# PhaseWin Search Framework Enable Efficient Object-Level Interpretation (*Supplementary Materials*)

Zihan Gu[1,2], Ruoyu Chen[1,2], Junchi Zhang[3], Yue Hu[1,2], Hua Zhang[1,2,*], Xiaochun Cao[4]

[1]Institute of Information Engineering, Chinese Academy of Sciences

[2]School of Cyber Security, University of Chinese Academy of Sciences [3]Shanghai Center for Mathematical Sciences, Fudan University

[4]School of Cyber Science and Technology, Sun Yat-sen University

{guzihan,chenruoyu,huyue,zhanghua}@iie.ac.cn caoxiaochun@mail.sysu.edu.cn

## A. Window Selection Polices

In this section, we first introduce the four algorithms (described in Table 1) we can choose from for the sub-process designed for a window of the phasewin algorithm.

First, the most basic approach is to apply greedy search within the window, which is also the slowest. Our three subsequent designs all use the submodularity assumption to varying degrees to reduce the number of searches within the window. $\pi_{BA}$ uses an adaptive scaling search strategy, $\pi_{T_2}$ considers two elements with the smallest reduction in combined return, and $\pi_{BAF}$ reduces the number of comparisons by maintaining an upper bound list.

Table 1. Window selection policies $\pi(\cdot)$ used within the `WindowSelection` subroutine.

| Policy | Description |
|---|---|
| $\pi_{\mathrm{LG}}$ | **Local-Greedy:** Picks the top candidate if its gain exceeds $\tau_{\mathrm{sel}}$. |
| $\pi_{\mathrm{BA}}$ | **Beta-Adaptive:** Selects all candidates above an adaptive cut-off based on the window's max gain. |
| $\pi_{\mathrm{T2}}$ | **Top-2:** Jointly selects the top two candidates if their gains are high and their relative gap is small. |
| $\pi_{\mathrm{BAF-B}}$ | **Batched Best-Above w/ Forward-checking:** Processes the window in batches, using cached gains to terminate early and reduce evaluations. |

## B. Complete Algorithm Process

The algorithm operates in discrete phases. At the start of each phase, it performs a full evaluation on all remaining candidate regions ($\mathcal{R}$) and greedily selects the single best region to anchor the current search state. This ensures consistent progress. Based on the maximum marginal gain ($G_t$) observed in this step, it computes two adaptive thresholds: a selection threshold $\tau_{\mathrm{sel}}$ to identify high-potential candidates and a pruning threshold $\tau_{\mathrm{del}}$ to discard low-utility regions. This adaptive pruning strategy dynamically narrows the search space, focusing computational resources on the most promising regions.

For the initial phases (controlled by a hyperparameter $m_{\mathrm{active}}$), PhaseWin employs a sliding window of size $w$ over the sorted candidate pool $\mathcal{P}_t$. Within this window, a selection policy $\pi(\cdot)$—such as Beta-Adaptive (BA)—is applied to identify a batch of one or more regions for selection. This allows the model to select complementary regions simultaneously, a capability absent in naive greedy search. To further refine the candidate evaluation, a simulated annealing mechanism may defer the entry of lower-scoring regions into the window, allowing more promising candidates to be assessed first. After $m_{\mathrm{active}}$ phases, the algorithm transitions to a simplified greedy selection over the candidate pool to ensure convergence.

A key innovation of PhaseWin is its *dynamic phase supervision*. We monitor the sequence of marginal gains of the selected regions, $\Delta_i = \mathcal{F}(S_i) - \mathcal{F}(S_{i-1})$. If the current gain drops precipitously compared to the previous one (i.e., $\Delta_i < \theta \cdot \Delta_{i-1}$, where $\theta$ is an adaptive supervision coefficient), it signals a potential breakdown of local submodularity. In this event, the algorithm calculates a probability $p_{\mathrm{exit}}$ to terminate the current phase prematurely. This probabilistic exit prevents the algorithm from wasting evaluations on a sequence of diminishing returns and allows it to restart with a new anchor region. The complete procedure is detailed in Algorithm 1.

## C. Evaluation Metrics

**Faithfulness.** To assess how well an attribution map reflects the model's reasoning, we compute the Insertion and

*Corresponding Author

**Algorithm 1:** Phase-Window (PhaseWin) Search Algorithm

**Input:** Region set $\mathcal{V}$, target size $k$, scoring function $\mathcal{F}(\cdot, \mathbf{b}_{\text{target}}, c)$
**Hyperparameters:** Window size $w$, active window phases $m_{\text{active}}$, selection ratio $\alpha_{\text{sel}}$, deletion ratio $\beta_{\text{del}}$, supervision coefficients $\{\theta_t\}$
**Output:** Ordered set $S$

$S \leftarrow \emptyset, \quad \mathcal{R} \leftarrow \mathcal{V}, \quad t \leftarrow 0, \quad \Delta_{\text{prev}} \leftarrow \infty$
**while** $|S| < k$ **and** $\mathcal{R} \neq \emptyset$ **do**
  $t \leftarrow t + 1$
  Compute $g_r \leftarrow \mathcal{F}(S \cup \{r\})$ for all $r \in \mathcal{R}$
  **if** $\max(g_r) \leq 0$ **then**
    **break**
  **end**
  $\alpha_{\text{best}} \leftarrow \arg\max_{r \in \mathcal{R}} g_r$
  $S \leftarrow S \cup \{\alpha_{\text{best}}\}$
  $\mathcal{R} \leftarrow \mathcal{R} \setminus \{\alpha_{\text{best}}\}$
  $\Delta_t \leftarrow g_{\alpha_{\text{best}}}$
  Recompute $g_r$ for all $r \in \mathcal{R}$ and let $G_t = \max_r g_r$
  $\tau_{\text{sel}} \leftarrow \alpha_{\text{sel}} \cdot G_t$
  $\tau_{\text{del}} \leftarrow \beta_{\text{del}} \cdot G_t$
  $\mathcal{R} \leftarrow \{r \in \mathcal{R} \mid g_r \geq \tau_{\text{del}}\}$ `// Prune low-gain regions`
  $\mathcal{P}_t \leftarrow \{r \in \mathcal{R} \mid g_r \geq \tau_{\text{sel}}\} \cup \text{RandomSample}(\{r \in \mathcal{R} \mid g_r < \tau_{\text{sel}}\})$
  Sort $\mathcal{P}_t$ in descending order of $g_r$
  **if** $t \leq m_{\text{active}}$ **then**
    `// Windowing Mode`
    Initialize window $W$ with top $w$ elements from $\mathcal{P}_t$
    **while** $|W| > 0$ **and** $|S| < k$ **do**
      $A \leftarrow \pi(W, \mathcal{F}, \tau_{\text{sel}})$ `// Selection policy (e.g., BA)`
      **if** $A = \emptyset$ **then**
        **break**
      **end**
      **foreach** $\alpha \in A$ **do**
        $\Delta_i \leftarrow \mathcal{F}(S \cup \{\alpha\}) - \mathcal{F}(S)$
        **if** $\Delta_i < \theta_t \cdot \Delta_{\text{prev}}$ **then**
          Compute exit probability $p_{\text{exit}}(\Delta_i, \Delta_{\text{prev}}, \theta_t)$
          **if** $\text{rand}() < p_{\text{exit}}$ **then**
            **goto** end_phase
          **end**
        **end**
        $S \leftarrow S \cup \{\alpha\}, \quad \Delta_{\text{prev}} \leftarrow \Delta_i$
      **end**
      Remove evaluated items from $W$ and refill from $\mathcal{P}_t$
    **end**
  **end**
  **else**
    `// Degenerate Greedy Mode`
    **foreach** $\alpha \in \mathcal{P}_t$ **do**
      $\Delta_i \leftarrow \mathcal{F}(S \cup \{\alpha\}) - \mathcal{F}(S)$
      **if** $\Delta_i < \theta_t \cdot \Delta_{\text{prev}}$ **then**
        Compute exit probability $p_{\text{exit}}$
        **if** $\text{rand}() < p_{\text{exit}}$ **then**
          **break**
        **end**
      **end**
      $S \leftarrow S \cup \{\alpha\}, \quad \Delta_{\text{prev}} \leftarrow \Delta_i$
      **if** $|S| \geq k$ **then**
        **break**
      **end**
    **end**
  **end**
  `// End current phase` end_phase:
**end**
**return** $S$

Deletion AUC scores, which quantify the change in model output as the most (or least) important superpixels are progressively revealed or removed [3]. We apply these metrics both to classification confidence and to Intersection-over-Union (IoU), thus capturing the attribution's influence on recognition and localization. We further measure the highest confidence score for any predicted box with $\mathrm{IoU} > 0.5$ relative to the target. For failure cases, we evaluate the *Explaining Successful Rate (ESR)*, which measures whether a saliency map can guide the model to a correct detection for initially misclassified or low-confidence predictions.

**Localization Accuracy.** We use two established metrics: (i) the *Point Game*, which checks whether the most salient pixel lies inside the ground-truth bounding box, and (ii) the *Energy Point Game*, which extends this by considering the energy concentration of saliency around the target [4]. These metrics are evaluated only on correctly detected samples.

**Efficiency.** To provide a fair and hardware-agnostic cost measure, we introduce the *Model Evaluation Count (MEC)* as our primary efficiency metric, where one unit corresponds to a single forward pass through the model. The total MEC reflects the algorithm's runtime cost. Additionally, we define the *Accuracy–Cost Ratio (AC-Ratio)* as the primary performance metric (faithfulness score) multiplied by 1000 and divided by the MEC. This ratio is most meaningful when the faithfulness score meets a predefined quality threshold.

## D. Implementation Details

In all experiments, the ground-truth bounding box $b_{\text{target}}$ and its category $c$ are provided as references for generating attributions. Each image is segmented into 100 sparse subregions using the SLICO superpixel algorithm, which serve as the interpretable units.

For PhaseWin, we apply a window size of 16 when selecting from 50 sub-regions and 32 when selecting from 100 sub-regions. Results are averaged over five random seeds, with variance consistently below 2%.

As the scoring function is not strictly monotonic submodular, the stopping criterion is implemented in a ratio-based form:

$$\frac{S_{k-2}}{S_{k-1}} - \frac{S_{k-1}}{S_k} \leq \tau.$$

We use $\tau = 0.025$ for 50 sub-regions and $\tau = 0.01$ for 100 sub-regions. This criterion ensures numerical stability across different settings.

## E. Full Proof

In this section, we will introduce the proof of Theorem 3.1. Property 3.1 is a classic result of combinatorial optimization. If you are interested in Property 3.1, you can find the relevant proof in [1, 2].

### Proof of Theorem 3.1

*Proof.* If the parameter for AdaptiveThreshold is $(\alpha, \gamma)$ (for select and delete), the parameter for WindowSelection when $|S| = i$ is $\beta_i$ with $\beta_i$ increasing and $\alpha\beta_i \geq \gamma$.

Let $S_{\text{PhaseWin}} = (v_1, v_2, \ldots, v_k)$, $S_0 = \emptyset$ and $S_i = \{v_1, v_2, \ldots, v_i\}$. Let $\rho_u(V) = \mathcal{F}(V \cup \{u\}) - \mathcal{F}(V)$.

For each $1 \leq i \leq k$ such that $v_i$ is an element directly added into $S_{\text{PhaseWin}}$ without going into WindowSelection, let $\mathcal{R}_i$ be the set of choosable elements before $v_i$ is selected, $\mathcal{D}_i$ be the set of deleted elements after $v_i$ is selected in PartitionCandidates. Then we have

$$\begin{aligned}
&a_i \triangleq \rho_{v_i}(S_{i-1}) = \max_{j \in \mathcal{R}_i} \rho_j(S_{i-1}),\\
&\mathcal{D}_i \triangleq \{j \in \mathcal{R}_i \mid \rho_j(S_{i-1}) < \gamma a_i\},\\
&V_i \triangleq \{j \in \mathcal{R}_i \mid \rho_j(S_{i-1}) > \alpha a_i\};\\
&W_i \triangleq (e_{i,1}, \ldots, e_{i,m_i}) \subseteq V_i \text{ is the max sequence s.t.}\\
&e_{i,j} = v_{i+j}\\
&= \text{argmax}\{\rho_e(S_{i+j-1}) \mid e \in S_i \setminus \{e_{i,1}, \ldots, e_{i,j-1}\}\}\\
&\text{and}\\
&b_{i,j} \triangleq \rho_{e_{i,j}}(S_{i+j-1}) \geq \beta_{i+j} b_{i,0} \geq \alpha\beta_{i+j} a_i\\
&\qquad \geq \alpha\beta_{i+j} \max_{j \in \mathcal{R}_i} \rho_j(S_{i+j-1}).
\end{aligned}$$

Thus for any $1 \leq l \leq k$, we have

$$\rho_{v_l}(S_{l-1}) \geq \alpha\beta_l \max_{j \in \mathcal{R}_l} \rho_j(S_{l-1}).$$

Since $\mathcal{F}$ is increasing and submodular, for any $1 \leq l \leq k$ we have

$$\begin{aligned}
\mathcal{F}(S_{\text{OPT}}) &\leq \mathcal{F}(S_{l-1}) + \sum_{j \in (T \setminus S_{l-1}) \cap \mathcal{R}_l} \rho_j(S_{l-1})\\
&\quad + \sum_{m=1}^{l-1} \sum_{x \in (T \setminus S_{l-1}) \cap D_m} \rho_j(S_{l-1})\\
&\leq \mathcal{F}(S_{l-1}) + \frac{k}{\alpha\beta_l}\rho_{v_l}(S_{l-1}) + k\gamma \sum_{m=1}^{l-1} \rho_{v_m}(S_{m-1})\\
&= \frac{k}{\alpha\beta_l}\left(\mathcal{F}(S_{l-1}) + \rho_{v_l}(S_{l-1})\right)\\
&\quad - \left(\frac{k}{\alpha\beta_l} - 1 - k\gamma\right)\mathcal{F}(S_{l-1})\\
&= \frac{k}{\alpha\beta_l}\mathcal{F}(S_l) - (\frac{k}{\alpha\beta_l} - 1 - k\gamma)\mathcal{F}(S_{l-1}).
\end{aligned}$$

Let $\lambda_i = \frac{\alpha\beta_i}{k}$ and $\mu_i = \frac{\alpha\beta_i}{k}(\frac{k}{\alpha\beta_i} - 1 - k\gamma)$, then we have

$$\mathcal{F}(S_{\text{PhaseWin}}) \geq \mathcal{F}(S_{\text{OPT}}) \cdot \big(\lambda_k + \mu_k\lambda_{k-1} + \mu_k\mu_{k-1}\lambda_{k-2} + \cdots + \mu_k\mu_{k-1}\ldots\mu_2\lambda_1\big).$$

In particular, if $\beta_i = \beta$ for $i = 1, 2, \ldots, k$, then

$$\mathcal{F}(S_{\text{PhaseWin}}) \geq \frac{\lambda_1(1-\mu_1^k)}{1-\mu_1}\mathcal{F}(S_{\text{OPT}}).$$

If $k, \alpha, \beta$ is big enough and $\gamma$ is small enough, then

$$\mathcal{F}(S_{\text{PhaseWin}}) \geq \left(1 - \frac{1}{e} - o(1)\right)\mathcal{F}(S_{\text{OPT}}).$$

□

Based on our derivation, when $\beta_i$ is constant, the exact approximation ratio is given by:

$$\Gamma(k, \alpha, \beta, \gamma) = \frac{1 - (1 - \frac{\alpha\beta(1+k\gamma)}{k})^k}{1 + k\gamma}.$$

This closed-form expression allows us to quantify the $o(1)$ term precisely. Substituting the parameter settings used in our experiments, we obtain the following theoretical lower bounds:

Setting 1 ($k = 50$): With $\alpha = 0.6, \beta = 0.9, \gamma = 0.025$, the guaranteed ratio is 0.315.

Setting 2 ($k = 50$): With $\alpha = 0.7, \beta = 0.9, \gamma = 0.025$, the guaranteed ratio is 0.339.

Setting 3 ($k = 100$): With $\alpha = 0.7, \beta = 0.9, \gamma = 0.01$, the guaranteed ratio is 0.360.

These calculations demonstrate that under our experimental configurations, the algorithm theoretically guarantees roughly $31\% - 36\%$ of the optimal performance in the worst case.

***Explanation of hyperparameters:*** Figure 4 in the main paper presents ablation results for the two most influential hyperparameters, demonstrating that PhaseWin is robust across a wide range of settings. We did not tune hyperparameters to optimize performance, and variations remain within 0.03–0.04 across all tested configurations. Other hyperparameters exhibit minimal impact in our experimental setting, primarily affecting the degree of acceleration rather than attribution quality.

# F. Submodularity and Supermodularity

## F.1. Definitions

Let $V$ denote a finite ground set of candidate regions and $F : 2^V \to \mathbb{R}$ be a set function that scores any subset $S \subseteq V$.

**Definition F.1** (Submodularity)**.** A set function $F$ is *submodular* if it satisfies the *diminishing returns property*: for all $A \subseteq B \subseteq V$ and $\alpha \in V \setminus B$,

$$F(A \cup \{\alpha\}) - F(A) \quad \geq \quad F(B \cup \{\alpha\}) - F(B).$$

That is, the marginal gain of adding an element decreases as the context grows.

**Definition F.2** (Supermodularity)**.** A set function $F$ is *supermodular* if it satisfies the *increasing returns property*: for all $A \subseteq B \subseteq V$ and $\alpha \in V \setminus B$,

$$F(A \cup \{\alpha\}) - F(A) \quad \leq \quad F(B \cup \{\alpha\}) - F(B).$$

That is, the marginal gain of adding an element increases as the context grows.

## F.2. Optimization Significance

Submodularity generalizes the notion of convexity to discrete set functions. Maximizing a monotone submodular function under a cardinality constraint admits a simple greedy algorithm with a $(1 - 1/e)$-approximation guarantee, which is provably optimal under polynomial-time complexity assumptions. In contrast, supermodular functions exhibit cooperative effects, and their maximization is generally intractable, while their minimization is often easier.

## F.3. AUC Curve Properties

In attribution evaluation, we consider the insertion process: progressively adding sub-regions $s_1, s_2, \ldots$ into the image. Let

$$\text{AUC}(k) = \frac{1}{|V|}\sum_{j=1}^{k} F(\{s_1, \ldots, s_j\})$$

denote the cumulative insertion-AUC score up to step $k$.

**Theorem F.1.** *If $F$ is submodular, then the insertion AUC curve* $\text{AUC}(k)$ *is concave in $k$. If $F$ is supermodular, then* $\text{AUC}(k)$ *is convex in $k$.*

*Sketch.* For submodular $F$, diminishing returns imply that the marginal gain $F(S \cup \{s\}) - F(S)$ is non-increasing in $|S|$. Thus, the discrete derivative of $\text{AUC}(k)$ decreases with $k$, yielding concavity. Conversely, if $F$ is supermodular, marginal gains increase with $k$, so $\text{AUC}(k)$ is convex. □

## F.4. Implications for Deep Learning

Deep neural networks do not strictly satisfy either submodularity or supermodularity. Instead, their attribution behavior reflects a hybrid of both: some features exhibit redundancy (submodular-like), while others rely on synergy (supermodular-like). From the perspective of distributed computation, submodularity and supermodularity describe not universal properties of the model but rather the modes

of feature aggregation. Submodularity corresponds to robust, redundant feature usage, while supermodularity corresponds to cooperative, highly interactive feature combinations. These patterns shed light on how models internally organize basic feature units, rather than providing exact guarantees.

The two models we selected are, respectively, dominated by submodularity and supermodularity. Below, we show the Insertion AUC curves (Figure 1) for Grounding DINO and Florence-2 on the same sample. Their unevenness indicates that Grounding DINO exhibits submodularity most of the time, while Florence-2 is almost universally submodular. Our algorithm achieved acceleration on both models, and the difference in performance is precisely due to the difference between submodularity and supermodularity.

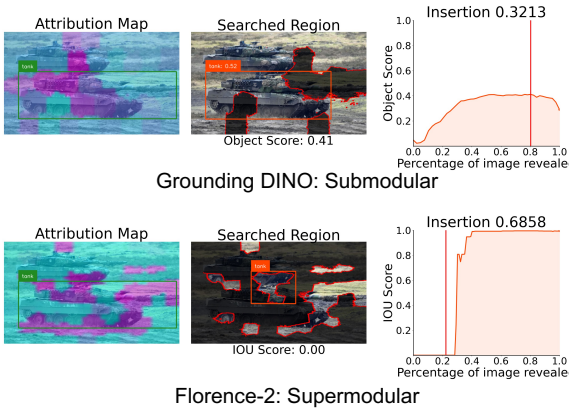


Figure 1. Insertion AUC under Greedy (VPS). Grounding DINO is almost concave with only a few exceptions, while Florence-2 is completely convex.

## G. Additional Visualization Results

In this section, we provide additional qualitative results to further illustrate the visual differences between the original Visual Precision Search (VPS) and our PhaseWin. Each figure presents one representative example from different tasks and datasets. For each case, we show side-by-side attribution maps highlighting how both methods localize critical regions that drive the prediction of object-level foundation models. These examples complement the quantitative results in Section 4.6, demonstrating that PhaseWin preserves interpretability quality while achieving substantial efficiency gains.

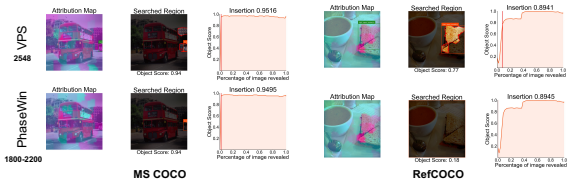


Figure 2. Comparison between VPS and PhaseWin on Florence-2 for correctly detected samples in MS COCO and RefCOCO. Both methods highlight semantically relevant regions, while PhaseWin produces equally faithful maps with far fewer evaluations.

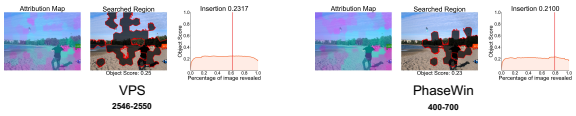


Figure 3. Visualization on Grounding DINO (MS COCO misclassification). VPS and PhaseWin consistently attribute the incorrect prediction to the same misleading region, confirming that PhaseWin maintains interpretive fidelity even in failure cases.

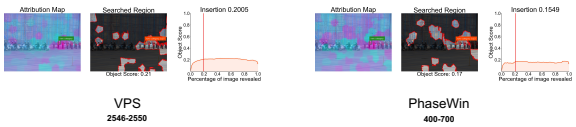


Figure 4. Visualization on Grounding DINO (LVIS misclassification). Both methods reveal the background context responsible for confusion, with PhaseWin matching the fine-grained localization quality of VPS at lower computational cost.

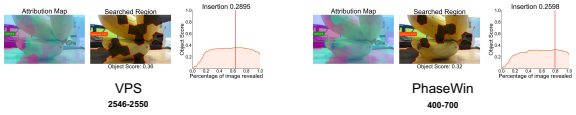


Figure 5. Visualization on Grounding DINO (MS COCO missed detection). VPS and PhaseWin identify the overlooked object region. PhaseWin effectively reproduces the trajectory of evidence accumulation with a fraction of the overhead.

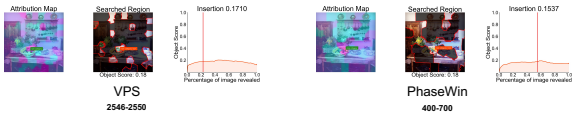


Figure 6. Visualization on Grounding DINO (LVIS missed detection). PhaseWin successfully recovers the same key evidence regions highlighted by VPS, showing its robustness on challenging zero-shot categories.

## References

[1] Jack Edmonds. Submodular functions, matroids, and certain polyhedra. *Combinatorial Structures and Their Applications*, pages 69–87, 1970. 3

[2] Satoru Fujishige. *Submodular functions and optimization*. Elsevier, 2005. 3

[3] Vitali Petsiuk, Abir Das, and Kate Saenko. RISE: Randomized input sampling for explanation of black-box models. In *BMVC*, page 151, 2018. 3

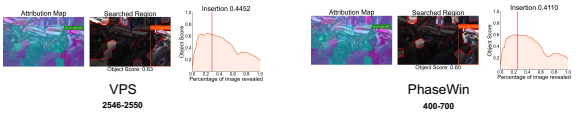


Figure 7. Visualization on Grounding DINO (RefCOCO grounding mistake). Both methods attribute the grounding failure to distractor regions, while PhaseWin provides nearly identical explanations with significantly fewer model evaluations.

[4] Jianming Zhang, Sarah Adel Bargal, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 126(10):1084–1102, 2018. 3