [30, 43, 46, 26] have proposed a quantita-
tive measure based on the remove-and-classify framework. Along a different axis, researchers have focused on the necessary conditions for attribution methods. They can be either theoretical properties, referred to as “axioms” [59, 22] or empirical properties, referred to as “sanity checks” [2]. In our work, we analyze CALM in terms of the axioms (§4.2) and evaluate it on a remove-and-classify benchmark (§5.3). Additionally, we contribute a new type of evaluation; we compare the attribution map against the known ground-truth attributions on a real-world dataset [62] (§5.2).

**Weakly-supervised object localization (WSOL)** is similar yet different from the feature attribution task. While the latter is focused on detecting the small, class-discriminative cues in the input, the former necessitates the detection of the full object extents. Despite the discrepancy in the objective, WSOL is mathematically identical functions, \( \phi_1 \equiv \phi_2 \), shall pass the same attribution maps, regardless of their implementations. CAM violates this axiom. Assume \( \phi_1(f) := \text{softmax}(\frac{1}{HW} \sum_{hw} f_{yh,w}) \) and \( \phi_2(f) := \text{softmax}(\frac{1}{HW} \sum_{hw} f_{yh,w} + C) \) for some constant \( C \). Since the softmax function is translation invariant, \( \phi_1 \equiv \phi_2 \) for any \( C \). However, the CAM attribution map for \( \phi_2 \) varies arbitrarily with \( C : s = (\max_{hw} f_{h,w} + C)^{-1}(f + C)^{+} \). Min-max normalization is a solution to the problem, but it alone does not let CAM meet other axioms. This observation reveals the inherent limitation of utilizing feature values before softmax (often called “logits”) as attribution.

**Sensitivity axiom** [59, 22] states that if the function response \( \phi(x) \) changes as the result of altering an input value \( x_{hw} \) at \((h, w)\), then the corresponding attribution value \( s_{hw} \) shall be non-zero. Conversely, if the response is not affected, then \( s_{hw} \) shall be zero. CAM fails to satisfy the sensitivity axiom. Depending on the normalization type, CAM assigns zero attributions to \((h, w)\) where \( f_{hw} \) is either negative (for max normalization) or smallest (for min-max normalization). However, being assigned a negative or smallest feature value \( f_{hw} \) has little connection to the insensitivity of the model to the input value \( x_{hw} \).

**Completeness (or conservation) axiom** [59, 22, 54, 5] states that the sum of attributions \( \sum_{hw} s_{hw} \) shall add up to the function output \( \phi(x) = p(y|x) \). The completeness criterion is violated by CALM in general because the summation \( \sum_{hw} s_{hw} \) for \( s \) in Equation 2 do not match \( \phi(x) = p(y|x) \) in Equation 1. In conclusion, CALM fails to satisfy key minimal requirements for an attribution method.

**4. Class Activation Latent Mapping (CALM)**

We fix the above issues by introducing a probabilistic learning framework involving input image \( X \), class label \( Y \), and the cue location \( Z \). We set up a probabilistic graphical model and discuss how each component is parametrized with a CNN. We then introduce learning algorithms to account for the unobserved latent variable \( Z \). An overview of our method is provided in Figure 2.
4.1. Probabilistic inference with latent Z

We define Z as the location index (h, w) of the cue for recognizing the image X as the class Y. Our aim is to let the model explicitly depend its prediction on the features corresponding to the location Z and later on use the distribution of possible cue locations Z as the attribution provided by the model. Z is a random variable over indices (h, w); for simplicity, we use the integer indices Z ∈ {1, ⋯, HW}.

Without loss of generality, we factorize p(x, y, z) as p(y, z|x)p(x) = p(y|x, z)p(z|x)p(x) (graph on the left). The recognition task is then performed via p(y|x) = \sum_z p(y, z|x).

4.1.1 Representing joint distribution with CNNs.

We factorize the joint distribution p(y, z|x) into p(y, z|x) = p(y|x, z)p(z|x) and parametrize p(y|x, z) and p(z|x) as two convolutional branches of a CNN trunk (Figure 2a). Since Y ∈ {1, ⋯, C} and Z ∈ {1, ⋯, HW}, we represent p(y|x, z) as a CNN branch g(x) with output dimensionality C × HW. Likewise, we represent p(z|x) with a CNN branch h(x) with output dimensionality HW. To make sure that the outputs of the two branches are proper distributions, we normalize the outputs with softmax for g and \ell_1 normalization followed by the softplus for h. We broadcast h to all class indices Y and multiply it elementwise with g to get p(y, z|x) (Figure 2a).

4.1.2 Training algorithms

Training a latent variable model is challenging because of the unobserved variable Z. We consider two methods for training such a model: (1) marginal likelihood (ML) [37] and (2) expectation-maximization (EM) [15].

CALM_ML directly minimizes the marginal likelihood

\[ -\log p_\theta(y|x) = -\log \sum_z p_\theta(y|x, z)p_\theta(z|x) \]

which is tractable for the discrete Z. See Figure 2d.

CALM_EM is based on the EM algorithm that generates pseudo-targets for Z to supervise the joint likelihood p(y, z|x). The EM algorithm introduces two running copies of the parameter set: \( \theta \) and \( \theta' \). The first signifies the model of interest, while the latter often refers to a slowly updated parameter used for generating the pseudo-targets for Z. The learning objective is

\[ -\log p_\theta(y|x) \leq -\sum_z p_\theta(z|x, y) \log p_\theta(y, z|x) \]

\[ = -\sum_z g_{yz} \cdot h_z \log (g_{yz} \cdot h_z) \]
where \( q' \) and \( h' \) denote the parametrization with \( \theta' \). Note that Equation 6 is the pixel-wise negative log likelihood, the loss function for semantic segmentation networks \([11]\). One may interpret the objective as self-supervising the pixel(\( z \))-wise predictions \( p(y, z|x) \) with its own estimation of the cue location \( z \) for the true class \( y \): \( \psi'_{y}(z|x, y) \). In practice, we use the current-iteration model parameter \( \theta = \theta' \) to generate the pseudo-target for \( Z \). See Figure 2d for an overview of the process. Even with \( \theta = \theta' \), we need to apply \( T\cdot \text{detach()} \) to block the gradient flow through the pseudo-target \( p_{\psi'}(z|x, y) \), as required by Equation 6.

A similar framework appears in the weakly-supervised semantic segmentation task. Papandreou et al. [42] have generated pseudo-target label maps to train a segmentation network. CALM\(_{EM} \) is different because our location-encoding latent \( Z \) takes integer values, while their \( Z \) takes values in the space of all binary masks; our formulation admits an exact computation of Equation 6, while theirs require an additional approximation step.

4.1.3 Inferring feature attributions

Unlike CAM, our probabilistic formulation enables principled computation of the attribution map as part of the probabilistic inference on \( p(y, z|x) \). \( Z \) is explicitly defined as the location of the cue for recognition. For CALM, the attribution score \( s_z \) for location \( z \) is naturally defined as the joint likelihood given the ground-truth class \( \hat{y} \)

\[
    s_z := p(\hat{y}, z|x),
\]

or in human language,

“The probability that the cue for recognition was at \( z \) and the ground truth class \( \hat{y} \) was correctly predicted for the image \( x \).”

Note that the definition is far more communicable than the one for CAM in §3.1. See Figure 2c for visualization.

Apart from the attribution map, one may compute additional interesting quantities. We show examples in Figure 3. Treating \( z \) as a free variable, the conditional attribution \( p(y|x, z) \) is explained as the likelihood of the cue being at position \( z \), given the prediction for image \( x \) as \( y \). The saliency \( p(z|x) \) encodes the likely location of any cue for recognizing classes \( y \in \{1, \cdots, C\} \) in image \( x \). It is the sum over all attribution maps for classes \( y \):

\[
    p(z|x) = \sum_{y} p(y, z|x). \quad \text{One may also compute the partial sum for classes } y \in \mathcal{Y} \text{ to obtain the subset attribution to highlight specific image regions of interest } p(z, \mathcal{Y}|x) := \sum_{y \in \mathcal{Y}} p(z, y|x).\]

Above quantities are later utilized for the weakly-supervised object localization (WSOL) task in §5.4. It is also possible to reason why the class label for input \( x \) is \( \hat{y} \) instead of \( y' \) by computing the counterfactual attribution \( p(\hat{y}, z|x) - p(y', z|x) \). Such counterfactual reasoning will be used in our analysis in §5.2.

4.2. Theoretical properties of CALM

Now we revisit the axioms for attribution methods that CAM fails to fulfill (§3.1). Implementation-invariance axiom is satisfied by CALM because the attribution map \( s := p(\hat{y}, z|x) \) is a mathematical object in the probabilistic graphical model. CALM attribution also does not depend on the fragile logit values. The completeness axiom trivially follows from CALM because the final prediction \( p(y|x) \) is the sum of attribution values \( p(y, z|x) \) over \( z \). Likewise, the sensitivity axiom follows trivially from the fact that \( p(y, z|x) \) is non-zero if and only if it contributes towards the sum \( p(y|x) = \sum_{z} p(y, z|x) \).

The superior interpretability of CALM comes with a cost to pay. It alters the formulation of the usual structure for CNN classifiers where the loss function has the structure “\( \text{NLL} \circ \text{SoftMax} \circ \text{Pool} \)” on the feature map \( f \) into the one with the structure “\( \text{Pool} \circ \text{NLL} \circ \text{SoftMax} \)” on \( f \). Compared to the former, CALM gain additional interpretability by making the last layer of the network as simple as a sum over the pixel-wise experts \( p(y, z|x) \). The reduced complexity in turn increases the representational burden for the earlier layers \( f(x) \) and induces a drop in the classification accuracies (§5.3).

The interpretability-performance trade-off is unavoidable \([34]\). Therefore, it benefits users to provide a diverse array of models with different degrees of interpretability and performance \([49]\). Our work contributes to this diversity of the ecosystem of models.

5. Experiments

We present experimental results for CALM. We present two experimental analyses on attribution qualities: evalua-
denote the difference between the maps for classes A and B: \( s^A - s^B \). Red: positive values. Blue: negative values.

Figure 4. Qualitative results on CUB. We compare the counterfactual attributions from CALM and baseline methods against the GT attribution mask. The GT mask indicates the bird parts where the attributes for the class pair (A,B) differ. The counterfactual attributions denote the difference between the maps for classes A and B: \( s^A - s^B \). Red: positive values. Blue: negative values.

Our experiments are built on three real-world image classification datasets (§5.3). We then show results on the weakly-supervised object localization task in §5.4. Naver Smart Machine Learning (NSML) platform [29] has been used in the experiments.

5.1. Implementation details and baselines

**Backbone.** CALM is backbone-agnostic, as long as it is fully convolutional. We use the ResNet50 as the feature extractor unless stated otherwise. As discussed in §3, we move the final linear layer before the global average pooling layer as a convolutional layer with \( 1 \times 1 \) kernels.

**Datasets.** Our experiments are built on three real-world image classification datasets: CUB-200-2011 [62], a subset of OpenImages [8], and ImageNet1K [50]. CUB is a fine-grained bird classification dataset with 200 bird classes. We use a subset of OpenImages curated by [12] that consists of 100 coarse-grained everyday objects. ImageNet1K has 1000 classes with mixed granularity, ranging from 116 fine-grained dog species to coarse-grained objects and concepts.

**Pretraining.** We use the ImageNet pre-trained weights for f unless stated otherwise. The two convolutional layers for computing \( p(y, z|x) \) in Figure 2a are trained from scratch.

**Attribution maps.** The attribution maps for CAM and CALM are scaled up to the original image size via bilinear interpolation. For gradient-based baseline attribution methods, we apply Gaussian blurring and min-max normalization, following [12].

**Other training details** are in the Supplementary Materials.

5.2. Cue localization results

The difficulty of attribution evaluation comes from the fact that it is difficult to obtain the ground truth cue locations \( \hat{z} \). We propose a way to estimate the true cue location using the rich attribute and part annotations on the bird images in CUB-200-2011 [62].

| #part differences | #class pairs | 1 | 2 | 3 | mean |
|-------------------|--------------|---|---|---|------|
| Vanilla Gradient [55] | 30.4 | 33.3 | 36.3 | 34.3 |
| Integrated Gradient [59] | 23.6 | 26.7 | 28.8 | 27.3 |
| Smooth Gradient [56] | 24.1 | 28.3 | 32.2 | 29.6 |
| Variance Gradient [3] | 16.7 | 21.1 | 23.1 | 21.4 |
| CAM [68] | 11.8 | 15.5 | 18.6 | 16.5 |
| CALM (Ours) | 10.0 | 13.7 | 15.3 | 13.9 |
| CALMML (Ours) | 12.0 | 15.1 | 17.3 | 15.7 |
| CALM (Ours) | 24.1 | 28.3 | 32.2 | 29.6 |
| CALM (Ours) | 11.8 | 15.5 | 18.6 | 16.5 |
| CAM [68] | 16.7 | 21.1 | 23.1 | 21.4 |
| Vanilla Gradient [55] | 10.0 | 13.7 | 15.3 | 13.9 |
| Integrated Gradient [59] | 12.0 | 15.1 | 17.3 | 15.7 |
| Smooth Gradient [56] | 11.8 | 15.5 | 18.6 | 16.5 |
| Variance Gradient [3] | 16.7 | 21.1 | 23.1 | 21.4 |

Table 1. Attribution evaluation on CUB. We use the estimated GT attribution masks (§5.2) to measure the performances of attribution methods. Mean pixel-wise average precision (mPxAP) values are reported. See Figure 4 for the setup and examples.

**Estimating GT cue locations.** We generate the ground-truth cue locations using following intuition: for two classes A and B differing only in one attribute \( a \), the location \( z \) for the cue for predicting A instead of B will correspond to the object part containing the attribute \( a \). We explain algorithmically how we build the ground-truth attribution mask for an image \( x \) with respect to two bird classes A and B in CUB. We first use the attribute annotations for 312 attributes in CUB to compute the set of attributes \( S^A \) and \( S^B \). For example, \( S^A \) consists of \{black crown, black wing, all-purpose bill-shape, · · · \}. We then compute the symmetric difference of the attributes for the two classes \( S^A \triangle S^B = (S^A \cup S^B) \setminus (S^A \cap S^B) \). Now, we map each attribute \( a \) to \( S^A \triangle S^B \) to the corresponding bird part \( p \in P \) among 7 bird parts annotated in CUB. For example, the attribute-mismatching bird parts for classes “Fish crow” and “Brandt cormorant” are \( P^{A,B} = \{\text{head, wing}\} \). We locate the parts \( p^{A,B} \) in samples \( x \) of classes A and B using the keypoint annotations in CUB: \( K^{A,B}(x) \). We expand the keypoint annotations to a binary mask \( M^{A,B}(x) \) using the nearest-neighbor assignment of pixels to bird parts. The final mask \( M^{A,B}(x) \) for the input \( x \) is used as the ground-truth attribution map. See the “GT mask” column in Figure 4 for example binary
masks. For evaluation we use all class pairs in CUB with the number of attribute-differing parts $|P_{A,B}| \leq 3$, resulting in $31 + 64 + 96 = 191$ class pairs.

**Counterfactual attributions.** To predict the difference in needed cues for recognizing classes A and B, we obtain the absolute values of counterfactual attributions from each method by computing the difference $|s^A - s^B| \in [0, 1]^{H \times W}$. The underlying assumption is that $s^A$ and $s^B$ point to cues corresponding to the attributes for A and B, respectively. Hence, by taking the difference, one removes the attributions on regions that are important for both A and B.

**Evaluation metric: mean pixel-wise AP.** To measure how well attribution maps retrieve the ground-truth part pixels $M_{A,B}(x)$, we measure the average precision for the pixel retrieval task [1, 12]. Given a threshold $\tau \in [0, 1]$, we define the positive predictions as the set of pixels in over multiple images: $\{(n, h, w) \mid |s^A_{hw}(x_n) - s^B_{hw}(x_n)| \geq \tau\}$ for images $x_n$ from classes A and B. With the pixel-wise binary labels $M^A_{A,B}(x_n)$, we compute the pixel-wise average precision (PxAP) for the class pair (A, B) by computing the area under the precision-recall curve. We then take the mean of PxAP over all the class pairs of interest (e.g. those with $|P_{A,B}| = 1$) to compute the mPxAP.

**Qualitative results.** See Figure 4 for the qualitative examples of CALM and baselines including CAM. We observe that the counterfactual attribution maps $s^A - s^B$ generated by CALM ML and CALM EM are more accurate than CAM and gradient-based attribution methods; CALM ML attributions are qualitatively more precise than CALM EM. CALM tends to assign close-to-zero attributions on irrelevant regions, while the baseline methods tend to produce noisy attributions. The sparsity of CALM makes it qualitatively more interpretable than the baselines.

**Quantitative results.** Table 1 shows the mPxAP scores for CALM and baseline methods for retrieving relevant pixels as attribution regions. We examine CUB class pairs with the number of parts with attribute differences $|P_{A,B}| \in \{1, 2, 3\}$. We observe that CALM EM outperforms the baselines in all three sets of class pairs, confirming the qualitative superiority of CALM ML in Figure 4. CALM EM attains 4.7% better mPxAP than CAM on average over the three sets. CALM ML tends to be sub-optimal, compared to CALM EM (27.3% vs 34.3% mPxAP). The variants of gradients perform below a mere 20% mPxAP on average. In conclusion, the counterfactual attribution by CALM generates precise localization of the important bird parts that matter for the recognition task.

**Figure 5. Remove-and-classify results.** Classification accuracies of CNNs when $k\%$ of pixels are erased according to the attribution values $s_{hw}$. We show the relative accuracies $R_k$ against the random-erasing baseline. Lower is better.

**5.3. Remove-and-classify results**

One of the most widely used frameworks for evaluating the attribution task is the remove-and-classify evaluation [51, 30, 43, 46, 26]. Image pixels $x_{hw}$ are erased in descending order of importance dictated by the attribution values $s_{hw}$. We write $x_{-k}$ for the image where the top-$k\%$ important pixels are removed. We use the meaningful perturbation of the “blur” type [20] for erasing the pixels. A good attribution method shall assign high attribution values on important pixels; erasing them will quickly drop the classification accuracy $A_k$ with increasing $k$. We set the base reference accuracy $A_k^*$ as the classifier’s accuracy with $k\%$ of the pixels erased at random. For each method, we report the relative accuracy $R_k = A_k / A_k^*$ for different $k$.

**Results.** We show the remove-and-classify results in Figure 5 for three image classification datasets. We observe that CALM variants show the lowest relative accuracies (lower is better) $R_k$ on cue-removed images in CUB and OpenImages, compared to CAM and other baselines. For the two datasets, CALM EM attains values even close to zero at $k \in [10, 50]$. On ImageNet, CALM EM outperforms the baselines with a smaller margin. Overall, CALM EM selects the important pixels for recognition best.

**Classification performances.** We study the trade-off between interpretability and performance. The improved attribution performances come at the cost of decreased classification accuracies. Our models will be useful in applications that require great attribution performances at a small cost in model accuracies.

**5.4. Weakly-supervised object localization (WSOL)**

WSOL is related to but different from the attribution task. For WSOL, one learns to detect object foreground re-
We apply CALM to WSOL. Since attribution maps \( p(\hat{y}, z|x) \) only point to sub-parts relevant for recognition, we aggregate the attributions them over multiple classes \( p(y, z|x) = \sum_{y \in \mathcal{Y}} p(y, z|x) \) (subset attribution in §4.1.3) to fully cover the foreground regions.

### Setting the superset \( \mathcal{Y} \)

For the ground-truth class \( \hat{y} \), we set the superset \( \mathcal{Y} \) as the set of classes sharing the same part composition as \( \hat{y} \). The intuition is that the attributions are mostly on object parts and that classes of such \( \mathcal{Y} \) have attributions spread across different object parts. For example, all bird classes in CUB [62] shall share the same superset \( \mathcal{Y} = \{ \text{all 200 birds} \} \), as they share the same body part composition. On the other extreme, 100 classes in OpenImages [8, 12] do not share part structures across classes. Thus, we always set \( \mathcal{Y} = \{ \hat{y} \} \). ImageNet1K [50] is mixed. Its 1000 classes include 116 dog species, but also many other objects and concepts that do not share the same part structure. For ImageNet, we have manually annotated the superset \( \mathcal{Y} \) for every class \( \hat{y} \), using the WordNet hierarchy [38]. Details in the Supplementary Materials.

### Results

We evaluate WSOL performances based on the benchmarks and evaluation metrics in [12]. The benchmark considers 3 architectures (VGG, Inception, ResNet) and 3 datasets (CUB, OpenImages, ImageNet). Implementation details are in Supplementary Materials. We show results in Table 3. We observe that the aggregation significantly enhances the WSOL performances for \( \text{CALM}_{\text{EM}} \): 52.5% to 65.4% on CUB. \( \text{CALM}_{\text{EM}} + \mathcal{Y} \) attains the best performances on CUB and OpenImages and second-best on ImageNet.

### Analysis

We study the 116 fine-grained dog species in ImageNet more closely. We show the aggregation of attribution maps in Figure 6. CALM for \( \hat{y} \) fails to cover the full extent of the object. As the maps are aggregated over all dog species \( \mathcal{Y} \), the map precisely covers the full extents of the dogs. However, if \( \mathcal{Y} \) covers all 1000 classes, the resulting saliency map \( p(z|x) \) starts to include non-dog pixels.

### 6. Conclusion

Despite its great contributions to the field, the class activation mapping (CAM) is not as interpretable as it could be. It lacks communicability in practice and fails to meet key theoretical requirements for feature attribution methods. This paper has introduced a novel visual feature attribution method, class activation latent mapping (CALM). Based on the probabilistic treatment of the last layers of CNNs, CALM is interpretable by design. CALM satisfies the theoretical requirements as an attribution method and outperforms CAM and other baselines on attribution tasks.

### Acknowledgement

This work has been partially funded by the ERC (853489 - DEXIM) and by the DFG (2064/1 – Project number 390727645). We thank members of NAVER AI Lab, especially Sanghyuk Chun for their helpful feedback and discussion. Kay Choi designed Figure 2.
References

[1] Radhakrishna Achanta, Sheila Hemami, Francisco Estrada, and Sabine Stuckensch.
Frequency-tuned salient region detection. In CVPR, 2009. 7

[2] Julius Adelayo, Justin Gilmer, Michael Mueh, Ian Goodfellow, Moritz Hardt, and Been Kim.
Sanity checks for saliency maps. In NeurIPS, 2018. 3

[3] Christa Agarwal and Sara Hooker.
Estimating example difficulty using variance of gradients. ICMLW, 2020. 2, 6

[4] David Alvarez-Melis and Tommi S Jaakkola.
Towards robust interpretability with self-explaining neural networks.
In NeurIPS, 2018. 2

[5] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek.
On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation.
PloS one, 2015. 1, 2, 3

[6] Wonho Bae, Junhyug Noh, and Gunhee Kim.
Rethinking class activation mapping for weakly supervised object localization. In ECCV, 2020. 1, 2

[7] David Bachs, Timon Schroeter, Stefan Harmeling, MortaKawanabe, Katja Hansen, and Klaus-Robert Müller.
How to explain individual classification decisions. JMLR, 2010. 2

[8] Rodrigo Benenson, Stefan Popov, and Vittorino Ferrari.
Large-scale interactive object segmentation with human annotators. In CVPR, 2019. 6, 8

[9] Aditya Chattopadhay, Amirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian.
Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In WACV, 2018. 1, 2

[10] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnette, Cynthia Rudin, and Jonathan K Su.
This looks like that: Deep learning for interpretable image recognition. In NeurIPS, 2019. 2

[11] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille.
DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. PAMI, 2017. 5

[12] Junsuk Choe, Seong Joon Oh, Sanghyuk Chun, Zeynep Akata, and Hyunjung Shim.
Evaluation for weakly supervised object localization: Protocol, metrics, and datasets.
arXiv preprint arXiv:2007.04178, 2020. 2, 3, 6, 7, 8

[13] Junsuk Choe, Seong Joon Oh, Seungho Lee, Sanghyuk Chun, Zeynep Akata, and Hyunjung Shim.
Evaluating weakly supervised object localization methods right.
In CVPR, 2020. 2, 3, 12, 13

[14] Junsuk Choe and Hyunjung Shim.
Attention-based dropout layer for weakly supervised object localization. In CVPR, 2019. 8, 13, 14

[15] Arthur P Dempster, Nan M Laird, and Donald B Rubin.
Maximum likelihood from incomplete data via the em algorithm. JRSS: Series B, 1977. 4

[16] Thibaut Durand, Taylor Mordan, Nicolas Thome, and Matthieu Cord.
Wildcat: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation. In CVPR, 2017. 2

[17] Thibaut Durand, Nicolas Thome, and Matthieu Cord.
Wild: Weakly supervised learning of deep convolutional neural networks. In CVPR, 2016. 2

[18] Daniel C Elton.
Self-explaining ai as an alternative to interpretability methods. In AGI, 2020. 2

[19] Ruth Fong, Mandela Patrick, and Andrea Vedaldi.
Understanding deep networks via extremal perturbations and smooth masks. In ICCV, 2019. 2

[20] Ruth C Fong and Andrea Vedaldi.
Interpretable explanations of black boxes by meaningful perturbation. In ICCV, 2017. 1, 2, 7

[21] Nicholas Frosst and Geoffrey Hinton.
Distilling a neural network into a soft decision tree. AIHW, 2017. 2

[22] Ruigang Fu, Q. Hu, X. Dong, Yulan Guo, Y. Gao, and Biao Li.
Axiom-based grad-cam: Towards accurate visualization and explanation of cnns. ArXiv, abs/2008.02312, 2020. 1, 3

[23] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee.
Counterfactual visual explanations. In ICML, 2019. 1, 2

[24] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi.
A survey of methods for explaining black box models. CSUR, 2019. 1

[25] David Gunning.
Explainable artificial intelligence (xai). DARPA, 2017. 1

[26] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim.
A benchmark for interpretability methods in deep neural networks. In NeurIPS, 2019. 2, 7

[27] Minsong Kong, YoungjuGuh, Wonyoung Lee, and Hyeran Byun.
In-sample contrastive learning and consistent attention for weakly supervised object localization. In ACCV, 2020. 8, 13, 14

[28] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres.
Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav).
In ICML, 2018. 1, 2

[29] Hanjoo Kim, Minkyu Kim, Dongjoo Seo, Hyungseok Park, Sowoon Park, Hyunwooo Jo, KyungHyun Kim, Youn-gil Yang, Youngkwan Kim, et al.
Nsmf: Meet the mlaas platform with a real-world case study.
arXiv preprint arXiv:1810.09957, 2018. 6

[30] Pieter-Jan Kindermans, Kristof T. Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne.
Learning how to explain neural networks via extremal perturbations and pattern attribution.
In ICLR, 2018. 2, 7

[31] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang.
Concept bottleneck models.
In ICML, 2020. 2

[32] Krishna Kumar Singh and Yong Jae Lee.
Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In ICCV, 2017. 8, 13, 14

[33] Isaac Lage, Andrew Ross, Samuel J Gershman, Been Kim, and Finale Doshi-Velez.
Human-in-the-loop interpretability beyond feature attribution of black boxes by meaningful perturbation. In NeurIPS, 2018. 2

[34] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis.
Explainable ai: A review of machine learning interpretability methods. Entropy, 2021. 4, 5
Supplementary Materials

Supplementary Materials contain supporting claims, derivations for formulae, and auxiliary experimental results for the materials in the main paper. The sections are composed sequentially with respect to the contents of the main paper.

A. Equivalence of CAM formulations

In §3 of the main paper, we have argued that the formulation in Equation 1 copied below,

\[
p(y|x) = \text{softmax} \left( \sum_{h,w} f_{yhw}(x) \right) \tag{8}
\]

is equivalent to CNNs with an additional linear layer \( W \in \mathbb{R}^{C \times L} \) after the global average pooling (e.g. ResNet):

\[
p(y|x) = \text{softmax} \left( \sum_{l} W_{yl} \left( \sum_{h,w} \tilde{f}_{lhw}(x) \right) \right) \tag{9}
\]

where \( \tilde{f} \) is a fully-convolutional network with output dimensionality \( \tilde{f}(x) \in \mathbb{R}^{L \times H \times C} \). This follows from the distributive property of sums and multiplications:

\[
\sum_{l} W_{yl} \left( \sum_{h,w} \tilde{f}_{lhw}(x) \right) = \frac{1}{HW} \sum_{h,w} \sum_{l} W_{yl} \tilde{f}_{lhw}(x) \tag{10}
\]

where we may re-define \( f_{yhw} := \sum_{l} W_{yl} \tilde{f}_{lhw} \) as another fully-convolutional network with a convolutional layer with \( 1 \times 1 \) kernels \( (W_{yl}) \) at the end.

For networks of the form in Equation 9, the original CAM algorithm computes the attribution map by first taking the sum

\[
f_{hw} = \sum_{l} W_{yl} \tilde{f}_{lhw}(x) \tag{11}
\]

and normalizing \( f \) as in Equation 2 in main paper:

\[
s = \begin{cases} 
(f_{\max}^{-1}) \max(0, f_{\tilde{y}}) & \text{max [68]} \\
(f_{\max} - f_{\min})^{-1} (f_{\tilde{y}} - f_{\min}) & \text{min-max [53]}
\end{cases}
\]

\[
\tag{12}
\]

Hence, for both training and interpretation, the family of architectures described by Equation 1 subsumes the family originally considered in CAM [68] (Equation 9).

B. Derivation of CALM_{EM} objective

In §4.1.2, we have introduced an expectation-maximization (EM) learning framework for our latent variable model. We derive the EM objective in Equation 6 here. Our aim is to minimize the negative log-likelihood \(- \log p_\theta(y|x)\). We upper bound the objective as follows.

\[
- \log p_\theta(y|x) = - \log \sum_{z} p_\theta(y, z|x) \tag{13}
\]

\[
= - \log \sum_{z} p_\psi(z|x, y) \frac{p_\theta(y, z|x)}{p_\psi(z|x, y)} \tag{14}
\]

\[
\leq - \sum_{z} p_\psi(z|x, y) \log \frac{p_\theta(y, z|x)}{p_\psi(z|x, y)} \tag{15}
\]

\[
\leq - \sum_{z} p_\psi(z|x, y) \log p_\theta(y, z|x) . \tag{16}
\]

The inequalities leading to Equation 15 and 16 follow from the Jensen’s inequality and the positivity of the entropy, respectively.

We parametrize each term with a neural network. \( p_\theta(y, z|x) \) is computed via \( g_{yz} \cdot h_z \) and \( p_\psi(z|x, y) \) is first decomposed as

\[
p_\theta(z|x, y) = \frac{p_\psi(y, z|x)}{\sum p_\psi(y, l|x)} \tag{17}
\]

and computed with neural networks

\[
p_\psi(z|x, y) = \frac{g'_{yz} \cdot h'_z}{\sum_l g'_{yl} \cdot h'_l} \tag{18}
\]

where \( g' \) and \( h' \) are neural networks parametrized with \( \theta' \).

C. Training details for CALM

We provide miscellaneous training details for CALM. See §5.1 in the main paper for major training details.

Architecture. We use the ResNet50 as the feature extractor. We enlarge the attribution map size to \( 28 \times 28 \) by changing the stride of the last two residual blocks from 2 to 1. Two CNN branches \( g \) and \( h \) are followed by the feature extractor. The branch \( g \) is the one convolutional layer of kernel size 1 and stride 1 with the number of output channel to be the number of classes. The branch \( h \) is composed of one convolutional layer of kernel size 1 and stride 1 with the number of output channel to be 1, followed by the ReLU activation function.

Optimization hyperparameters. We use the stochastic gradient descent with the momentum 0.9 and weight decay \( 1 \times 10^{-4} \). We set the learning rate as \((3 \times 10^{-5}, 5 \times 10^{-5}, 7 \times 10^{-5})\) for (CUB, OpenImages, ImageNet).

D. More qualitative results

See qualitative results in Figure 7 for the comparison of attributions against the ground-truth attributions using the
counterfactual maps $s^A - s^B$. As in Figure 4 of the main paper, we observe that CALM attains the best similarity with the ground-truth masks. CALM are also the most human-understandable among the considered methods.

We add more qualitative results of the attribution maps $s^A$ for the ground-truth class for CUB (Figure 8), OpenImages (Figure 9), and ImageNet (Figure 10). For each case, we do not have the GT attribution maps as in Figure 7, but we show the GT foreground object bounding boxes, or masks if available (OpenImages). Note that the attribution maps (CALM_EM and CALM_ML) are not designed to highlight the full object extent, while the aggregated versions (+$\mathcal{Y}$) are designed so. We observe that in all three datasets, CALM_EM +$\mathcal{Y}$ and CALM_ML +$\mathcal{Y}$ tend to generate high-quality foreground masks for the object of interest; for OpenImages, note that "+$\mathcal{Y}$" does not change CALM_EM and CALM_ML, as the supersets $\mathcal{Y}$ are all singletons.

**E. Evaluation protocol for WSOL**

In §5.4 of the main paper, we have presented experimental results on WSOL benchmarks. In this section, we present metrics, datasets, and validation protocols for the WSOL experiments. We follow the recently proposed WSOL evaluation protocol [13].

**Metrics.** After computing the attribution $s$, WSOL methods binarize the attribution by a threshold $\tau$ to generate a foreground mask $M = 1[s_{ij} \geq \tau]$. When there are mask annotations in the dataset, we compute pixel-wise precision and recall at the threshold $\tau$. The $P_{x}A_{P}$ is the area under the precision and recall curve at all possible thresholds $\tau \in [0, 1]$. On the other hand, when only the box annotations are available, we generate a bounding box that tightly bound each connected component on the mask $M$. Then, we calculate intersection over union (IoU) between all pairs of ground-truth boxes and predicted boxes at all thresholds $\tau \in [0, 1]$. When there is at least one pair of (ground-truth, prediction, $\tau$) with $IoU \geq \delta$, we regard the localization prediction of the attribution as correct. The $MaxBoxAccV2$ is the average of three ratios of correct images in the dataset at three $\delta = 0.3, 0.5, 0.7$.

**Evaluation protocol.** Every dataset in our WSOL experiments consists of three disjoint splits: train, val, and test. The val and test splits contain images with localization and class labels, while the train split contains images only with class labels. We use the train split to train the classifier, and tune our hyperparameter by checking the localization performance on the val split. Specifically, we only tune the learning rate by randomly sampling 30 learning rates from the log-uniform distribution $LogUniform[5^{-5},5^{-1}]$. Then, based on the performance on val, we select the optimal learning rate. Finally, we measure the final localization performance on test split with the selected learning rate. Since previous WSOL methods in [13] are also evaluated under this evaluation protocol, we can compare fairly our method with the previous methods.

**F. Superset $\mathcal{Y}$ for ImageNet classes**

In the main paper §5.4, we have discussed the disjoint goals pursued by two tasks, visual attribution and WSOL. The former aims to locate cues that make the class distinguished from the others and typically locates a small part of the object; the latter aims to locate the foreground pixels for objects. To bridge the two, we have considered an aggregation strategy to produce a foreground mask from attribution maps. Our assumption is that attribution maps for different classes for the same family sharing the identical object structure will cover various object parts in the foreground mask. For example, 200 bird classes in CUB share the part structure of “head-beak-breast-wing-belly-leg-tail”, but each class will highlight different parts. Combining the attribution maps for the 200 classes will roughly cover all the object parts, providing a higher-quality foreground mask.

For 1000 classes in ImageNet1K, we manually annotate the corresponding supersets $\mathcal{Y}$. We first build a tree of concepts over the classes using the WordNet hierarchy [38]. The leaf nodes correspond to 1000 classes, while the root node corresponds to the concept “entity” that includes all 1000 classes. For each leaf node (ImageNet class) $y$, we have annotated the superset $\mathcal{Y}$ by choosing an appropriate parent node of $y$. The parent selection criteria is as follows:

- Choose the parent node $PA(y)$ of $y$ as close to the root as possible;
- Such that $\mathcal{Y}$, the set of all children of $PA(y)$, consists of classes with the same object structure as $y$.

**Algorithm.** We efficiently annotate the parent nodes by traversing the tree in a breadth-first-search (BFS) manner from the root node, “entity”. We start from the direct children (depth= 1) of the root node. For each node of depth 1, we mark if the concept contains classes of the same object structure (hom or het). For example, the family of classes under the organism parent is not yet specific enough to contain classes of homogeneous object structures, so we mark het. We continue traversing in depths 2, 3, and so on. If a node at depth $d$ is marked hom, we treat all of its children to have the same superset $\mathcal{Y}$ and do not traverse its descendants for depth > $d$. For example, the canine superset of 116 ImageNet classes is reached by following the genealogy of organism $\rightarrow$ animal $\rightarrow$ chordate $\rightarrow$ mammal $\rightarrow$ placental $\rightarrow$ carnivore $\rightarrow$ canine. We note that the task is fairly well-defined for humans.

**Results.** We find 450 supersets in ImageNet1K. 120 of them are non-singleton, consisting of at least two ImageNet
classes. The rest 330 classes are singleton supersets. See Table 4 for the list.

G. A complete version of WSOL results

See Table 5 for the complete version of Table 3 of the main paper. We show the architecture-wise performances for CALM, CAM, and six previous WSOL methods (HaS [32], ACoL [66], SPG [67], ADL [14], CutMix [64], InCA [27]).

In the ImageNet1K dataset, the proposed method achieves competitive performance (62.8%) with CAM (62.4%) and InCA (63.1%). The superset $\mathcal{Y}$ aggregation improves the localization performances for $\text{CALM}_{\text{EM}}$ (62.5% → 62.8%), but it decreases the CAM performances significantly (62.4% → 60.6%). For CUB, we observe that $\text{CALM}_{\text{EM}}$ does not localize the birds effectively without the superset $\mathcal{Y}$ (52.5%). With the superset aggregation, $\text{CALM}_{\text{EM}}$ attains 65.4%. This is expected behavior because $\text{CALM}_{\text{EM}}$ attributions often highlight small discriminative object parts (e.g. Figure 4 in the main paper). For OpenImages, $\text{CALM}_{\text{EM}}$ achieves the state-of-the-art performances on all three backbone architectures (61.3%, 64.4%, 62.5%). Since the modified OpenImages [13] consists of classes of unique object structures, the supersets are all singletons. The performances are thus identical with or without the aggregation $+\mathcal{Y}$. In summary, $\text{CALM}_{\text{EM}}$ on ImageNet is competitive, compared to the state of the art, and is the new state of the art on CUB and OpenImages.

H. More qualitative examples for WSOL

In Figure 6 of the main paper, we have shown the aggregation of attribution maps at different depths of hierarchy for the “canine” class. We show additional qualitative examples of the superset aggregation for other classes in Figure 11. We note that the optimal depths that precisely cover the object extents differ across classes.
Table 4. List of supersets \( \mathcal{Y}_{\text{line}} \). We list 120 supersets for classes in ImageNet1K. We omit the rest 330 supersets from the list, as they only have single elements (Singletons).

| Class name | WordNet ID | # Classes | Class name | WordNet ID | # Classes | Class name | WordNet ID | # Classes |
|------------|------------|-----------|------------|------------|-----------|------------|------------|-----------|
| canine     | n02083346  | 116       | wheel      | n04574999  | 4         | power tool | n03997484  | 2         |
| bird       | n01503061  | 52        | swine      | n02395003  | 3         | farm machine | n03322940  | 2         |
| reptile    | n01661091  | 36        | lagomorph  | n02323449  | 3         | slot machine | n04243941  | 2         |
| insect     | n02159955  | 27        | marsupial  | n01874343  | 3         | free-reed instrument | n03933242  | 2         |
| primate    | n02469934  | 20        | coelenterate | n01903422  | 3         | piano       | n03928116  | 2         |
| ungulate   | n02370806  | 17        | echinoderm | n02316707  | 3         | lock        | n03682487  | 2         |
| aquatic vertebrate | n01473806 | 16  | person | n00007846  | 3         | breathing device | n02895606  | 2         |
| building   | n02913152  | 12        | firearm    | n03343853  | 3         | heater      | n03508102  | 2         |
| car        | n02958343  | 10        | clock      | n03046257  | 3         | locomotive  | n03684823  | 2         |
| boid       | n02401031  | 9         | portable computer | n03985232  | 3         | bicycle     | n02834778  | 2         |
| arachnid   | n01769347  | 9         | brass      | n02891788  | 3         | railcar     | n02959942  | 2         |
| ball       | n02778669  | 7         | cart       | n02970849  | 3         | handcart    | n03480883  | 2         |
| headaddress | n03502509  | 9        | sailing vessel | n04128837  | 3         | warship     | n04552696  | 2         |
| feline     | n02120997  | 8         | aircraft   | n02686568  | 3         | sled        | n04235291  | 2         |
| amphibian  | n01627424  | 8         | bus        | n02924116  | 3         | reservoir   | n04078578  | 2         |
| decapod crustacean | n01976146 | 7       | pot        | n03990474  | 3         | jar         | n03593526  | 2         |
| place of business | n03953020 | 8       | dish       | n03206908  | 3         | basket      | n02801938  | 2         |
| mustelid mammal | n02441326 | 7       | pot        | n03990474  | 3         | glass       | n03438257  | 2         |
| fungus     | n12992868  | 7         | pen        | n03906997  | 3         | shaker      | n04183329  | 2         |
| truck      | n04490091  | 7         | telephone, phone | n04401088  | 3         | opener       | n03848348  | 2         |
| bottle     | n02876657  | 7         | gymnastic apparatus | n03472232  | 3         | power tool  | n03979484  | 2         |
| seat       | n04161981  | 7         | neckwear   | n03816005  | 3         | pan, cooking pan | n03880531  | 2         |
| rodent     | n02329401  | 6         | swimsuit   | n04371563  | 3         | cleaning implement | n03039947  | 2         |
| mollusk    | n01940736  | 6         | body armor | n02862048  | 3         | puzzle      | n04028315  | 2         |
| stringed instrument | n04338517 | 6       | footwear   | n03381126  | 3         | camera      | n02942699  | 2         |
| boat       | n02858304  | 6         | bridge     | n02898711  | 3         | weight      | n04571292  | 2         |
| box        | n02883344  | 6         | memorial   | n03743902  | 3         | cabinet     | n02933112  | 2         |
| toiletry   | n04447443  | 6         | alcohol    | n07884567  | 3         | curtain     | n03151077  | 2         |
| bag        | n02773037  | 5         | dessert    | n07690840  | 3         | sweater     | n04370048  | 2         |
| stick      | n04317420  | 5         | cruciferous vegetable | n07713395  | 3         | robe        | n04097866  | 2         |
| bear       | n02131653  | 4         | baby bed   | n02766320  | 3         | scarf       | n04148397  | 2         |
| woodwind   | n04598582  | 4         | procyonid  | n02507649  | 2         | gown        | n03450516  | 2         |
| source of illumination | n04263760 | 4       | viverrine  | n02134971  | 2         | protective garment | n04015204  | 2         |
| ship       | n04194289  | 4         | cetacean   | n02062430  | 2         | oven        | n03862676  | 2         |
| edge tool  | n03265032  | 4         | edentate   | n02453611  | 2         | sheath      | n04187061  | 2         |
| overgarment | n03863923  | 4         | elephant   | n02503517  | 2         | movable barrier | n03795580  | 2         |
| skirt      | n04208308  | 4         | prototherian | n01871543  | 2         | sheet       | n04186643  | 2         |
| roof       | n04105068  | 4         | worm       | n01922303  | 2         | plaything   | n03964744  | 2         |
| fence      | n03327234  | 4         | flower     | n11669261  | 2         | mountain    | n03953803  | 2         |
| piece of cloth | n03932670  | 4       | timer      | n04438304  | 2         | shore       | n04933442  | 2         |

Table 5. WSOL results on CUB, OpenImages, and ImageNet. Extension of Table 2 in main paper. CALM\_EM and CALM\_EM +\( \mathcal{Y} \)' are compared against the baseline methods. CALM\_EM +\( \mathcal{Y} \)' denote the aggregated attribution map for classes in \( \mathcal{Y} \).
Figure 7. **Counterfactual attribution maps on CUB.** Extension of Figure 4 in main paper. We compare the counterfactual attributions from CALM and baseline methods against the GT attribution mask. The GT mask indicates the bird parts where the attributes for the class pair (A,B) differ. The counterfactual attributions denote the difference between the maps for classes A and B: $s^A - s^B$. Red: positive values. Blue: negative values.
| Image | CALM_{EM} (Ours) | CALM_{EM} + \hat{y} (Ours) | CALM_{MC} (Ours) | CALM_{MC} + \hat{y} (Ours) | CAM | Vanilla Gradient | Variance Gradient |
|-------|------------------|-----------------------------|------------------|-----------------------------|-----|------------------|------------------|
| ![Image](image1.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![CAM](cam.png) | Vanilla Gradient | Variance Gradient |
| ![Image](image2.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![CAM](cam.png) | Vanilla Gradient | Variance Gradient |
| ![Image](image3.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![CAM](cam.png) | Vanilla Gradient | Variance Gradient |
| ![Image](image4.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![CAM](cam.png) | Vanilla Gradient | Variance Gradient |
| ![Image](image5.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![CAM](cam.png) | Vanilla Gradient | Variance Gradient |

Figure 8. **Examples of attribution maps on CUB.** We show the object bounding boxes to mark the foreground regions. \( + \hat{y} \) denotes the class aggregation.
Figure 9. **Examples of attribution maps on OpenImages.** We show the object bounding boxes to mark the foreground regions. We do not show the class aggregation (+Y) because it does not change our methods (CALM EM and CALM ML) on OpenImages (Y are all singletons).
| Image | CALM (Ours) | CALM + \( \mathcal{Y} \) (Ours) | CALM \( \mathcal{M} \) (Ours) | CALM \( \mathcal{M} \) + \( \mathcal{Y} \) (Ours) | CAM | Vanilla Gradient | Variance Gradient |
|-------|-------------|-------------------------------|----------------|---------------------------------|-----|----------------|------------------|
| ![Image](image1.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![Map5](map5.png) | ![Map6](map6.png) | ![Map7](map7.png) |
| ![Image](image2.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![Map5](map5.png) | ![Map6](map6.png) | ![Map7](map7.png) |
| ![Image](image3.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![Map5](map5.png) | ![Map6](map6.png) | ![Map7](map7.png) |
| ![Image](image4.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![Map5](map5.png) | ![Map6](map6.png) | ![Map7](map7.png) |
| ![Image](image5.png) | ![Map1](map1.png) | ![Map2](map2.png) | ![Map3](map3.png) | ![Map4](map4.png) | ![Map5](map5.png) | ![Map6](map6.png) | ![Map7](map7.png) |

Figure 10. **Examples of attribution maps on ImageNet.** We show the object bounding boxes to mark the foreground regions. \(+\mathcal{Y}\) denotes the class aggregation.
Figure 11. **Examples for aggregation at different hierarchies on ImageNet.** Extension of Figure 6 of main paper. We show the aggregated attribution maps at different depths of the hierarchy and the correspondingly expanding superset $\mathcal{Y}$. 