Semantic Correspondence with Visual Foundation Models

Men Ziyi s336749
Yang Xinyi s336897
Liu Xinhao s334446
github link:https://github.com/YAMA6331/Semantic-Correspondence-Project/tree/main

Abstract

This project investigates semantic correspondence using the visual foundation model DINOv2. To improve the stability of keypoint localization, lightweight fine-tuning strategies and a Window Soft-Argmax mechanism are introduced. Experiments conducted on SPair-71k, PF-Pascal, and PF-Willow demonstrate that the proposed approach achieves consistent performance gains and exhibits strong cross-dataset generalization ability.

1. Introduction

1.1. Background: What is Semantic Correspondence

Semantic correspondence aims to match semantically consistent keypoints or pixels across images despite large appearance and geometric variations. Recent visual foundation models learned via self-supervised pretraining provide transferable representations that effectively support this task.

1.2. Project Objective and Workflow

This project aims to investigate the effectiveness and cross-dataset generalization of pretrained visual foundation models for semantic correspondence. Given a source image with annotated keypoints and an unlabeled target image, the goal is to predict the corresponding locations of each source keypoint in the target image.

The workflow starts from a training-free baseline that establishes semantic correspondences using frozen pretrained features. Building on this baseline, lightweight fine-tuning is performed by unfreezing only the last layer of the model to improve task adaptation. During inference, keypoint locations are predicted based on feature similarity and refined using a more stable prediction strategy. Finally, the same model is evaluated on different datasets without additional training, and performance is analyzed using PCK metrics to assess cross-dataset generalization.

2. Training-free Baseline

2.1. DINOV2

2.1.1. Backbone and Feature Extraction

As a training-free baseline, we evaluate semantic correspondence using visual features extracted from a pretrained Vision Transformer. We adopt DINOv2 Base (ViT-B/14) as the backbone, which has been shown to encode rich semantic representations without explicit supervision.

Given a source image I_s and a target image I_t, both images are resized and normalized following the standard ImageNet preprocessing protocol. Each image is then forwarded through the DINOv2 encoder, from which we extract the patch-level features from the last transformer layer. For an input image of resolution $H \times W$ and patch size 14×14, this produces a dense feature grid of size $h \times w$, where $h = H/14$ and $w = W/14$. Each patch is represented by a D-dimensional feature vector.

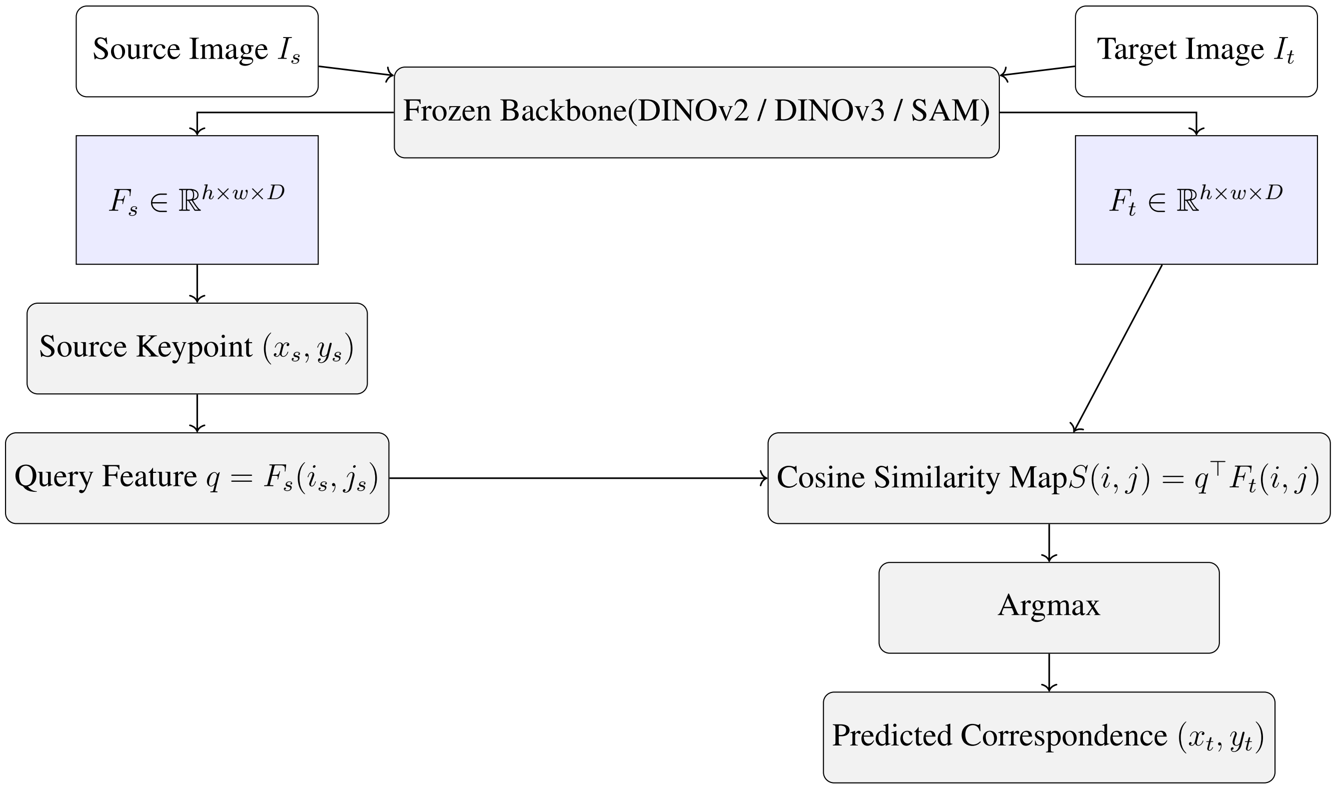
Formally, let $\mathbf{F}_s \in \mathbb{R}^{h \times w \times D}$ and $\mathbf{F}_t \in \mathbb{R}^{h \times w \times D}$ denote the patch-level feature grids extracted from the source and target images, respectively. All patch features are ℓ_2-normalized along the feature dimension prior to correspondence computation. During this stage, all backbone parameters remain fixed, such that the correspondence performance reflects the quality of the pretrained representations.

2.1.2. Correspondence Prediction

For each source image, annotated keypoints are provided by the SPair-71k dataset. Each source keypoint with image coordinates (x_s, y_s) is first mapped to the corresponding spatial location (i_s, j_s) in the source feature grid. The feature vector at this location, $\mathbf{q} = \mathbf{F}_s(i_s, j_s)$, is used as a query feature representing the semantic content of the source keypoint.

To identify the corresponding location in the target image, we compute a dense similarity map by comparing the query feature $\mathbf{q}$ with all patch features in the target feature grid. Specifically, for each target patch location (i, j), the cosine similarity is computed as

$$S(i,j) = \mathbf{q}^\top \mathbf{F}_t(i,j), \tag{1}$$



Source Image I_s
Target Image I_t
Frozen Backbone(DINOv2 / DINOv3 / SAM)
$F_s \in \mathbb{R}^{h\times w\times D}$
$F_t \in \mathbb{R}^{h\times w\times D}$
Source Keypoint (x_s, y_s)
Query Feature $q = F_s(i_s, j_s)$
Cosine Similarity Map$S(i,j) = q^\top F_t(i,j)$
Argmax
Predicted Correspondence (x_t, y_t)

Figure 1. Overview of the training-free semantic correspondence pipeline. Both source and target images are processed by a frozen backbone to extract dense feature grids. A query feature is obtained from the source feature grid at the annotated keypoint location. Correspondence is predicted by dense cosine similarity matching and argmax selection in feature space.

where both $\mathbf{q}$ and $\mathbf{F}_t(i,j)$ are ℓ_2-normalized. The resulting similarity map $S \in \mathbb{R}^{h\times w}$ assigns a similarity score to every target patch, indicating how well it matches the source keypoint in the learned feature space.

The final correspondence prediction is obtained by selecting the target patch with the maximum similarity score:

$$(i^*, j^*) = \arg\max_{(i,j)} S(i,j) \tag{2}$$

The predicted patch index (i^*, j^*) is then mapped back to image coordinates by rescaling according to the original target image resolution. This procedure yields a single correspondence prediction for each source keypoint.

2.1.3. Evaluation Protocol

We evaluate correspondence accuracy using the Percentage of Correct Keypoints (PCK) metric, following standard practice. Given N annotated keypoints, PCK at threshold α is defined as

$$\mathrm{PCK}(\alpha) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\big(\|\hat{\mathbf{p}}_i - \mathbf{p}_i\|_2 \le \alpha \cdot s\big) \tag{3}$$

where $\hat{\mathbf{p}}_i$ and $\mathbf{p}_i$ denote the predicted and ground-truth locations of the i-th keypoint, respectively, $\mathbb{I}(\cdot)$ is the indicator function, and s is a normalization factor that accounts for object scale.

In this work, the normalization scale s is defined as

$$s = \max(w_{\text{bbox}}, h_{\text{bbox}}) \tag{4}$$

where w_{bbox} and h_{bbox} denote the width and height of the target bounding box, respectively. A predicted keypoint is considered correct if its Euclidean distance to the ground-truth location is within $\alpha \cdot s$.

We report PCK at multiple thresholds $\alpha \in \{0.05, 0.1, 0.2\}$. Results are aggregated both per keypoint and per image, and per-category breakdowns are provided to analyze variations across object classes.

2.1.4. Results

We report results on the full SPair-71k test set using the DINOv2 Base backbone. Despite the absence of any task-specific training, the training-free baseline achieves competitive correspondence performance, indicating that meaningful semantic alignment naturally emerges from pre-trained visual features.

At $\alpha = 0.05$, the method achieves an overall PCK of approximately 36% per keypoint. Performance increases consistently for looser thresholds, reaching about 53% at

Table 1. Per-category breakdown of semantic correspondence performance (PCK per point) on SPair-71k.

Category	PCK@0.05	PCK@0.1	PCK@0.2
aeroplane	55.34	70.05	82.22
bicycle	38.67	60.14	74.99
bird	58.33	83.05	94.30
boat	19.70	34.05	50.95
bottle	27.17	44.69	64.43
bus	36.27	49.11	61.48
car	34.72	48.16	60.19
cat	59.31	68.99	82.35
chair	24.18	36.80	50.93
cow	45.26	64.37	81.67
dog	40.90	62.14	79.88
horse	39.03	63.33	82.12
motorbike	37.45	56.29	75.69
person	39.04	62.71	78.79
pottedplant	12.53	29.10	50.51
sheep	38.53	58.63	74.79
train	31.88	50.31	70.75
tvmonitor	12.85	25.86	43.78

$\alpha = 0.1$ and nearly 70% at $\alpha = 0.2$. These results demonstrate the robustness of DINOv2 features for semantic correspondence under a simple argmax-based matching strategy.

Per-category analysis reveals substantial variation across object classes. Articulated categories such as birds, cats, and dogs achieve relatively high PCK scores, while rigid or texture-poor objects such as bottles, potted plants, and TV monitors remain more challenging. This suggests that correspondence performance is strongly influenced by object appearance diversity and the availability of distinctive semantic cues.

2.2. DINOv3

2.2.1. Backbone and Feature Extraction

In addition to DINOv2, we evaluate DINOv3 as a stronger visual foundation model under a training-free setting. We adopt the DINOv3 ViT-Large-16 backbone, which features increased model capacity and is pretrained on more recent large-scale data.

The overall preprocessing and evaluation pipeline is identical to that used for DINOv2, ensuring a fair comparison without any task-specific training or adaptation. Input images are resized and normalized following the standard ImageNet protocol.

Given an input image of resolution $H \times W$ and a patch size of 16×16, the DINOv3 encoder produces a dense patch-level feature grid of size $h \times w$, where $h = H/16$ and $w = W/16$. Unlike DINOv2, the output token sequence of DINOv3 includes several special tokens at the beginning, namely one `[CLS]` token followed by four register tokens. These tokens are removed prior to correspondence computation, and only the remaining patch tokens are reshaped into the spatial feature grid.

Let $\mathbf{F}_s \in \mathbb{R}^{h\times w\times D}$ and $\mathbf{F}_t \in \mathbb{R}^{h\times w\times D}$ denote the resulting ℓ_2-normalized feature grids extracted from the source and target images, respectively.

2.2.2. Correspondence Prediction and Evaluation

Correspondence prediction and evaluation follow exactly the same procedure as described for the DINOv2 baseline. Source keypoint features are used as queries to compute dense cosine similarity maps over the target feature grid, and final correspondences are obtained via argmax selection and mapped back to image coordinates. Performance is evaluated using the PCK metric at thresholds $\alpha \in \{0.05, 0.1, 0.2\}$.

2.2.3. Results

We report training-free correspondence results of DINOv3 on the SPair-71k test set. DINOv3 achieves a PCK of approximately 36% at $\alpha = 0.05$, increasing to about 55% at $\alpha = 0.1$ and over 70% at $\alpha = 0.2$.

Overall, DINOv3 exhibits correspondence accuracy comparable to that of DINOv2 under the same argmax-based matching strategy. This suggests that the increased capacity of DINOv3 is not fully exploited in the training-free setting, motivating further investigation through fine-tuning or improved prediction strategies.

2.3. SAM

2.3.1. Segment Anything Backbone

In addition to DINO-based models, we evaluate the Segment Anything Model (SAM) as a training-free baseline for semantic correspondence. We adopt the SAM ViT-Large image encoder, which is pretrained for generic image segmentation and provides dense visual representations. For a fair comparison, the SAM backbone is kept fully frozen, and the same correspondence prediction and evaluation protocol as the DINO baselines is used.

2.3.2. Feature Extraction with SAM

Input images are resized to 1024×1024 and processed by the SAM image encoder. Unlike DINO-based Vision Transformers that output patch tokens, SAM produces dense convolutional feature maps.

Specifically, the SAM ViT-Large encoder outputs a feature tensor of spatial size $H \times W$ with $H = W = 64$, where each feature location corresponds to a 16×16 pixel region in the input image. We denote the extracted feature maps from the source and target images as $\mathbf{F}_s \in \mathbb{R}^{H\times W\times D}$ and $\mathbf{F}_t \in \mathbb{R}^{H\times W\times D}$, respectively. All feature vectors are ℓ_2-

Table 2. Comparison of training-free baselines on SPair-71k. Results are reported using PCK at different thresholds.

Threshold	DINOv2		DINOv3		SAM	
	Point (%)	Image (%)	Point (%)	Image (%)	Point (%)	Image (%)
0.05	36.04	33.64	35.78	32.80	15.67	13.21
0.10	53.42	50.75	54.86	51.12	24.58	21.27
0.20	69.93	67.15	71.32	67.44	38.68	34.30

normalized along the channel dimension prior to similarity computation.

2.3.3. Correspondence Prediction and Results

Correspondence prediction with SAM follows the same argmax-based matching strategy used for the DINO baselines. A query feature is selected from the source feature map at the annotated keypoint location, and a dense cosine similarity map is computed over the target feature map. The final correspondence is obtained by argmax selection and mapped back to image coordinates.

We report training-free correspondence results on the SPair-71k test set. Under the same prediction strategy, SAM consistently underperforms DINO-based models. At $\alpha = 0.05$, SAM achieves a PCK of approximately 16%, increasing to about 25% at $\alpha = 0.1$ and remaining below 40% at $\alpha = 0.2$.

This performance gap suggests that, although SAM learns strong representations for segmentation, its features are less directly aligned with semantic correspondence when used without task-specific adaptation.

2.4. Comparison of Training-free Baselines

We summarize the performance of all training-free baselines in Tab. 2, comparing DINOv2, DINOv3, and SAM under the same evaluation protocol.

Overall, DINOv2 and DINOv3 exhibit comparable correspondence accuracy across all thresholds. DINOv3 consistently performs slightly better at looser thresholds, suggesting that its increased model capacity benefits coarse semantic alignment. However, the performance gap between the two models remains relatively small in the absence of task-specific adaptation.

In contrast, SAM yields substantially lower PCK scores at all thresholds. This observation suggests that representations optimized for generic segmentation do not directly transfer to semantic correspondence under a simple argmax-based matching rule, motivating the exploration of adaptation strategies and improved prediction mechanisms in subsequent stages.

3. Light Fine-tuning Strategy

While DINOv2 offers robust pre-trained features, its frozen representations struggle with significant appearance or pose changes in semantic correspondence. To address this, we employ a Light Fine-tuning strategy by updating only the last few layers. This adapts the feature space for pixel-level matching, balancing pre-trained generalization with task-specific discrimination while effectively mitigating overfitting on limited data.

3.1. Methodology

3.1.1. Network Architecture and layers freezing

We utilize dinov2-vitb14 as our backbone network. To ensure the process keeping correctly and to preserve the robustness of the pretrained features, we freeze the majority of the network's parameters. We define a parameter k to control the depth of fine-tunin g. We only unfreeze and update the last k Transformer Blocks and the final normalization layer. In our experiments, we tested $k \in \{1, 2, 3\}$ to find the optimal balance.

3.1.2. Similarity Computation and Heatmap

Given a source image I_s and a target image I_t, the model extracts their feature maps F_s and F_t. For a specific annotated keypoint $p_s^{(i)}$ in the source image, we extract its corresponding feature vector $f_s^{(i)}$. We then calculate the cosine similarity between this vector and every position in the target feature map to generate a similarity heatmap $S^{(i)}$:

$$S^{(i)}(x, y) = \frac{f_s^{(i)} \cdot F_t(x, y)}{\|f_s^{(i)}\| \|F_t(x, y)\|} \tag{5}$$

3.1.3. Differentiable Coordinate Regression

To train the model using keypoint coordinates, we need a way to back propagate the error. However, the standard argmax operation (which simply picks the highest value) is not differentiable. Therefore, we use a global Soft-argmax mechanism.

First, we convert the similarity heatmap $S^{(i)}$ into a probability distribution $P^{(i)}$ using a Spatial Softmax with a temperature parameter β:

$$P^{(i)}(x, y) = \frac{\exp(\beta \cdot S^{(i)}(x, y))}{\sum_{x',y'} \exp(\beta \cdot S^{(i)}(x', y'))} \tag{6}$$

Here, β controls how sharp the distribution is. Next, we calculate the expected coordinates (a weighted average) to get the predicted position $\hat{p}_t^{(i)}$:

$$\hat{p}_t^{(i)} = \sum_{x,y} \begin{pmatrix} x \\ y \end{pmatrix} \cdot P^{(i)}(x, y) \tag{7}$$

This allows gradients to flow from the loss function back into the feature extraction network.

3.1.4. Loss Function

The model is supervised using the ground-truth keypoints provided by the SPair-71k dataset. The loss function is defined as the Mean Squared Error (MSE) between the predicted coordinates $\hat{p}_t^{(i)}$ and the ground truth coordinates $p_{gt}^{(i)}$:

$$\mathcal{L}_{dense} = \frac{1}{N} \sum_{i=1}^{N} \|\hat{p}_t^{(i)} - p_{gt}^{(i)}\|_2^2 \tag{8}$$

We also utilized Automatic Mixed Precision (AMP) to accelerate the training process.

3.2. Results and Analysis

3.2.1. Results of Fine-tuning

To comprehensively evaluate the light fine-tuning strategy, we conducted experiments by unfreezing different depths of the backbone network (specifically the last 1, 2, and 3 layers) and assessing the performance across multiple precision thresholds ($\alpha \in \{0.05, 0.1, 0.2\}$). The comparative results against the frozen baseline are visualized in Figure 2.

As illustrated in the figure, a consistent trend is observed across all thresholds: the model achieves optimal performance when fine-tuning only the last 1 Layer. Taking the primary metric $\alpha = 0.1$ as an example, the PCK score improves significantly from the baseline of 53.4% to a peak of **73.7%**.

However, extending the fine-tuning depth further leads to a performance regression. As the number of unfrozen layers increases to 2 and 3, the accuracy at $\alpha = 0.1$ drops to 71.1% and 68.5%, respectively. This pattern indicates that unfreezing excessive deep parameters on this dataset causes overfitting, confirming that fine-tuning just the single last layer is the most effective configuration.

3.2.2. Analysis

Table 3. Performance of DINOv2 with different fine-tuning depths (PCK@0.1) compare with baseline.

Layers	PCK@0.1 (PP)	PCK@0.1 (PI)	Improvement
Baseline	53.42	50.75	-
1	**73.74**	**70.91**	**+20.32**
2	71.05	67.88	+17.63
3	68.51	64.61	+15.09

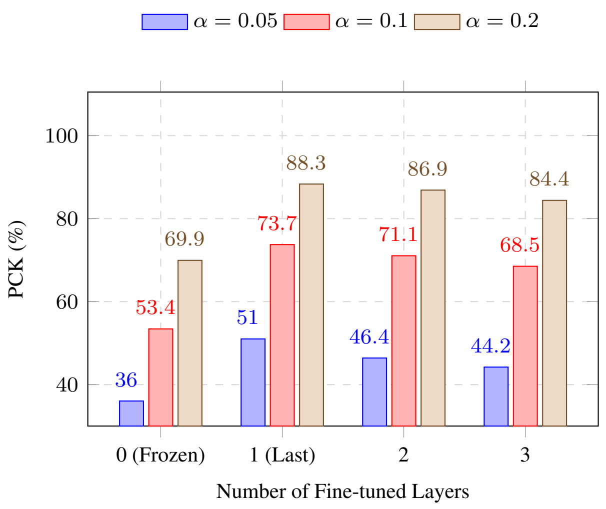


Figure 2. **Comparison of Fine-tuning Depths.**

Data Analysis and Conclusion

- **Significant Gains from Supervised Fine-tuning:** Compared to the completely frozen baseline model, implementing light fine-tuning resulted in a massive performance leap. By unfreezing and training just the last 1 Layer, the keypoint-level accuracy (Per Point PCK@0.1) surged from **53.42%** to **73.74%**, an improvement of 20.32%. This provides strong evidence that while pretrained generic features are powerful, task-specific adaptation can significantly unlock their potential for fine-grained semantic matching tasks.
- **Effect of layer depth:** Our experiments show that unfreezing more layers (increasing from 1 to 3) actually lowers accuracy. This is mainly due to overfitting on a small dataset. The deep layers of DINOv2 are already well-trained on massive data and contain robust general features. Since our SPair-71k dataset is small, unfreezing the last three layers opens up too many parameters. The model starts to memorize the noise of the training data instead of learning general rules, which damages the original pretrained features. In contrast, fine tuning only the last layer acts like a small adjustment to the existing powerful features. It adapts the model to the new task without losing its original generalization ability, leading to the best results.

Based on the quantitative analysis above, fine-tuning only the Last 1 Layer is confirmed as the optimal strategy for the DINOv2 model in semantic correspondence tasks. This strategy not only achieved the highest accuracy with minimal computational cost but also effectively avoided the risk of overfitting. Therefore, the subsequent experiments on Prediction Refinement will be based on this 1-Layer fine-tuned model.

4. Prediction Strategy

4.1. Objective

In the previous phase, Light Fine-tuning significantly enhanced the discriminative power of DINOv2 features for semantic correspondence. However, optimizing the feature space alone is insufficient to achieve pixel-level precision due to the fundamental limitations of the coordinate decoding method used during inference.

The objective of this phase is to introduce the **Window Soft-Argmax (WSA)** strategy to replace the simple `argmax`. This method aims to combine the robustness of global search with the precision of local regression, overcoming grid resolution limits to achieve precise dense matching.

4.2. Methodology

4.2.1. Coarse Localization

The prediction process is divided into three steps: "Coarse Localization" "Windowing" and "Local Refinement". In the part of coarse localization, we compute the similarity heatmap $S \in \mathbb{R}^{H \times W}$ for a given source keypoint. We use argmax to identify the global peak position p_{max}, which serves as the candidate matching center:

$$p_{max} = \arg\max_{q \in \Omega} S(q) \tag{9}$$

where Ω represents the entire feature grid domain. This step ensures that the match does not fall into local optima far from the true target.

4.2.2. Windowing

To isolate noise and focus on the target region, we crop a fixed-size local window $\mathcal{W}$ centered at p_{max}. In our experiments, the window size was set to 5×5 (radius $r = 2$). The set of positions within the window is defined as:

$$\mathcal{W} = \{q \mid \|q - p_{max}\|_\infty \leq r\} \tag{10}$$

4.2.3. Local Refinement with Soft-argmax

Within the local window $\mathcal{W}$, we apply the **Soft-argmax** algorithm to calculate the probabilistic centroid. First, we use a Softmax function with a temperature coefficient τ to convert the similarity scores within the window into a probability distribution $P_{\mathcal{W}}$:

$$P_{\mathcal{W}}(q) = \frac{\exp(\tau \cdot S(q))}{\sum_{z \in \mathcal{W}} \exp(\tau \cdot S(z))}, \quad \forall q \in \mathcal{W} \tag{11}$$

In our code implementation, the temperature coefficient τ is set to **10.0**. A higher temperature helps sharpen the distribution, focusing the model's attention on high-confidence areas.

Finally, we calculate the spatial expectation of this local probability distribution to obtain the final sub-pixel predicted coordinate $\hat{p}_{final}$:

$$\hat{p}_{final} = \sum_{q \in \mathcal{W}} q \cdot P_{\mathcal{W}}(q) \tag{12}$$

By doing so, the prediction is no longer constrained to integer grid points but can lie anywhere in the continuous space between grid cells, effectively eliminating quantization error.

4.3. Results and Analysis

4.3.1. Results of WSA

In table 4, to evaluate the effectiveness of the prediction refinement strategy, we conducted a comparative analysis using our best-performing model—specifically, the DINOv2 backbone with the last layer fine tuned as the foundation. We contrasted the performance of the standard argmax approach against the Window Soft-Argmax (WSA) mechanism under identical experimental conditions.

Table 4. **Effectiveness of Window Soft-Argmax (WSA).**

Method	Per Point (PCK)			Per Image (PCK)		
	0.05	0.1	0.2	0.05	0.1	0.2
Frozen Baseline	36.04	53.42	69.93	33.64	50.75	67.15
Fine-tuned (Last 1)	51.00	73.74	88.33	47.91	70.91	86.41
Fine-tuned + WSA	**53.16**	**75.74**	**88.99**	**49.87**	**72.81**	**87.14**

4.3.2. Analysis of Results

We focus our analysis on the $\alpha = 0.1$ threshold, which serves as the primary benchmark for precision. The results demonstrate a stepwise improvement:

First, comparing the Frozen Baseline with the Fine-tuned (Last 1 layer) model, the PCK@0.1 (Per Point) surges from 53.42% to 73.74%, a massive improvement of **+20.32%**. This confirms that task adaptation via light fine-tuning is critical for high-precision matching.

Second, after introducing Window Soft-Argmax, the performance further climbs to **75.74%**, adding an additional gain of **2.00%**. This improvement is significant as it stems solely from inference optimization. While standard Argmax is limited to discrete grid centers, WSA enables *sub-pixel* accuracy by mitigating quantization errors through local centroid regression.

5. Test on New Datasets

5.1. PF-Pascal and PF-Willow Dataset Comparison

PF-Pascal and PF-Willow are semantic correspondence benchmarks based on paired images with keypoint annotations.

Table 5. **Overall PCK Results on PF Datasets (Per-point vs. Per-image).**

Dataset	**Per-point (PCK)**			**Per-image (PCK)**		
	0.05	0.1	0.2	0.05	0.1	0.2
PF-Willow (Overall)	46.7	72.5	93.2	46.7	72.5	93.2
PF-Pascal (Overall)	67.5	86.1	95.2	66.8	85.2	94.3

PF-Pascal contains 20 semantic categories, providing broad category coverage and making it suitable for evaluating overall cross-category generalization.

PF-Willow is smaller in scale and includes only four basic semantic categories, but introduces finer-grained intra-class variations, imposing higher demands on medium-precision keypoint localization stability.

Compared to PF datasets, SPair-71k is larger in both category number and dataset scale, with different data distributions and annotation protocols, making PF datasets particularly useful for analyzing cross-dataset generalization.

All experiments on PF-Pascal and PF-Willow use DINOv2-ViT-B/14 with one-layer fine-tuning and Window Soft-Argmax (WSA).

5.2. Results

5.2.1. Evaluation Consistency on PF-Willow

In the PF-Willow dataset, each image pair is annotated with a fixed number of semantically consistent keypoints, and the keypoint count remains identical across samples. Under this annotation setting, per-point PCK (aggregated over all keypoints) and per-image PCK (first computing keypoint accuracy per image, then averaging over images) are mathematically equivalent, resulting in identical or highly similar numerical values (see Table 5). This behavior reflects the intrinsic annotation structure of the PF-Willow dataset rather than any abnormality in model predictions.

In contrast, PF-Pascal contains image pairs with varying numbers of annotated keypoints, leading to different statistical weightings between per-point and per-image PCK. This difference explains the observed discrepancy between the two metrics.

5.2.2. Cross-Dataset Comparison under PCK@0.1

Under the PCK@0.1 threshold, the model exhibits clear performance differences across datasets. On PF-Pascal, the model achieves an overall accuracy of 86.10, representing an improvement of 10.36 percentage points over the SPair-71k baseline (75.74). This gain indicates that the broader category coverage and relatively stable object structures in PF-Pascal enable more effective preservation of semantic consistency and reliable medium-precision keypoint localization.

In contrast, the model attains a PCK@0.1 score of 72.50 on PF-Willow, which is 3.24 percentage points lower than that on SPair-71k. This drop can be attributed to the smaller dataset scale and finer-grained intra-category structure of PF-Willow, where more concentrated keypoint distributions make medium-precision semantic alignment more challenging. The quantitative comparison across datasets under the PCK@0.1 threshold is summarized in Table 6.

Table 6. Cross-dataset comparison under PCK@0.1

Dataset	PCK@0.1	Difference vs. SPair-71k
SPair-71k	75.74	–
PF-Pascal	86.10	+10.36
PF-Willow	72.50	−3.24

6. Conclusion

This work investigates the applicability of visual foundation models to the semantic correspondence task. A training-free semantic correspondence baseline is constructed using DINOv2 ViT-B/14, demonstrating that pretrained visual features possess strong cross-instance semantic alignment capabilities. Comparisons with DINOv3 and SAM further show that the DINO family is better suited for keypoint-level semantic correspondence, while purely training-free approaches suffer from clear performance limitations.

Building upon this baseline, a lightweight fine-tuning strategy is adopted by unfreezing only the final Transformer layer, which significantly improves performance while effectively avoiding overfitting. In addition, Window Soft-Argmax (WSA) is introduced for prediction-stage keypoint refinement, leading to stable and consistent performance gains under medium-precision thresholds such as PCK@0.1.

Cross-dataset experiments indicate that the proposed method generalizes well to PF-Pascal, which features broader category coverage, while remaining challenged by the finer-grained and smaller-scale PF-Willow dataset.