# Doodle Your Keypoints: Sketch-Based Few-Shot Keypoint Detection
Supplementary Material

## 6. Insights on Framework Design

### 6.1. Transport Loss for Keypoint Localization

**Background:** Prototypical domain adaptation [115] is particularly curated for classification that differs from keypoint detection [67] significantly at the task level. Thus, to adapt the same for the sketch-photo domain gap, we needed to analyze every piece of the mechanics it is principled on. The bi-directional transport loss [115] precisely employs two transport objectives: one to pull the samples in the target domain towards the source domain prototype, and the other to pull the prototype towards respective targets. Taking the unsupervised setting into account, the source prototype is considered to be an overall class prototype that encompasses both the source and target domains.

It is worth noting that the prototype formulation in prototypical domain adaptation [115] is particularly dependent on the *weights of the last linear layer* responsible for performing classification, while the conventional prototype formulation [108] uses an *mean of the vector representations* in the latent space. Although in both cases, prototypes are essentially built in the same latent embedding, the metric relation between the prototype and the source embedding is completely disparate. Likewise, the original prototypical network [108] can be used with any distance metric, while the former [115] is restricted to element-wise multiplication between source embeddings and prototypes or weights.

**Dissecting Bidirectional Transport Loss:** Considering $\vartheta_j(\alpha_j) = \frac{(\alpha_j)}{\sum(\alpha_j)}$ as a normalized weighing function, the bidirectional transport loss [115] is given as:

$$\begin{aligned}\mathcal{L}_{t\to\mu}+&\mathcal{L}_{\mu\to t} = \\ &\mathbb{E}[\sum_n \mathbf{C}(\mu_n, \hat{\Phi}_{m,n}) \cdot \vartheta_n(p(\mu_n)\exp(\mu_n \hat{\Phi}^{\mathbf{T}}_{m,n}))] + \\ &\mathbb{E}[\sum_n p(\mu_n) \sum_m \mathbf{C}(\mu_n, \hat{\Phi}_{m,n}) \vartheta_m(\exp(\mu_n \hat{\Phi}^{\mathbf{T}}_{m,n}))]\end{aligned} \tag{12}$$

Here, the terms $\mathcal{L}_{t\to\mu}$ and $\mathcal{L}_{\mu\to t}$ signify two directions, target domain to prototypes and prototypes to target domain, respectively, of the bi-directional aspect of the loss function in Eq. (12). The term $\mathbf{C}(\mu_n, \hat{\Phi}_{m,n})$ refers to a direct *point-to-point* cost between a prototype and the corresponding target embeddings and implements a *cosine distance* between prototype $\mu_n$ and respective $m$th target embedding $\hat{\Phi}_{m,n}$ for any class $n$. Also, the expansion of $\mathcal{L}_{\mu\to t}$ term uses the normalized weighing function $\vartheta$ to mimic `softmax` operation implicitly, and the expansion of $\mathcal{L}_{t\to\mu}$ uses the `softmax` function and directly provides a discriminative entropy minimization linking to the classification task regime. Moreover, this is especially aligned with the corresponding prototype formulation with weights of the last layer.

As the keypoint detection does not need to predict discriminative probabilities between different classes or keypoints, we remove the weighing function $\vartheta$ and Eq. (12) reduces to:

$$\begin{aligned}\mathcal{L}_{t\to\mu} + \mathcal{L}_{\mu\to t} = &\mathbb{E}[\sum_n \mathbf{C}(\mu_n, \hat{\Phi}_{m,n}) p(\hat{\mu}_n)\exp(\mu_n \hat{\Phi}^{\mathbf{T}}_{m,n})] + \\ &\mathbb{E}[\sum_n p(\hat{\mu}_n) \sum_m \mathbf{C}(\mu_n, \hat{\Phi}_{m,n})\exp(\mu_n \hat{\Phi}^{\mathbf{T}}_{m,n})]\end{aligned} \tag{13}$$

In Eq. (13), the term $\exp(\mu_n \hat{\Phi}^{\mathbf{T}}_{m,n})$ is essentially a *similarity measure* that closely resembles the cosine similarity. Taking **Sim** as a similarity function, Eq. (13) can be further expressed as Eq. (14), identifying the key factors responsible for the domain adaptation in the transport loss.

$$\begin{aligned}\mathcal{L}_{t\to\mu} + \mathcal{L}_{\mu\to t} = &\mathbb{E}[\sum_n \mathbf{C}(\mu_n, \hat{\Phi}_{m,n}) p(\hat{\mu}_n)\mathbf{Sim}(\mu_n, \hat{\Phi}_{m,n})] \\ &+ \mathbb{E}[\sum_n p(\hat{\mu}_n) \sum_m \mathbf{C}(\mu_n, \hat{\Phi}_{m,n})\mathbf{Sim}(\mu_n, \hat{\Phi}_{m,n})]\end{aligned} \tag{14}$$

**Adaptation to Keypoints Learning Paradigm:** From Eq. (14), it is clear that both the loss terms, $\mathcal{L}_{t\to\mu}$ and $\mathcal{L}_{\mu\to t}$ are reduced to the similar form consisting of target prototype $p(\hat{\mu}_n)$, a point-to-point cost $\mathbf{C}(\mu_n, \hat{\Phi}_{m,n})$ and a similarity score $\mathbf{Sim}(\mu_n, \hat{\Phi}_{m,n})$. Thus, having a similar form of both transport losses [115], the domain adaptation loss $\mathcal{L}_{\text{DA}}$ in Eq. (10) takes a unified form of the transport losses. Considering the aforementioned inherent differences at the task level, $\mathbf{C}(\mu_n, \hat{\Phi}_{m,n})$ and $\mathbf{Sim}(\mu_n, \hat{\Phi}_{m,n})$ are realized by $l_2$ distances and a derived similarity score for the keypoint learning task as mentioned in details in Sec. 3.2.

It is also to be noticed that $\hat{\mu}_n$ is used in Eqs. (13) and (14) to represent the target prototype. Due to the absence of class information of the target domain, Tanwisuth *et al.* [115] uses a single prototype system for source and target domains. We consider the gradual movement of prototypes from the source to the target domain, and thus we replace the same with a prototype calculation for our target domain, *i.e.* query keypoints using Eq. (9). The term $p(\mu_n)$ essentially represents the *probability* of the prototypes given all the target samples. Given the task-level setting of unsupervised classification, $p(\mu_n)$ is interpreted as class proportion in the original bi-directional transport loss [115] and is iteratively updated starting from a uniform class distribution. On the contrary, keypoints learning replaces it by an equivalent term $p(\hat{\mu}_n)$ which refers to $l_2$ distance-based probability for the prototypes (Eq. (9)) and it could be calculated dynamically using the keypoint-level class information of the query photos, turning the problem setup to a supervised one and dismissing the necessity of iterative updates of the prototype likelihood.

## 6.2. Design Specifications of Descriptor Network

The descriptor network [67] $D$ is particularly employed to *refine and encode* the features at the local scale from the correlated query feature maps $\mathcal{A}_{m,n}$ having dense encoded features pertaining to both the query feature map $f_m$ and the support prototype $\mu_n$, to a descriptor $\Psi_{m,n}$ so that it contains necessary positional information to localize the relevant keypoint $n$ in query photo $x_m$.

The architecture of the descriptor network $D$ is taken from FSKD [67], and the design specification is kept the same as well. The network $D$ consists of three consecutive convolution layers with kernel size $3 \times 3$, a stride of 2, a padding of 1, and `ReLU` as activation. The input channels for the first convolution layer are $c = 2048$, the output channels of the last layer are 1024, and all the intermediate input or output channels are 512. Considering the input size of $x_i$ being $384 \times 384$ we have correlated query features $f_m$ of size $\mathbb{R}^{2048\times12\times12}$ as input of descriptor network $D$ which results in tensors of $\mathbb{R}^{512\times6\times6}$, $\mathbb{R}^{512\times3\times3}$ and $\mathbb{R}^{1024\times2\times2}$ as the consecutive outputs of convolution layers. Thus, the final output descriptor $\hat{\Psi}_{m,n}$ of dimension $d = 4096$ is formed by flattening the last output from $D$.

## 6.3. Architecture Choice for De-stylization Network

**Background:** The de-stylization network $Z$ as per the architecture given in Fig. 3 is designed after the multi-scale channel attention module [25] to fuse the keypoint level local information with the global context of sparsity and style present in a sketch or edgemap. While multi-scale channel attention [25] is specifically curated for convolutional feature maps, our design needs to deal with keypoint embeddings of $\mathbb{R}^c$, keeping a similar notion of context fusion [25]. The original design [25] uses two parallel branches on convolutional feature maps, one with the input feature map as it is, and the other using a globally pooled vector from the input convolutional feature map, encoded with separate learnable convolution layers and both the branches are aggregated using an element-wise addition for fusing the global context into feature maps. The resulting convolutional map is then passed through `sigmoid` activation to adjust the weight to which the fused context should affect the original input map while it is multiplied with the fused feature map.

**Proposed Design:** Designing our de-stylization network $Z$ (see Fig. 3), we need it to cater to the keypoint embeddings of $\mathbb{R}^c$ with dense local features, and thus they are fused with globally pooled vectors from the corresponding feature map $f_k$. The context fusion for any keypoint $n$ in $x_k$ is achieved by the concatenation of the extracted keypoint embedding $\Phi_{k,n}$ and the corresponding global pooled features from $f_k$, followed by two linear layers with a `ReLU` activation in between. This, in particular, is used as a context at both *local* and *global* scales and is added element-wise to $\Phi_{k,n}$ and a `sigmoid` activation, along with two more linear layers with a `ReLU` activations in between help in learning keypoint embeddings $\delta_{k,n}$ from the dense fused features at local and global scales.

| Architecture of $Z$ | Local Context | Global Context | PCK@0.1 |
|---|---|---|---|
| B-DA (No $\mathcal{L}_{\text{style}}$) | ✓ | ✗ | 31.76 |
| None (Identity) | ✓ | ✗ | 36.84 |
| MLP | ✓ | ✗ | 37.78 |
| MLP (Concatenated) | ✓ | ✓ | 38.11 |
| **Proposed** | ✓ | ✓ | **39.00** |

Table 3. A comparison of performance for different architecture designs of de-stylization network $Z$ along with the usage of local and global contexts. The PCK@0.1 is measured on the Animal Pose dataset [15] for novel keypoints on unknown classes.

# 7. Design Analysis of De-stylization Network

The de-stylization network $Z$ disentangles style and sparsity from the keypoint embeddings using *attentional* global and local *context fusion* [25]. However, to understand the role of global context realized by the global pooled vector from the support feature $f_k$ and also to justify our choice of architecture, we designed a few different architectures of $Z$ and rigorously experimented and evaluated for novel keypoints on unseen classes (Tab. 3).

*(a)* The baseline B-DA is used as a control measure as there is no $Z$ or $\mathcal{L}_{\text{style}}$ involved. *(b)* Using an identity function as $Z$, we essentially ensure $\Phi_{k,n} = \delta_{k,n}$. However, the framework still tries to learn style-agnostic keypoint embeddings using the style loss $\mathcal{L}_{\text{style}}$ with additional edgemaps, which have a significant performance upgradation ($\uparrow 5.08$) over B-DA. This proves that the loss $\mathcal{L}_{\text{style}}$ contributes significantly and thus, the assumption regarding existing style diversity in the synthetic sketches or edgemaps is validated. *(c)* Using a multi-layer perceptron (MLP) as an alternative architecture of $Z$ is taken into consideration. While the MLP takes only extracted keypoint embedding $\Phi_{k,n}$ as input, it has an improvement of 0.94 over the case where $\Phi_{k,n} = \delta_{k,n}$. This signifies the learning of better keypoints with $Z$ while preserving *alignment* between the support keypoint and the query. *(d)* Using the MLP designed to take the dense features, formulated by concatenation of global pooled features and the extracted keypoint embeddings, the performance gets a further boost of 0.33, which proves that global context has a significant role in disentanglement and can encode the style and sparsity information. *(e)* The proposed design architecture has the best performance with an improvement of 0.89 over MLP with a global context aggregation. *(f)* Apart from the architectures presented in the Tab. 3, we also experimented with several other attention modules. However, all such models had *intractable* training with exploding gradients and loss values greater than $10^8$ times the usual.

| Edge Detector | Animal Pose [15] | Animal Kingdom [79] |
|---|---|---|
| [60] + [109] + [129] | 36.87 | 12.46 |
| [129] + [56] + [14] | 37.33 | 12.98 |
| [14] + [60] + [109] | 38.38 | 13.45 |
| [110] + [60] + [56] | 38.61 | 13.87 |
| **Ours** ([110] + [129] + [14]) | **39.00** | **14.42** |

Table 4. Performance of the proposed framework with various edge detection algorithms used for generating synthetic sketch data for training. Performances (PCK@0.1) are measured on both datasets [15, 79] for novel keypoints on unseen species.

## 8. Choosing Edge Detection Algorithms

The choice of edge detection algorithm is particularly crucial for our problem, as it directly connects with the training data for the proposed framework. As we treat edgemaps as *synthetic* sketch data, we performed in-depth experimentation with the different edge detection algorithms as mentioned in Sec. 4.4. Precisely, we include modern and popular edge detector algorithms, including Im2pencil [60], DeXiNed [109], Photo-Sketch [56], HED [129], Canny [14], PiDiNet [110] in multiple combinations. The top 5 performing combinations on both datasets [15, 79] for the most challenging evaluation setting (*novel* keypoints on *unseen* classes) are given in Tab. 4. While the proposed combination performs the best, the presented data shows that the performance does not vary by more than 3 PCK@0.1 for both datasets [15, 79], essentially indicating that the proposed framework is robust enough to accommodate different levels of style and sparsity in sketch data. Moreover, we have experimented with real photos in the support set instead of sketches or edgemaps, achieving a similar accuracy of 43.72% (only a gain of ↑ 4.72 over sketches) on the Animal Pose dataset [15] for novel keypoints on query photos of unseen classes. See Secs. 4.3 and 10 for detailed results and analysis. Thus, it could be argued that the proposed method is *robust* and capable enough to *encode keypoint-level information* across diverse styles of sketches as well as photos. While we understand the capability of encoding synthetic sketches, the following section (Sec. 9) illustrates the practicality aspect of using synthetic sketch data or edgemaps.

## 9. Empirical Study with Free-Hand Sketches

The proposed framework is entirely trained with edgemaps or synthetic sketches as given in Secs. 3 and 4, and a few sample visualizations of support edgemap [110] and detection with ground-truth on query photos [15] are given in Fig. 10. Thus, we perform extended experimentations with pre-trained models as described in Sec. 4.2 along with a sample depiction of base and novel keypoints on support sketches [104] and query photos [15] in Fig. 6. From the data presented in Tab. 5, it is evident that our framework has *adequate* few-shot capability for generalizing across real support sketches, as PCK with $\tau = 0.1$ for real sketches is within a range of 5 from the extensive evaluation results on edgemaps. More visualizations for base and novel keypoints are in Fig. 11.

| Class | Keypoints | Support | PCK@0.1 | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Cat | Cow | Dog | Horse | Sheep | **Mean** |
| Seen | Base | Edgemap | 67.34 | 49.89 | 56.28 | 56.35 | 45.65 | 55.10 |
| | | Sketch | 66.69 | 45.79 | 55.43 | 56.13 | 43.40 | 53.29 |
| | Novel | Edgemap | 55.69 | 43.09 | 46.58 | 43.94 | 36.39 | 45.14 |
| | | Sketch | 55.45 | 42.96 | 46.35 | 43.88 | 36.31 | 44.99 |
| Unseen | Base | Edgemap | 47.36 | 42.97 | 38.30 | 46.17 | 41.03 | 43.17 |
| | | Sketch | 45.90 | 42.47 | 37.82 | 45.36 | 40.45 | 42.40 |
| | Novel | Edgemap | 44.42 | 40.13 | 36.91 | 37.77 | 35.77 | 39.00 |
| | | Sketch | 43.79 | 39.91 | 36.17 | 37.56 | 35.02 | 38.49 |

Table 5. A quantitative comparison of the proposed method on query photos [15] using edgemaps and real free-hand sketches [104] with $K = 1$ for all evaluation settings.

## 10. Experiments with Support Photos

Apart from using sketches or edgemaps as support, we also experiment with photos as support, solving the simple few-shot keypoint detection problem Lu *et al*. [67] solves. This experimentation was conducted to prove the robustness of the proposed work. While photos do not have any style or abstraction diversity, we first experiment with only photos in a setting similar to the original work of FSKD [67]. In this setting, we completely turn off the de-stylization loss $\mathcal{L}_{\text{style}}$ (Sec. 3.3) due to lack of additional support inputs. However, the de-stylization network $Z$ being an integral part of the architecture keeps aiding the learning of deeper features. Empirically, although our method goes close to the state-of-the-art FSKD [67], it fails to outperform. Next, we carefully devise the experimentation strategy using edgemaps [110, 129] as additional sketches turn-

| Class | Keypoints | Method | PCK@0.1 | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Cat | Cow | Dog | Horse | Sheep | **Mean** |
| Seen | Base | FSKD [67] | 68.66 | 52.70 | 59.24 | 58.53 | 45.04 | 56.83 |
| | | Ours | 66.97 | 51.38 | 57.72 | 57.31 | 43.81 | 55.44 |
| | | Ours (MM) | **80.16** | **61.34** | **73.70** | **67.44** | **57.85** | **68.10** |
| | Novel | FSKD [67] | 60.84 | 47.78 | 53.44 | 49.21 | 38.47 | 49.95 |
| | | Ours | 59.17 | 46.49 | 51.89 | 47.93 | 37.65 | 48.63 |
| | | Ours (MM) | **67.51** | **49.92** | **59.05** | **53.06** | **43.45** | **54.60** |
| Unseen | Base | FSKD [67] | 56.38 | 48.24 | 51.29 | 49.77 | 43.95 | 49.93 |
| | | Ours | 55.67 | 46.94 | 50.47 | 48.21 | 42.88 | 48.83 |
| | | Ours (MM) | **57.68** | **52.06** | **51.75** | **52.27** | **47.74** | **52.30** |
| | Novel | FSKD [67] | 52.36 | 44.07 | 47.94 | 42.77 | 36.60 | 44.75 |
| | | Ours | 50.88 | 43.34 | 46.67 | 42.52 | 35.19 | 43.72 |
| | | Ours (MM) | **54.61** | **45.92** | **48.02** | **43.86** | **40.31** | **46.54** |

Table 6. A quantitative comparison of the proposed method on query photos [15] using photo only and both edgemap and photos (MM) with the FSKD [67] in $K = 1$ for all evaluation settings.

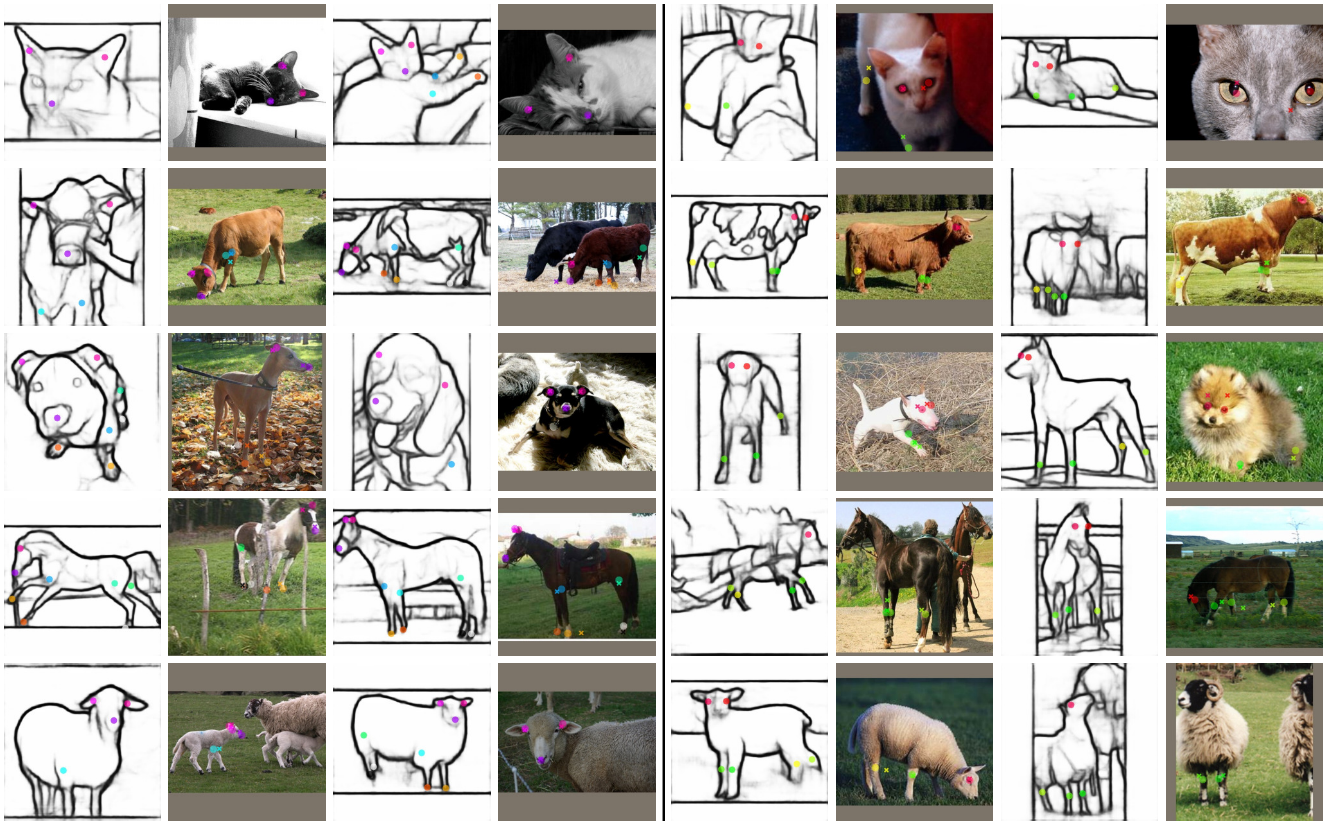

Figure 10. Visualizations of sample detection (✖) along with ground-truth (●) for base (left) and novel (right) keypoints on query photos [15] using support edgemaps [110].

ing on the de-stylization loss $\mathcal{L}_{\text{style}}$ (with reduced weight of $\lambda_{\text{style}} = 10^{-8}$) to learn the keypoint representation with *mutual information* from both photos and edgemaps. In this scenario, our method seems to outperform FSKD [67] by a significant margin ($\approx 2 - 11$) in all evaluation settings, proving the superiority of the multi-modal paradigm [70].

## 11. Style Diversity in Real Sketches

We simulate user sketch styles [100] with different edgemaps as the de-stylization network (Sec. 3.3) disentangles the style-invariant features. In order to understand its style-invariance and generalization capability to real sketches, we perform a *human study*, where each of the 20 participants was asked to draw 10 sketches, totalling 200 sketches. The participants were asked to rate every keypoint predicted on the query photos by the models in question, on a scale of $1 \rightarrow 5$ (bad→excellent) based on their *opinion* of how closely it matched their expectation. The proposed method achieves an average score of 4.42, compared to 2.91 of the B-Vanilla and 3.38 of the FSKD [67], underpinning the *generalizability* and *user-style independence* of our work. While we have used a limited number of real sketches from the Sketchy Extended [104] database, this study further proves the *practicality* of our framework.

## 12. Challenges with Additional Modalities

Our few-shot framework is particularly curated for sketch-photo cross-modal learning, and is vastly different from traditional sketch research [7, 8, 20], as careful addressing of the domain shift is well-observed in sketch-photo cross-modal literature [55, 71, 97], depending on tasks. The major sketch applications like sketch-based image retrieval [102, 103, 142] conventionally use a *joint representation* space due to the availability of instance-level sketch-photo pairs. However, *without* overlap of support and query sets by definition, the unavailability of such *instance-level sketch-photo pairs* necessitates the need for explicit keypoint-level domain adaptation (see Sec. 3.2) using a transport loss [115] on the de-stylized keypoint representation, accounting for the unique sparse nature of sketches.

This sketch-photo cross-modal domain adaptation becomes challenging when extended to other modalities. OpenKD [70] utilizes *text* as an additional guidance, along with annotated support photos, resulting in a multi-modal setup. In the context of sketch, we have experimented with a similar multi-modal setting with sketch and photo in Secs. 4.3 and 10. Taking the inspiration from OpenKD [70], we attempted *text-to-photo* cross-modal keypoint learning in our framework using an off-the-shelf frozen CLIP tex-

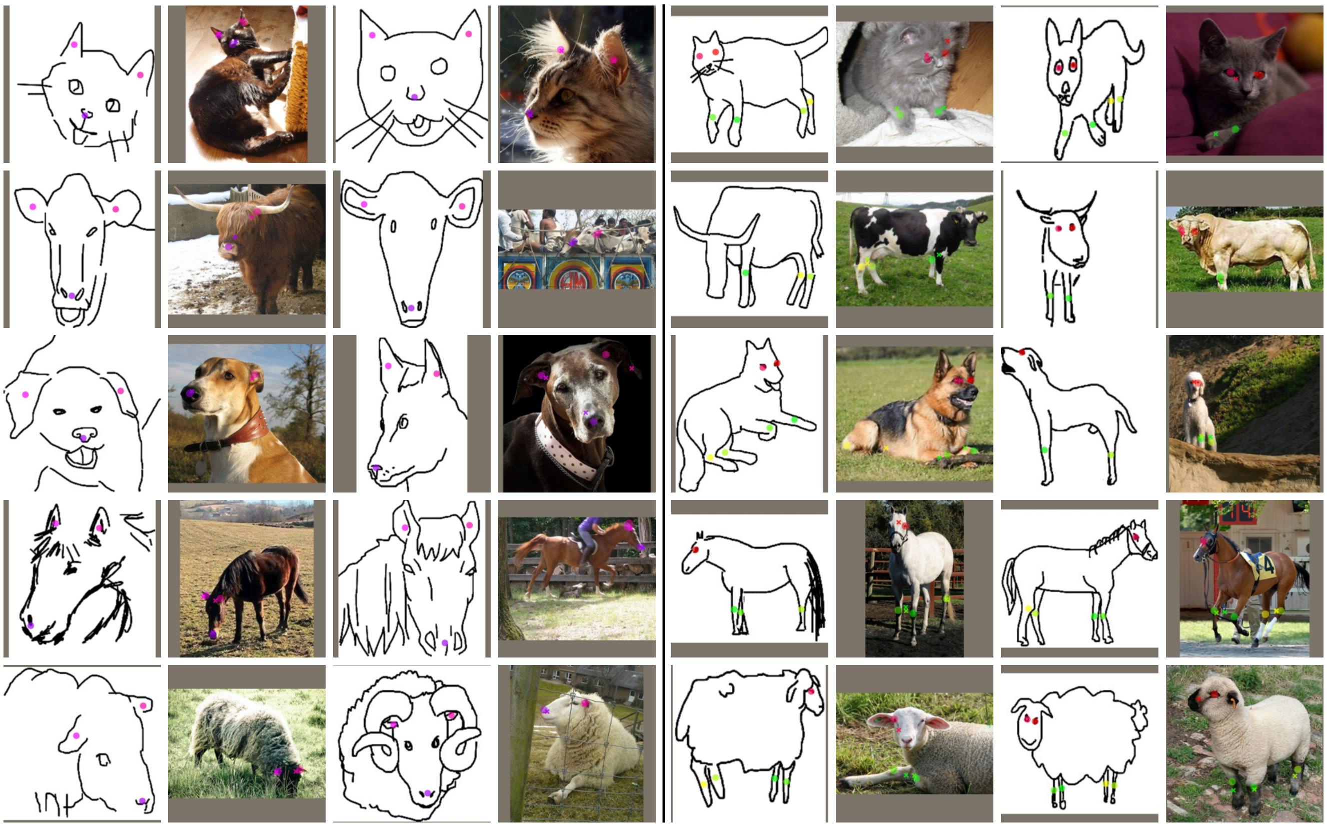

Figure 11. Visualizations of sample detection (✖) along with ground-truth (●) for base (left) and novel (right) keypoints on query photos [15] using real support sketches [104] with manual annotation prompt.

tual encoder [89] to obtain textual keypoint embeddings that replace support prototypes. The rest of the framework follows B-Vanilla (Sec. 3.1), using the feature modulation $\mathcal{M}$ to correlate the textual support prototypes with query features from the image encoder $F$, followed by the descriptor network $D$, and a GBL module for localization. This particular experiment achieves 20.13% (↑ 2.74 over B-Vanilla, ↓ 18.87 below proposed) for the novel keypoints on unseen classes in the Animal Pose [15] dataset. This poor performance is expected due to certain factors. Firstly, OpenKD [70] uses the annotated RGB photos as support along with text in a multi-modal setup. Incorporating it into a *source-free* paradigm is challenging, as the *text-photo joint keypoint representation* becomes unavailable due to the absence of such pairs in the support set. Secondly, our framework is designed to handle *image-like* data, *e.g*. sketch and photo, but it is not equipped to handle textual data, and needs explicit text-photo cross-modal design.

| Class | Keypoints | Methods | PCK@0.1 | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Cat | Cow | Dog | Horse | Sheep | **Mean** |
| Seen | Base | GeometryKP [41] | 33.47 | 22.57 | 26.96 | 27.08 | 19.84 | 25.98 |
| | | ProbIntr [81] | 47.93 | 36.77 | 41.11 | 42.47 | 32.92 | 40.24 |
| | | OpenKD [70] | 60.44 | 46.28 | 50.98 | 51.85 | 42.03 | 50.32 |
| | | **Proposed** | **67.34** | **49.89** | **56.28** | **56.35** | **45.65** | **55.10** |
| | Novel | GeometryKP [41] | 22.07 | 14.18 | 19.65 | 12.71 | 17.39 | 17.20 |
| | | ProbIntr [81] | 31.29 | 24.53 | 28.37 | 22.83 | 26.33 | 26.67 |
| | | OpenKD [70] | 49.92 | 38.27 | 42.74 | 38.46 | 34.17 | 40.71 |
| | | **Proposed** | **55.69** | **43.09** | **46.58** | **43.94** | **36.39** | **45.14** |
| Unseen | Base | GeometryKP [41] | 27.39 | 23.76 | 19.51 | 26.93 | 20.58 | 23.63 |
| | | ProbIntr [81] | 40.76 | 36.13 | 30.15 | 42.84 | 34.52 | 36.88 |
| | | OpenKD [70] | 44.96 | 41.54 | 36.49 | 43.18 | 39.06 | 41.05 |
| | | **Proposed** | **47.36** | **42.97** | **38.30** | **46.17** | **41.03** | **43.17** |
| | Novel | GeometryKP [41] | 14.25 | 8.69 | 11.81 | 13.44 | 9.37 | 11.51 |
| | | ProbIntr [81] | 23.91 | 17.59 | 18.37 | 13.85 | 21.63 | 19.07 |
| | | OpenKD [70] | 40.38 | 39.72 | 36.07 | 35.91 | 34.67 | 37.35 |
| | | **Proposed** | **44.42** | **40.13** | **36.91** | **37.77** | **35.77** | **39.00** |

Table 7. Additional quantitative comparison of the state-of-the-art strategies with the proposed framework in $K = 1$ shot setting delineating the superiority of the proposed method in overall performance on Animal Pose [15] dataset.

## 13. Additional Comparisons

Although few-shot keypoint learning [33, 69, 112] has been around for some time, only a few state-of-the-art methods are suitable for comparison to the proposed method. Our framework follows FSKD [67] closely, and we compare them in Tab. 1. Additionally, we adapt ProbIntr [81] to a few-shot framework. We also consider GeometryKP [41] and OpenKD [70] for comparison. Due to the unavailability of open-source code bases for these works, we implement them to the best of our ability and compare them in Tab. 7.