Towards Complementary Keypoint Detection via Mixture of Detectors

Xiaoyong Lu
Southeast University
luxiaoyong@seu.edu.cn

Guobao Xiao
Tongji University
gbx@tongji.edu.cn

Dong Liang
Nanjing University of Aeronautics and Astronautics
liangdong@nuaa.edu.cn

Songlin Du*
Southeast University
sdu@seu.edu.cn

Abstract

Keypoint detection plays a central role in geometric vision pipelines. Existing detectors exhibit complementary strengths across different scene structures and keypoint budgets, and reliance on a single detector often results in suboptimal keypoint distributions. Motivated by this observation, we propose **Mixture of Detectors (MoD)** for dense matching pipelines, a unified framework for complementary keypoint detection through the adaptive composition of heterogeneous detectors. MoD first distills heterogeneous detectors into a unified model, allowing diverse detection maps to be predicted efficiently within a single forward pass. Detector selection is subsequently formulated as a patch-wise, budget-conditioned decision process, in which a lightweight router assigns the most suitable detector to each local region. To avoid heuristic supervision, the router is trained with task-driven rewards derived from downstream matching quality, jointly capturing both keypoint quantity and precision. Extensive experiments on MegaDepth, ScanNet, and HPatches demonstrate that MoD consistently outperforms strong baselines across a wide range of keypoint budgets, demonstrating its effectiveness in improving keypoint distribution and downstream geometric performance.

1 Introduction

Keypoint detection aims to identify distinctive image structures and constitutes a fundamental component of geometric vision pipelines, including Structure-from-Motion (SfM) Schonberger and Frahm (2016), Simultaneous Localization and Mapping (SLAM) Engel et al. (2017), and image matching Sarlin et al. (2020). In classical sparse matching pipelines, two sets of keypoints are detected and subsequently matched, and the quality of matching relies heavily on repeatability under varying illumination and viewpoint conditions. Recent dense matching approaches Edstedt et al. (2024c); Lu et al. (2026) have partially relaxed this requirement by detecting keypoints in one image and directly establishing correspondences in another, shifting the focus from strict repeatability to the spatial distribution of detected keypoints. As a result, how to obtain an effective and adaptive keypoint distribution becomes increasingly important for downstream performance.

In practice, different detectors exhibit complementary behaviors across scene structures and keypoint budgets. This variation stems from their distinct design choices, including normalization strategies (*e.g.*, local or global Softmax versus independent Sigmoid), training objectives (*e.g.*, supervised learning, self-supervised learning, or reinforcement learning), and the types of geometric primitives they capture (*e.g.*, points, lines, and edges).

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

These differences lead to substantially different detection patterns: some detectors favor sparse and highly reliable keypoints, while others provide denser coverage over structured or low-texture regions. Moreover, their relative performance varies significantly with the target number of keypoints, *i.e.*, the keypoint budget. Consequently, no single detection mechanism performs consistently well across all spatial regions and budget settings.

Prior work such as RCM+ Lu et al. (2026) shows that combining multiple detectors improves performance due to complementary distributions. However, such approaches rely on simple concatenation or heuristic fusion, which does not explicitly model detector selection and cannot fully exploit complementary strengths. In particular, the optimal detector varies across images and depends on **local textures** and the target **keypoint budget**. This leads to a more general formulation, where keypoint detection selects suitable detectors across spatial locations under a given budget. Such a formulation introduces several challenges: (i) running all candidate detectors is computationally expensive, (ii) detector selection must jointly account for local textures and global keypoint budget, and (iii) no direct supervision is available for detector selection.

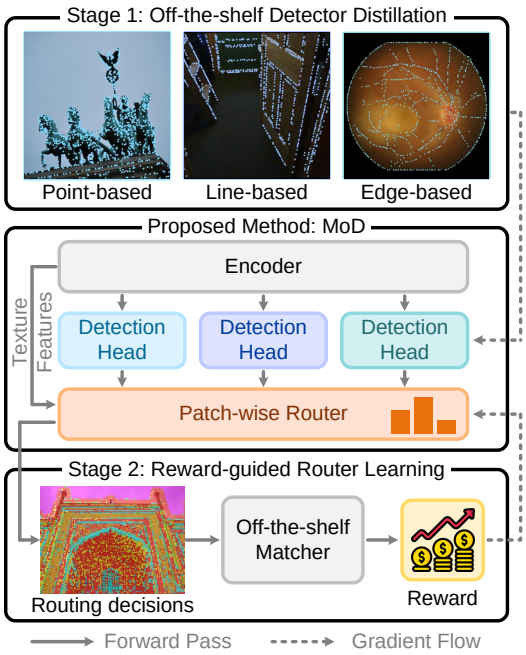

Figure 1: Different detectors exhibit complementary behaviors across varying image structures and keypoint budgets. MoD leverages this by first distilling heterogeneous detectors into a unified model for joint prediction (Stage 1), and then learning a reward-guided router that selects detectors based on local texture and the keypoint budget (Stage 2).

To address these challenges, we propose **Mixture of Detectors (MoD)**, a unified framework for adaptive detector composition. Motivated by the Mixture-of-Experts (MoE) paradigm Shazeer et al. (2017), we treat point-, line-, and edge-based detectors as specialized experts, and formulate selection as a routing problem over these experts. (i) To reduce the cost of multiple detectors, MoD distills heterogeneous detectors into a single model that predicts diverse detection maps in a forward pass, preserving complementary behaviors while maintaining efficiency. (ii) Detector selection is formulated as a patch-wise, budget-conditioned process. A lightweight router assigns detectors to regions based on local texture, while a budget-aware mechanism introduces a global prior that modulates preference by desired keypoint density. (iii) To address the lack of supervision, detector selection is guided by task-driven rewards from downstream matching performance, providing patch-wise supervision that captures keypoint quantity and precision.

To summarize, this paper makes the following contributions: **(1)** We present a new perspective on keypoint detection by modeling the complementary behaviors of heterogeneous detectors, and introduce a detector selection problem conditioned on spatial regions and keypoint budgets. **(2)** We introduce a unified detector that distills heterogeneous detectors into a shared model, enabling efficient prediction of diverse detection maps in a single forward pass. **(3)** We propose a patch-wise, budget-aware router that adaptively selects detectors based on local texture and global keypoint budget, trained with task-driven rewards that jointly capture keypoint quantity and precision. **(4)** Extensive experiments on MegaDepth, ScanNet, and HPatches show that MoD consistently improves downstream geometric estimation across a wide range of keypoint budgets while maintaining competitive efficiency.

2 Related Work

Detection. Early keypoint detection methods relied on handcrafted designs, among which SIFT Lowe (2004) is representative. These methods detect salient structures using manually designed operators, achieving reasonable repeatability under moderate variations. Recent learning-based approaches have advanced keypoint detection. Methods such as SuperPoint DeTone et al. (2018),

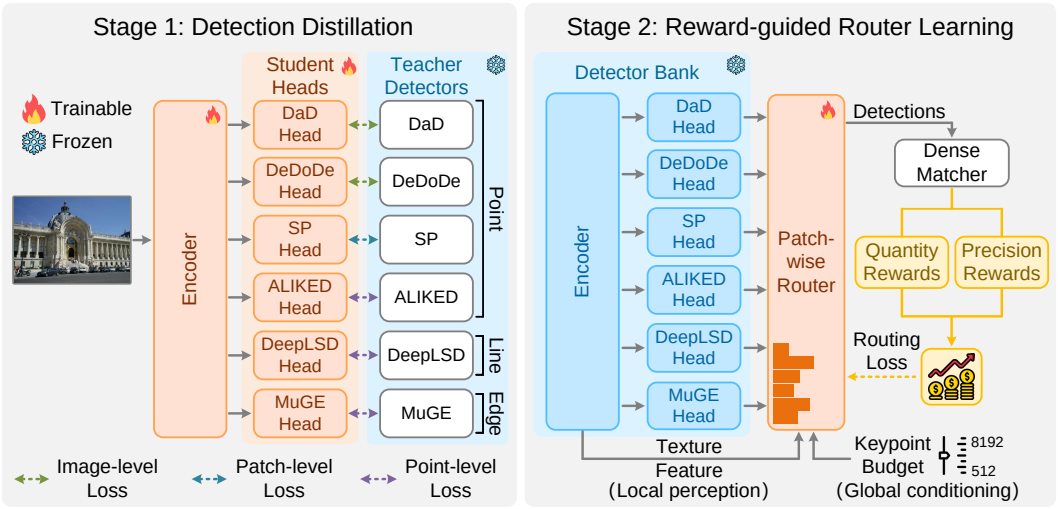

Figure 2: **Overview of the proposed method.** In stage 1, we distill multiple teacher detectors into student heads. In stage 2, we train a patch-wise router with reward-guided supervision to select detectors based on local texture and keypoint budget.

D2-Net Dusmanu et al. (2019), DISK Tyszkiewicz et al. (2020), and ALIKED Zhao et al. (2023) use deep neural networks to jointly learn detection and description from data, improving robustness across applications. Furthermore, DeDoDe Edstedt et al. (2024b,a) and DaD Edstedt et al. (2025a) adopt a decoupled detection-description paradigm, motivated by inconsistent optimization objectives between detection and description. Joint optimization can lead to suboptimal solutions and restrict flexibility of keypoint distributions. Following this line, MoD adopts a fully decoupled detection formulation. Beyond point-based detection, line segment detection methods such as LSD Grompone von Gioi et al. (2010), SOLD^2 Pautrat et al. (2021), and DeepLSD Pautrat et al. (2023) capture structured geometric primitives, particularly in man-made environments with prevalent linear structures. Similarly, edge detection methods, including Canny Canny (1986), EDTER Pu et al. (2022), and MuGE Zhou et al. (2024), identify object boundaries and fine-grained contours often missed by point-based detectors. These detection methods provide complementary information to point-based detection, enriching spatial coverage. However, the complementarity among different detection methods remains largely under-explored.

Mixture of Experts. Mixture of Experts (MoE) Shazeer et al. (2017); Riquelme et al. (2021); Zhou et al. (2022); Chen et al. (2023) is a widely adopted paradigm that improves model capacity through experts and a routing mechanism. In most works, particularly in large-scale language and vision models, experts correspond to different feed-forward networks with the same architecture, and the router activates a subset of them for each input token. Such designs enhance representation learning and computational efficiency via conditional computation. In this work, different types of detectors, including point-, line-, and edge-based methods, are treated as experts rather than multiple networks with similar functions. These detectors capture distinct geometric structures and exhibit complementary behaviors across scenes, improving keypoint distribution and downstream performance in dense matching pipelines.

3 Method

As illustrated in Figure 2, MoD employs a shared encoder to extract features, followed by multiple detection heads that produce detection maps. In Stage 1, distillation from six state-of-the-art detectors enables MoD to generate six maps in a single forward pass. In Stage 2, the encoder and heads are frozen, and a router uses texture features and keypoint budget to assign the most suitable detector to each 8×8 patch. Routing decisions are evaluated with an off-the-shelf matcher, RoMa v2 Edstedt et al. (2025b), which provides rewards based on keypoint quantity and precision to supervise training.

3.1 Stage 1: Detection Distillation

3.1.1 Feature Extraction

We adopt a VGG Simonyan and Zisserman (2015) network as the encoder to extract multi-scale dense feature maps from the input image I. VGG provides rich local texture features, essential for keypoint detection and subsequent routing. Specifically, features are collected from different backbone stages to form a feature pyramid $\{F_1, F_{1/2}, F_{1/4}, F_{1/8}\}$, where each subscript denotes the spatial resolution relative to I. The unified encoder aggregates knowledge from multiple teachers during distillation, yielding more robust and generalizable texture representations. As a result, *the student may even outperform individual teachers*.

Table 1: **Inductive Biases of the Selected Detector Experts.**

Detector	Inductive Biases
SuperPoint	Self-supervised corners + patch-wise softmax
DaD	Reinforcement learning + image-wise softmax
DeDoDe	SfM-supervised points + image-wise softmax
ALIKED	Reprojection/peak losses + pixel-wise sigmoid
DeepLSD	Line detection
MuGE	Edge detection

3.1.2 Detection Head

Because detectors produce detection probabilities via different mechanisms, we introduce two types of detection heads: **image-level** and **patch-level**. Specifically, ALIKED Zhao et al. (2023), DeDoDe Edstedt et al. (2024a), DaD Edstedt et al. (2025a), DeepLSD Pautrat et al. (2023), and MuGE Zhou et al. (2024) generate keypoint probabilities at the image level, whereas SuperPoint DeTone et al. (2018) predicts probabilities at the patch level.

Image-level head. For detectors operating on full-resolution maps, we adopt depthwise convolution refiners from DKM Edstedt et al. (2023). The prediction follows a coarse-to-fine refinement across $\{F_{1/8}, F_{1/4}, F_{1/2}, F_1\}$. At each scale, a refiner takes the encoder feature and predicts a residual added to upsampled logits from the previous scale. This progressive refinement recovers fine spatial details while preserving global consistency. After obtaining the final logits, different normalization functions are applied to align with teacher behaviors. For DeDoDe and DaD, we apply Softmax over all pixels. For ALIKED, DeepLSD, and MuGE, we apply Sigmoid independently per pixel.

Patch-level head. For SuperPoint, which predicts keypoints at the patch level, we introduce a head on $F_{1/8}$. Specifically, a VGG-style block predicts a 65-channel logit map, where 64 channels correspond to the 8×8 locations within each patch and one channel represents a *dustbin* class, *i.e.*, a no-keypoint class. Softmax is applied over the 65 channels, then the dustbin channel is discarded. The remaining probabilities are reshaped to recover a dense detection map at the original resolution.

3.1.3 Learning Objective

To faithfully distill heterogeneous detectors, we adopt different objectives according to how each method generates detection probabilities. **(1) Image-level Supervision.** For DeDoDe and DaD, which generate probabilities via spatial Softmax over image-level logits, we use image-level Kullback-Leibler (KL) divergence to align the predicted distributions. **(2) Patch-level Supervision.** For SuperPoint, which predicts probabilities in each 8×8 patch, we compute KL divergence within each patch to align the predicted distributions. **(3) Point-level Supervision.** For ALIKED, DeepLSD, and MuGE, which model probabilities independently per pixel using Sigmoid, we regress the logit maps with an ℓ_2 loss. The overall objective sums the losses over all detectors.

3.2 Stage 2: Reward-guided Router Learning

3.2.1 Router

In Stage 2, a router is trained with reward-guided supervision to select the most suitable detector per patch based on local texture and global keypoint budget. As illustrated in Figure 3, the router has two branches, a **budget branch** and a **texture branch**, whose outputs are fused for final routing decisions.

Texture branch. The texture branch captures local patterns to determine the appropriate detector for each region. It takes the frozen encoder feature at $1/8$ resolution and processes it through a CNN-based encoder to produce a spatial feature map.

This branch models local structures such as corners, edges, and low-texture regions, enabling the router to exploit spatial heterogeneity. Let $h = H/8$ and $w = W/8$. The output, denoted by $\hat{R} \in \mathbb{R}^{(K+1)\times h\times w}$, provides patch-wise preferences over detectors. Here, K is the number of detectors, and the additional class represents a background option where no detector is selected.

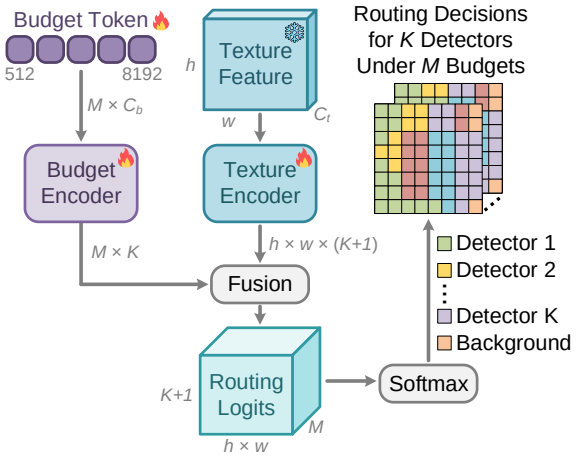

Figure 3: **Detector router conditioned on keypoint budget and local texture.**

Budget branch. In addition to texture, detector selection depends on the desired number of keypoints. We introduce a budget branch that encodes the target keypoint budget into a learnable representation. Specifically, $M = 5$ budget tokens, corresponding to keypoint budgets $\{512, 1024, 2048, 4096, 8192\}$, are learned during training. Budget tokens are processed through an MLP-based encoder to produce a global bias $b \in \mathbb{R}^{K\times M}$. In addition to these five budgets, other budgets can be handled by interpolating or extrapolating the budget tokens, *e.g.*, linearly interpolating the 1024 and 2048 tokens to obtain the detector bias for a budget of 1500.

Fusion and routing decision. The outputs of the two branches are combined to produce routing logits. For each budget level $m \in \{1, \ldots, M\}$, the bias $b^m \in \mathbb{R}^K$ is added to the first K channels of $\hat{R}$. The routing logits are denoted by $R \in \mathbb{R}^{(K+1)\times M\times h\times w}$. The budget bias is broadcast to all locations and acts as a global prior modulating detector preference by keypoint density. The fused logits are passed through a Softmax along the detector dimension to yield selection probabilities for each 8×8 patch, denoted by $P \in \mathbb{R}^{(K+1)\times M\times h\times w}$.

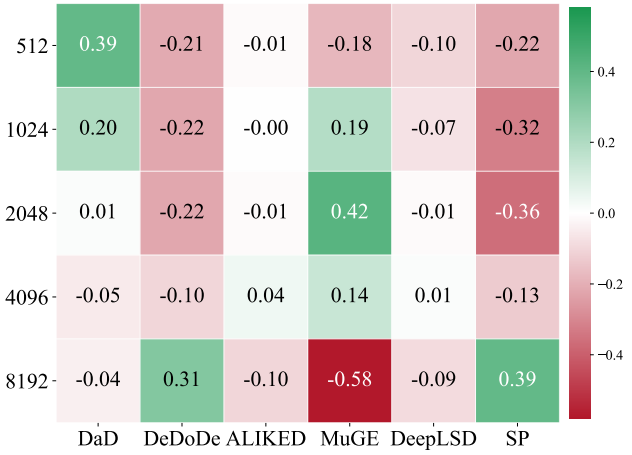

Figure 4: **Learned biases of different detectors under varying keypoint budgets.** The vertical axis denotes the budgets, and the horizontal axis corresponds to the detectors.

The router can be interpreted as a patch-wise policy determining how the final detection map is composed from distilled heads. By jointly modeling texture and budget, the router enables flexible, context-aware detector selection, allowing detectors to specialize across local regions and keypoint densities.

3.2.2 Reward

We construct a patch-wise reward from matching quality of keypoints predicted by different detection heads under varying budgets. We extract keypoints from all K detection heads under M budgets in the source image, and use an off-the-shelf matcher Edstedt et al. (2025b) to find correspondences in the target image. Let x denote keypoints in the source image, x_p their predicted matched positions, and x_{gt} the ground-truth matched positions. We compute the error e for each match as the Euclidean distance between x_p and x_{gt}. Given threshold τ, a keypoint is an inlier if $e < \tau$, yielding two rewards.

Quantity reward. The first reward $r^{\text{qty}}_{p,k}$ evaluates the ability of each detector to produce a larger number of reliable keypoints within a patch, where p indexes the patch and $k \in \{1, \ldots, K\}$ denotes the detector class. We compute the number of inliers for each detector within each patch. Detectors

are then ranked according to the inlier counts within each patch, and a reward of 1 is assigned to the top-k detectors, while the remaining detectors receive 0.

Precision reward. The second reward $r^{\text{prec}}_{p,k}$ measures the precision of keypoints within a patch. For each detector, we compute the average error of its inlier matches within each patch, since detectors whose keypoints are more precisely localized on distinctive textures tend to produce more accurate correspondences in the target image. Detectors are then ranked by this average error in each patch, and a reward of 1 is assigned to the top-k detectors, while the remaining detectors receive 0.

The final reward combines both terms as a weighted sum:

$$r_{p,k} = \alpha r^{\text{qty}}_{p,k} + (1-\alpha) r^{\text{prec}}_{p,k}, \tag{1}$$

where α controls the trade-off between keypoint quantity and precision. Since optimal detector assignment depends on the target keypoint budget, rewards are computed for each budget index $m \in \{1, \dots, M\}$ using the top-B_m keypoints of each detector and normalized across detectors, where $(B_1, \dots, B_M) = (512, 1024, 2048, 4096, 8192)$. Specifically, for each patch and budget, the reward of each detector is divided by the sum of rewards over all detectors. Stacking all budgets yields $r \in \mathbb{R}^{K \times M \times h \times w}$, encoding detector preferences conditioned on local texture and keypoint budget.

3.2.3 Learning Objective

We train the router with a *routing loss* that favors high-reward detectors and a *background loss* that suppresses assignments in invalid regions.

Routing loss. Given the routing probabilities $P \in \mathbb{R}^{(K+1) \times M \times h \times w}$ defined in Section 3.2.1, we exclude its background class and compute a reward-guided cross-entropy loss

$$\mathcal{L}_{\text{router}} = -\frac{1}{MKhw} \sum_{m,k,p} r^m_{k,p} \log P^m_{k,p}, \tag{2}$$

encouraging higher probability for detectors with better rewards.

Background loss. We derive background supervision from the depth maps of MegaDepth Li and Snavely (2018), where unstable regions such as sky are assigned zero depth. We then compute the cross-entropy loss between the predicted background probabilities and background ground truth.

The final objective for stage 2 is defined as the sum of the two losses:

$$\mathcal{L} = \mathcal{L}_{\text{router}} + \mathcal{L}_{\text{background}}. \tag{3}$$

3.3 Keypoint Sampling

Since heterogeneous detector heads produce differently normalized scores, MoD avoids direct cross-head score comparison. After patch-wise routing, regions not assigned to each detector are masked out, and the total keypoint budget B is allocated proportionally to the number of patches assigned to each detector: $B_d = B \cdot N_d / N_{\text{non-background}}$, where N_d and $N_{\text{non-background}}$ denote the numbers of patches assigned to detector d and all non-background patches, respectively. Top-B_d keypoints are then selected independently from each detector's masked response map, and their union forms the final keypoint set. This strategy avoids cross-head score calibration while enabling adaptive keypoint allocation under a target keypoint budget.

4 Experiments

4.1 Implementation Details

In stage 1, the encoder is initialized with ImageNet pretraining and jointly optimized with all detection heads using AdamW Loshchilov and Hutter (2017). The learning rate is 1×10^{-4} with weight decay 1×10^{-5}, and the model is trained for 200,000 iterations with batch size 4. In stage 2, the router is trained with AdamW using a learning rate of 2×10^{-4} and weight decay 1×10^{-5}, for 20,000 iterations with batch size 4. Rewards are assigned to the top-2 detectors for both quantity and precision rewards. The error threshold is 1 pixel, and the trade-off parameter in the reward is $\alpha = 0.8$.

Table 2: **Pose Estimation and Homography Estimation using RoMa as the matcher.** Performance is measured as the AUC@5° of the pose error and the AUC@3 pixels of the end-point error. MoD† denotes the student detection head corresponding to the teacher detector. In each MoD† row, green means the student matches or outperforms its teacher. In the final MoD row, green means MoD matches or outperforms the best result among all teacher detectors and student heads.

Method	MegaDepth					ScanNet					HPatches				
	512	1024	2048	4096	8192	512	1024	2048	4096	8192	512	1024	2048	4096	8192
SP	64.6	65.4	65.7	65.4	64.3	28.4	28.7	28.6	28.9	28.6	58.6	59.4	59.1	59.4	58.9
MoD†	64.6	65.4	65.4	65.6	65.0	28.5	28.9	28.6	29.0	29.3	60.4	61.0	60.6	60.8	60.7
ALIKED	65.8	65.7	65.6	65.9	65.9	25.7	26.4	27.2	27.7	27.9	56.8	57.6	57.9	57.4	57.4
MoD†	66.3	66.6	66.4	66.7	65.3	26.3	27.1	27.3	27.8	28.1	59.9	59.9	59.7	59.6	59.6
DeDoDe	63.3	64.7	65.6	65.7	66.4	27.0	27.7	28.2	28.6	28.7	58.1	59.7	61.1	61.9	62.1
MoD†	63.6	65.3	65.8	66.1	66.4	27.1	27.8	28.3	28.6	28.9	58.8	60.2	61.3	61.9	62.1
DaD	66.3	67.2	67.6	67.8	66.0	28.7	29.0	29.1	29.1	29.0	60.5	61.1	61.0	60.8	60.6
MoD†	66.2	67.5	67.7	67.8	66.1	29.0	29.4	29.7	29.5	29.1	60.7	61.3	61.1	61.0	60.9
DeepLSD	57.3	60.8	63.1	65.0	63.7	18.2	22.5	26.4	28.2	27.5	48.6	52.3	55.6	59.2	59.3
MoD†	58.0	61.8	64.0	65.1	64.3	17.9	23.1	26.6	28.8	28.2	52.1	55.0	57.5	60.2	60.7
MuGE	59.6	63.1	64.1	63.5	63.5	19.7	23.0	25.8	27.4	28.7	51.6	53.5	56.6	58.3	59.6
MoD†	62.2	63.8	64.2	64.6	63.0	21.5	25.4	28.5	28.6	28.7	53.7	56.9	59.9	60.9	59.9
MoD	66.3	67.5	68.3	68.4	68.5	28.8	29.9	30.1	30.3	29.3	60.7	62.4	62.5	62.6	62.4

All models are trained on an NVIDIA RTX 4090 GPU. Following DaD Edstedt et al. (2025a), the longer image side is resized to 1024 during evaluation. We use RoMa v2 Edstedt et al. (2025b) as the matcher during training, and evaluate with both RoMa Edstedt et al. (2024c) and RCM+ Lu et al. (2026) to assess cross-matcher generalization and avoid overfitting to a single matcher.

4.2 Pose and Homography Estimation

Datasets. MegaDepth Li and Snavely (2018) contains one million Internet images spanning 196 outdoor scenes, with sparse 3D reconstructions from COLMAP and depth maps from multi-view stereo. We evaluate on 1,500 image pairs from the Sacre Coeur and St. Peter's Square scenes following LoFTR Sun et al. (2021). ScanNet Dai et al. (2017) contains 1613 monocular sequences with ground-truth poses and depth maps. We evaluate on 1,500 image pairs from SuperGlue Sarlin et al. (2020). The HPatches dataset Balntas et al. (2017) includes 52 illumination and 56 viewpoint sequences. Following DaD Edstedt et al. (2025a), we evaluate on the 56 viewpoint sequences with geometric transformations.

Metrics. For MegaDepth and ScanNet, we report the AUC of pose error at 5° under five keypoint budgets: 512, 1024, 2048, 4096, and 8192. For HPatches, we report the AUC of end-point error at 3 pixels under the same five keypoint budgets.

Results. As shown in Table 2, MoD†, which denotes the student detection head, outperforms the original detectors in most settings, indicating effective multi-teacher distillation. On MegaDepth, the distilled DeDoDe and DeepLSD heads match or outperform their teachers at all budgets, while SP, ALIKED, DaD, and MuGE match or outperform their teachers at four of the five budgets. Full MoD with router achieves the best performance at all budgets, reaching 66.3, 67.5, 68.3, 68.4, and 68.5. On ScanNet, MoD achieves the best performance at four budgets and, at 512, remains above the original DaD despite being lower than its student head. The larger margin suggests stronger benefits in indoor scenes. On HPatches, MoD† matches or improves upon the teachers in all cases, and full MoD achieves the highest AUC at all budgets, reaching 60.7, 62.4, 62.5, 62.6, and 62.4, consistently outperforming DaD. Across datasets, detector-wise trends are consistent, with DeDoDe improving at high budgets and DaD remaining competitive at low budgets, supporting budget-aware adaptive selection. Additional evaluations using RCM+ Lu et al. (2026) as the matcher are provided in the Appendix A.

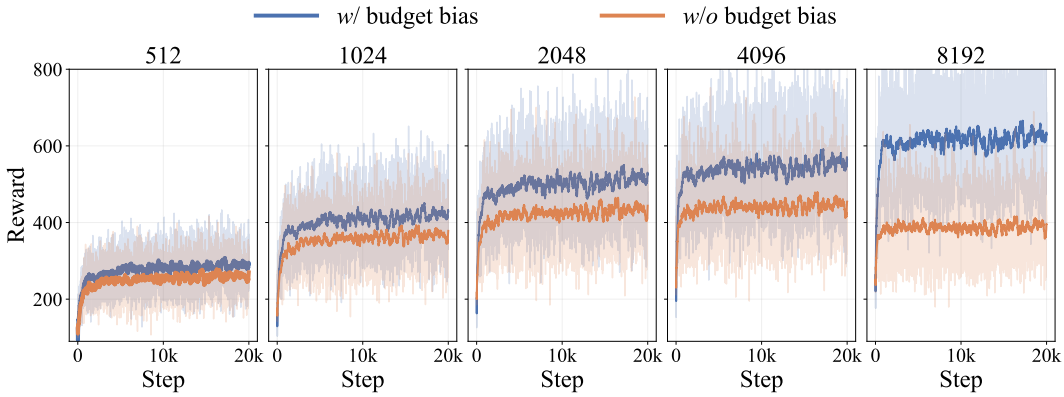

Figure 5: **Reward curves under different keypoint budgets with and without budget bias.** MoD without budget bias exhibits conflicting detector preferences across budgets.

4.3 Ablation Study

As shown in Table 3, we analyze the contribution of key design components in MoD, including the budget awareness, background modeling, routing strategy, and reward formulation.

Budget Awareness and Background Modeling. Removing the budget bias significantly degrades performance, particularly at higher keypoint budgets. This suggests that conditioning the router on the target keypoint count provides a useful global prior. A similar trend appears in Figure 5, which shows the training rewards of the router. Without budget awareness, the reward at 8192 is lower than at 4096, indicating conflicting routing decisions across budgets. In contrast, introducing budget awareness improves rewards across all settings, with larger gains at higher budgets.

Removing the background class also degrades performance at several budgets. This suggests that modeling non-matchable regions helps the router avoid low-quality areas, improving the quality and stability of routing decisions.

Table 3: **Ablation Study on HPatches.** Performance is measured by AUC@3 pixels of the end-point error.

Method	Homography Estimation AUC ↑				
	512	1024	2048	4096	8192
w/o budget bias	59.2	61.0	61.4	61.7	61.7
w/o background	59.5	61.1	61.9	61.6	**62.4**
avg. det. map	59.9	60.3	60.8	61.0	61.2
max det. map	60.0	60.4	61.0	60.6	60.8
global routing	60.2	60.5	61.2	61.2	61.5
soft routing	60.2	60.8	61.3	61.5	61.4
$\alpha = 0$	59.9	61.5	62.0	**62.6**	62.3
$\alpha = 1$	59.9	62.0	62.1	62.4	62.3
$\alpha = 0.5$	60.0	62.0	62.1	62.5	**62.4**
MoD	**60.7**	**62.4**	**62.5**	**62.6**	**62.4**

Routing Strategy. We investigate alternative routing and fusion strategies based on the six distilled detection maps. The avg. det. map and the max. det. map denote non-learnable fusion schemes that combine detection maps through pixel-wise averaging and maximum operations, respectively. After fusion, non-maximum suppression is applied, and the top-k responses are selected as keypoints. Both approaches consistently yield inferior performance across all keypoint budgets. We further evaluate learnable routing variants, including a global router that takes texture features as input and predicts M weights to perform image-level routing, and a soft router that uses patch-level routing probabilities to compute a weighted sum of detection maps instead of performing hard selection. These results indicate that the proposed router more effectively selects suitable detectors across different regions.

Reward Formulation. The effect of the reward formulation is examined via the trade-off parameter α, which balances the quantity and precision. Using either term alone yields suboptimal performance across most budgets. The balanced setting generally matches or improves upon the single-term variants. The best performance is achieved by the full model with $\alpha = 0.8$. This indicates that emphasizing keypoint quantity while accounting for precision yields more effective routing decisions.

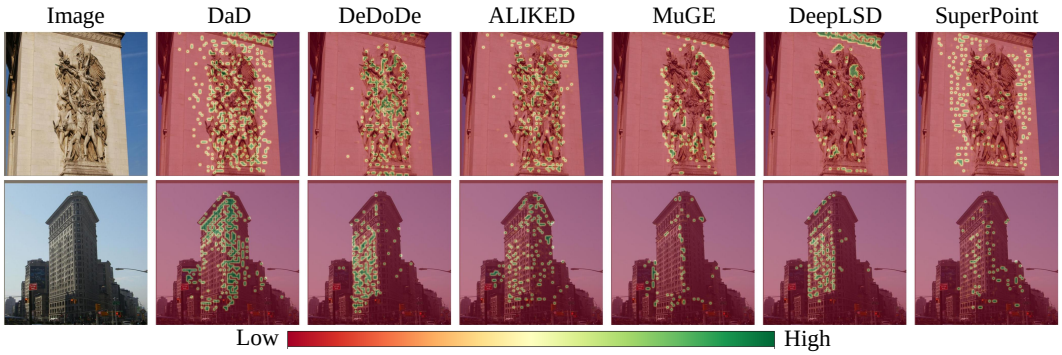

Figure 6: **Reward distributions for different detection methods** (best viewed when zoomed in).

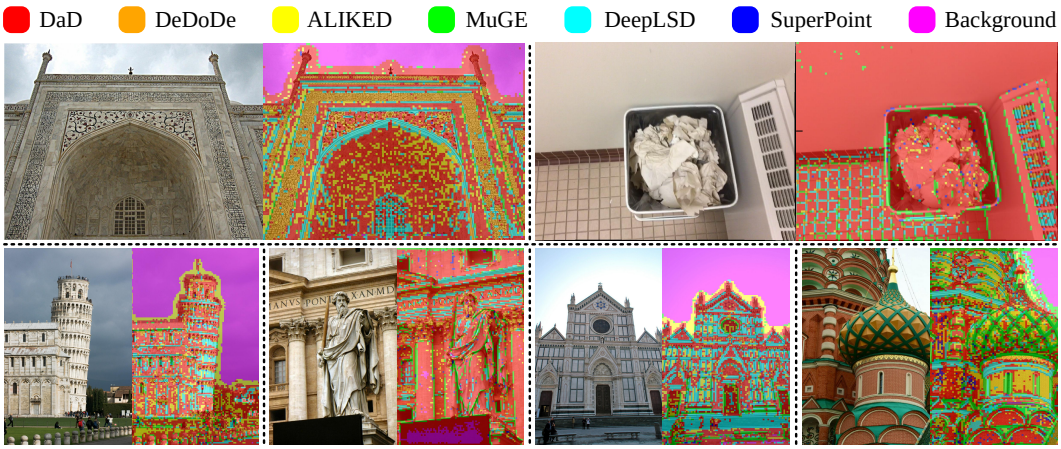

Figure 7: **Distribution of predicted routing decisions** (best viewed when zoomed in).

4.4 Understanding MoD

To gain deeper insights into the behavior of MoD, we visualize and analyze its key components.

Budget-aware bias. We visualize the learned budget biases for different detectors in Figure 4. Each row corresponds to a keypoint budget, and each column represents a detector. The learned biases exhibit clear and interpretable patterns across different budgets. At lower budgets, detectors such as DaD receive strong positive biases, while others are suppressed, indicating a preference for methods that produce sparse and reliable keypoints. As the budget increases, the preference gradually shifts toward detectors such as MuGE, suggesting that edge-based detectors become more advantageous when more keypoints are allowed. At the highest budget, the preference changes noticeably, where detectors such as DeDoDe and SuperPoint receive positive biases, while MuGE is strongly suppressed, reflecting different behaviors under dense keypoint regimes.

Reward. We visualize the reward distributions in Figure 6, where brighter regions indicate higher rewards. The reward maps exhibit clear spatial structure, revealing which detector performs best at each location. DaD and DeDoDe are favored in richly textured regions, while line- and edge-based detectors dominate along structural boundaries. ALIKED and SuperPoint produce more balanced responses in moderately textured areas. Different detectors dominate in different regions of the same image, indicating that no single detector is universally optimal.

Routing decisions. The predicted routing decisions are shown in Figure 7, where each color corresponds to a selected detector or the background class. The routing maps exhibit strong spatial coherence, with detector assignments varying systematically across different regions. Point-based detectors are predominantly selected in textured areas, whereas line- and edge-based detectors are activated along prominent geometric structures. Regions with insufficient texture or unreliable geom-

Table 5: **Teacher–Student Detection Map Correlation.** Pearson correlation coefficients between teacher detectors (rows) and distilled student heads (columns), averaged over 1500 MegaDepth images.

Teacher / Student	DaD	DeDoDe	ALIKED	MuGE	DeepLSD	SP
DaD	**0.9683**	0.5689	0.2075	0.3924	0.1255	0.2900
DeDoDe	0.5746	**0.9775**	0.1338	0.6035	0.2169	0.3277
ALIKED	0.2052	0.1413	**0.9604**	0.1221	0.1171	0.2651
MuGE	0.3621	0.5718	0.1174	**0.9178**	0.3539	0.3760
DeepLSD	0.1186	0.2211	0.1125	0.3639	**0.9239**	0.1928
SP	0.2877	0.3301	0.2373	0.3878	0.2261	**0.8882**

etry are assigned to the background class, avoiding ineffective keypoints. Additional visualizations of routing decisions are provided in Appendix C and D.

Efficiency analysis. The efficiency comparison is reported in Table 4. MoD maintains practical inference time while producing diverse detection maps and adaptive routing, achieving efficient mixture of detectors without running all candidate detectors separately.

Table 4: **Efficiency Analysis.**

Method	Param. (M)	Time (ms)
SP	0.9	16.3
ALIKED	0.4	17.2
DeDoDe	14.6	65.8
DaD	6.5	27.8
DeepLSD	8.6	45.7
MuGE	72.5	58.7
MoD	9.4	37.4

Preservation of detector-specific behaviors. To quantitatively assess whether multi-teacher distillation preserves detector-specific spatial preferences, we compute the Pearson correlation coefficients between the detection maps of the six teacher detectors and their corresponding student heads. For each teacher–student pair, we flatten their detection logit maps, calculate the Pearson correlation coefficient, and average the results over 1500 images.

As shown in Table 5, the diagonal correlations range from 0.8882 to 0.9775, indicating that each distilled student head closely reproduces the spatial response patterns of its corresponding teacher. In contrast, the substantially lower off-diagonal correlations indicate that different student heads retain distinguishable detector-specific preferences rather than collapsing into similar detection maps. These results provide direct evidence that multi-teacher distillation preserves both teacher–student fidelity and the diversity required for effective adaptive detector routing.

5 Limitations and Discussion

We investigate the use of pose estimation error as an additional reward signal for router training. However, this strategy does not yield observable improvements over the current design. This observation suggests that directly using downstream task errors as supervision is challenging, likely due to the indirect and noisy relationship between detector selection and final task performance. Future work may explore more principled approaches for incorporating downstream objectives, such as homography or pose estimation accuracy, into the reward formulation through more stable and structured supervision mechanisms.

6 Conclusion

We presented MoD, a unified framework that integrates multiple heterogeneous keypoint detectors through knowledge distillation and budget-aware routing. By distilling diverse detectors into a shared encoder with multiple lightweight heads, MoD preserves their complementary inductive biases while achieving efficient inference within a single forward pass. Building on this, the proposed router adaptively selects detectors at the patch level by jointly considering local texture characteristics and global keypoint budget, and is trained via a reward signal that captures both keypoint quantity and precision. Extensive experiments on MegaDepth, ScanNet, and HPatches demonstrate that MoD consistently outperforms strong baselines across a wide range of keypoint budgets, validating the effectiveness of both the distillation and routing components.

References

Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 5173–5182, 2017.

John Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 8(6):679–698, 1986.

Tianlong Chen, Xuxi Chen, Xianzhi Du, Abdullah Rashwan, Fan Yang, Huizhong Chen, Zhangyang Wang, and Yeqing Li. AdaMV-MoE: Adaptive multi-task vision mixture-of-experts. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 17346–17357, 2023.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 5828–5839, 2017.

Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 224–236, 2018.

Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2019.

Johan Edstedt, Ioannis Athanasiadis, Mårten Wadenbäck, and Michael Felsberg. DKM: Dense kernelized feature matching for geometry estimation. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2023.

Johan Edstedt, Georg Bökman, and Zhenjun Zhao. DeDoDe v2: Analyzing and improving the DeDoDe keypoint detector. In *IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4245–4253, 2024a.

Johan Edstedt, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. DeDoDe: Detect, Don't Describe – Describe, Don't Detect for Local Feature Matching. In *Int'l Conf. 3D Vision (3DV)*. IEEE, 2024b.

Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. RoMa: Robust dense feature matching. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2024c.

Johan Edstedt, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. DaD: Distilled reinforcement learning for diverse keypoint detection. *arXiv preprint arXiv:2503.07347*, 2025a.

Johan Edstedt, David Nordström, Yushan Zhang, Georg Bökman, Jonathan Astermark, Viktor Larsson, Anders Heyden, Fredrik Kahl, Mårten Wadenbäck, and Michael Felsberg. RoMa v2: Harder better faster denser feature matching. *arXiv preprint arXiv:2511.15706*, 2025b.

J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 40(3):611–625, 2017.

Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel, and Gregory Randall. LSD: a fast line segment detector with a false detection control. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 32(4):722–732, 2010.

Zhengqi Li and Noah Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2041–2050, 2018.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

David G Lowe. Distinctive image features from scale-invariant keypoints. *Int'l J. Computer Vision (IJCV)*, 60: 91–110, 2004.

Xiaoyong Lu, Songlin Du, Yaping Yan, Xiaobo Lu, and Takeshi Ikenaga. Toward free-form local feature matching. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 48(2):1470–1484, 2026.

Remi Pautrat, Juan-Ting Lin, Viktor Larsson, Martin R. Oswald, and Marc Pollefeys. SOLD2: Self-supervised occlusion-aware line description and detection. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 11368–11378, 2021.

Rémi Pautrat, Daniel Barath, Viktor Larsson, Martin R Oswald, and Marc Pollefeys. DeepLSD: Line segment detection and refinement with deep image gradients. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 17327–17336, 2023.

Mengyang Pu, Yaping Huang, Yuming Liu, Qingji Guan, and Haibin Ling. EDTER: Edge detection with transformer. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1402–1412, 2022.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:8583–8595, 2021.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2020.

J. L Schonberger and J. Frahm. Structure-from-motion revisited. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Int'l Conf. Learning Representations (ICLR)*, 2015.

Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 8922–8931, 2021.

Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. DISK: Learning local features with policy gradient. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:14254–14265, 2020.

Xiaoming Zhao, Xingming Wu, Weihai Chen, Peter CY Chen, Qingsong Xu, and Zhengguo Li. ALIKED: A lighter keypoint and descriptor extraction network via deformable transformation. *IEEE Transactions on Instrumentation and Measurement (T-IM)*, 72:1–16, 2023.

Caixia Zhou, Yaping Huang, Mengyang Pu, Qingji Guan, Ruoxi Deng, and Haibin Ling. MuGE: Multiple granularity edge detection. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 25952–25962, 2024.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:7103–7114, 2022.

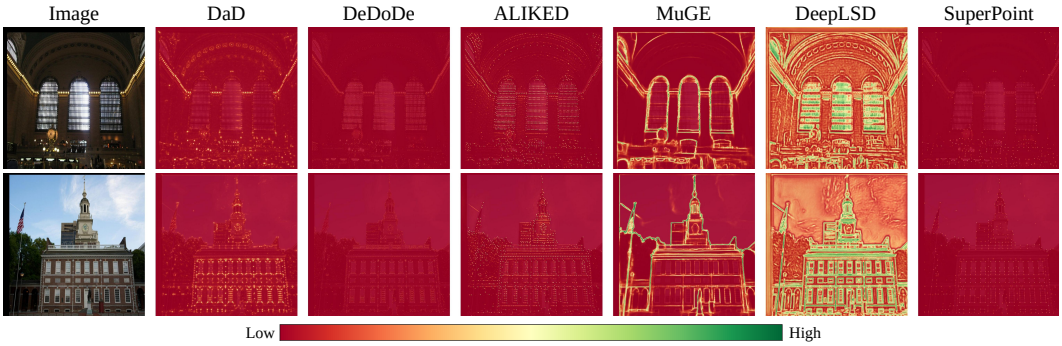

Figure 8: **The detection probability maps predicted by the distilled detection head** (best viewed when zoomed in).

A Evaluation with RCM+

To further assess generalization across different matching pipelines, we additionally evaluate all methods using RCM+ as the matcher, as shown in Table 6. MoD consistently achieves the best performance across all keypoint budgets. The performance gains over strong baselines such as DaD and DeDoDe remain consistent, indicating that the advantages of MoD are not tied to a specific matcher. Notably, the relative trends among detectors are largely preserved, further confirming that the proposed routing mechanism captures intrinsic detector complementarity rather than overfitting to a particular matching strategy.

Table 6: **Homography Estimation on HPatches using RCM+ Lu et al. (2026) as the matcher.** Performance is measured as the AUC@3 pixels of the end-point error.

Method	Homography Estimation AUC ↑				
	512	1024	2048	4096	8192
DeepLSD	43.9	47.9	49.8	50.6	51.1
Canny	34.1	40.2	46.8	50.7	51.3
UAED	41.2	45.4	48.1	49.7	51.5
MuGE	42.7	46.1	49.3	50.5	51.9
SP	48.9	51.1	51.7	52.0	51.6
DISK	41.0	44.2	46.4	47.8	49.0
ALIKED	44.8	46.9	47.8	48.9	49.0
DeDoDe	49.4	50.5	51.6	52.4	52.5
DaD	49.9	51.2	52.0	51.8	51.6
MoD	**50.4**	**51.8**	**52.2**	**53.1**	**53.3**

B Distilled Detection Maps

We visualize the detection probability maps produced by the distilled heads in Figure 8. Each head exhibits a distinct response pattern that closely matches its corresponding teacher. In particular, point-based detectors produce sparse and localized activations around salient keypoints, whereas line- and edge-oriented detectors (e.g., DeepLSD and MuGE) strongly emphasize geometric structures such as edges and contours. These results demonstrate that the distillation process effectively preserves the heterogeneous inductive biases of different detectors within a unified architecture. Consequently, the distilled heads maintain complementary behaviors, providing a diverse set of candidates for subsequent selection.

C Distribution of predicted routing decisions

We visualize the distribution of predicted routing decisions with a keypoint budget of 2048 in Figure 9. Each image illustrates how the patch-wise router assigns detectors across diverse scenes, with colors representing different detectors and the background class. The results reveal that the router effectively adapts to varying local textures and scene structures, consistently assigning detectors to regions where they are most effective. Notably, the allocation patterns reflect detector complementarity: some detectors dominate structured or repetitive textures, while others are preferentially selected for edges, corners, or high-detail regions. Background regions are largely assigned to the background class, indicating that the router successfully avoids assigning detectors to unstable or uninformative areas. These visualizations confirm that the routing decisions are spatially structured and align with the complementary strengths of the underlying detectors, providing interpretability for the learned adaptive allocation mechanism.

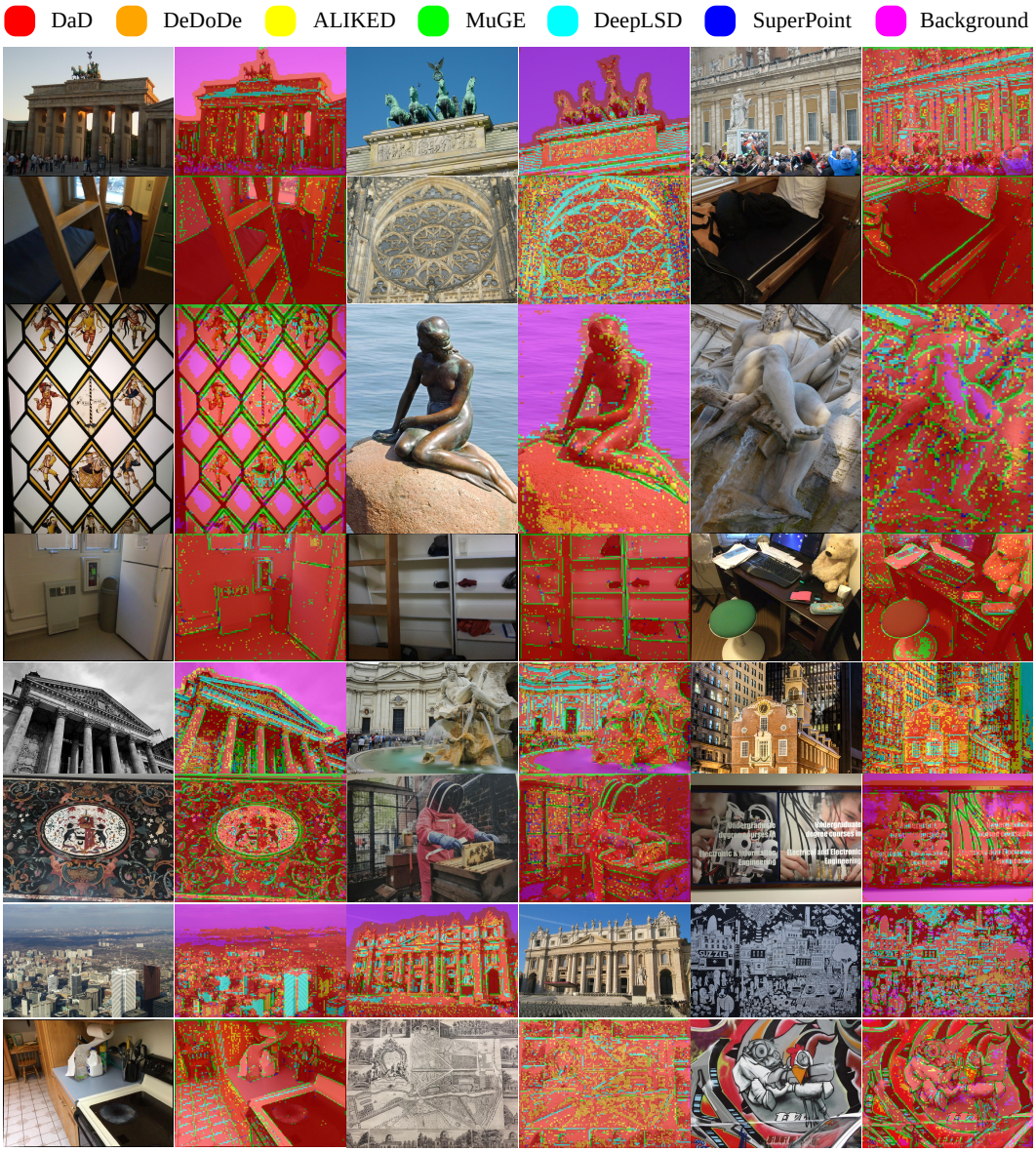

Figure 9: **Distribution of predicted routing decisions** (best viewed when zoomed in).

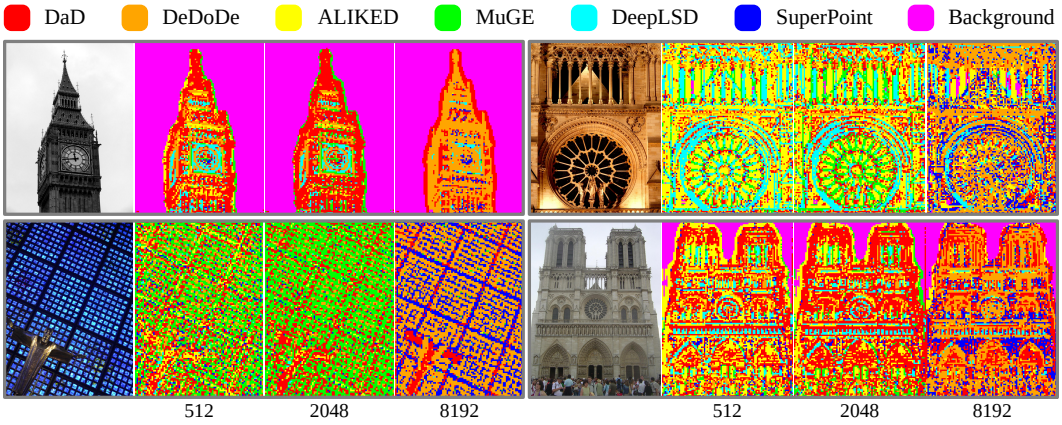

Figure 10: **Routing decisions under varying keypoint budgets** (best viewed when zoomed in).

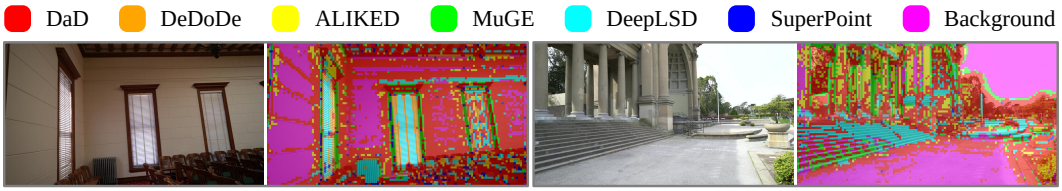

Figure 11: **Failure cases of the router** (best viewed when zoomed in).

D Routing decisions under varying keypoint budgets

We further analyze the effect of keypoint budget in Figure 10. As the budget increases, the routing patterns change noticeably. At lower budgets, the router favors detectors that produce sparse and reliable keypoints such as ALIKED and DaD. In contrast, at higher budgets, the router increases activation of detectors that can generate a large number of valid keypoints in highly textured regions, such as DeDoDe. This behavior reflects the model's ability to adapt detector selection according to the desired keypoint density.

E Failure Cases

While MoD generally produces robust keypoint distributions, some texture-less regions, such as floors or walls, are occasionally classified as background despite containing useful keypoints. Figure 11 shows examples of such misclassifications. Analyzing these cases highlights limitations of the current reward and routing mechanisms and suggests potential improvements, such as incorporating additional context or uncertainty-aware routing.

F Keypoint Distribution

We visualize the final keypoint distributions predicted by the seven detectors on MegaDepth, ScanNet, and HPatches, with a unified keypoint budget of 2048, as shown in Figure 12. The visualizations reveal that MoD produces a more balanced and spatially diverse distribution compared to individual detectors. While single detectors tend to concentrate keypoints in specific texture patterns or edges, MoD effectively integrates the complementary strengths of all detectors, covering both high-texture and low-texture regions across scenes. This results in a more uniform keypoint coverage, which aligns with the adaptive routing decisions and contributes to improved downstream geometric performance. The figures also highlight that detectors specialize in different scene components, reinforcing the importance of detector complementarity in generating robust keypoint distributions.

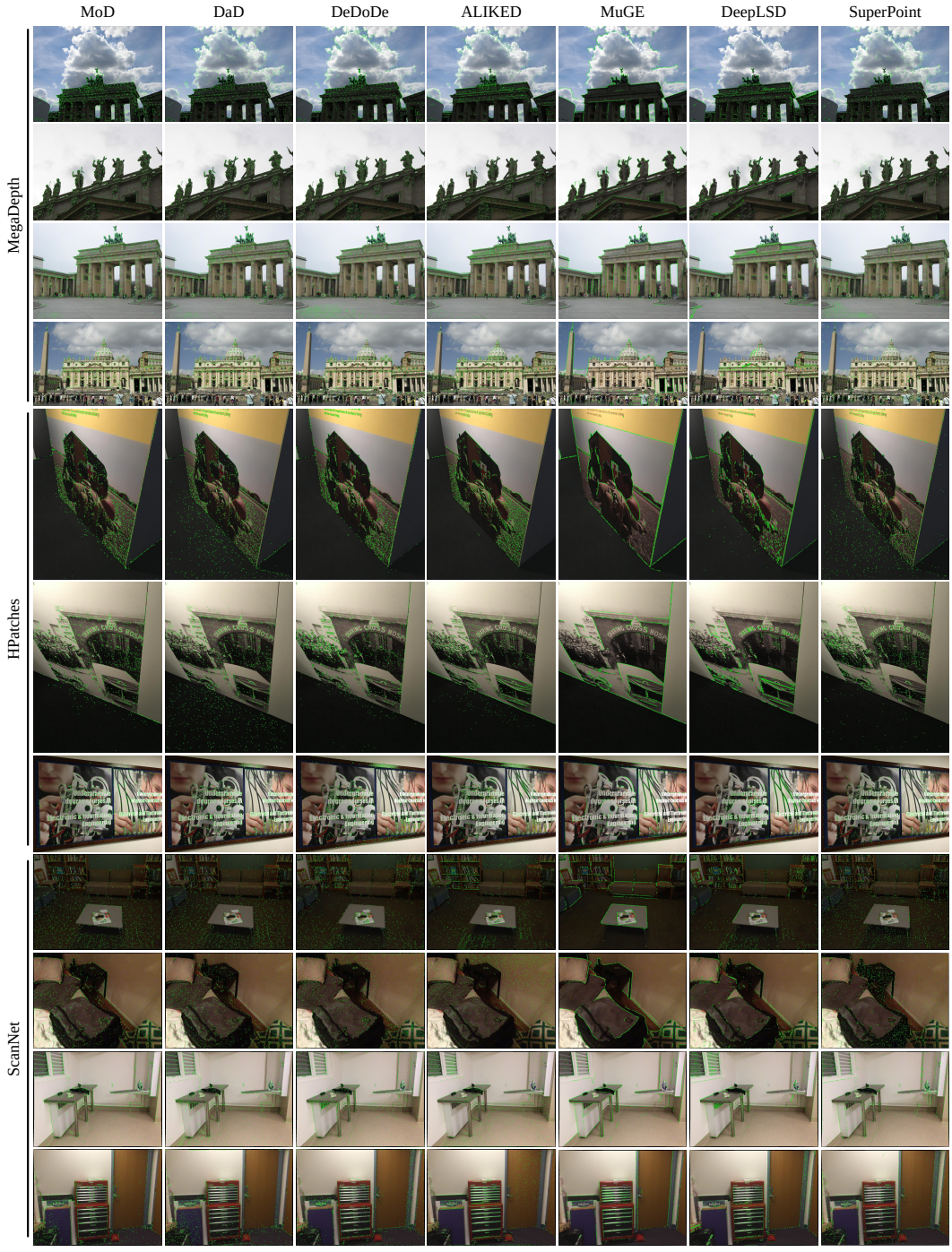

Figure 12: **Keypoint visualization** (best viewed when zoomed in).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly present the core contributions of the paper, which are subsequently supported by experimental results in the main body.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our model in the Limitations and Discussion section (Section 5).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not contain formal theoretical results or proofs. The focus is on the design, implementation, and evaluation of the proposed method.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides sufficient implementation details to reproduce the main experimental results, including model architecture, training settings, datasets, evaluation metrics.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code will be released to facilitate reproducibility.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides comprehensive details on the experimental setup in Section 4.1, including dataset splits, hyperparameter settings and optimizer type.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We use well-established metrics for all experiments in this paper. These metrics are calculated by averaging over all samples in the test set.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details of the computational resources and runtime are provided in Section 4.1 and Table 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the Code of Ethics and confirm that our research conforms with it.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This work primarily focuses on detection improvements. There are currently no direct or significant societal impacts associated with the research. Therefore, no discussion on positive or negative societal impacts is included.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not involve the release of models or datasets that pose a high risk of misuse. Therefore, no specific safeguards are necessary.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use publicly available datasets and pretrained models, all of which are properly cited in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Although the assets are not released at submission time, we plan to release them upon acceptance. The released assets will be properly documented, including usage instructions, license information, and known limitations. No new datasets are introduced in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing experiments or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.