CoKe: Contrastive Learning for Robust Keypoint Detection

Yutong Bai[1*] Angtian Wang[1*] Adam Kortylewski[2,3†] Alan Yuille[1†]
[1]Johns Hopkins University [2]University of Freiburg [3]Max-Planck-Institute for Informatics

Abstract

In this paper, we introduce a contrastive learning framework for keypoint detection (CoKe). Keypoint detection differs from other visual tasks where contrastive learning has been applied because the input is a set of images in which multiple keypoints are annotated. This requires the contrastive learning to be extended such that the keypoints are represented and detected independently, which enables the contrastive loss to make the keypoint features different from each other and from the background. Our approach has two benefits: It enables us to exploit contrastive learning for keypoint detection, and by detecting each keypoint independently the detection becomes more robust to occlusion compared to holistic methods, such as stacked hourglass networks, which attempt to detect all keypoints jointly. Our CoKe framework introduces several technical innovations. In particular, we introduce: (i) A clutter bank to represent non-keypoint features; (ii) a keypoint bank that stores prototypical representations of keypoints to approximate the contrastive loss between keypoints; and (iii) a cumulative moving average update to learn the keypoint prototypes while training the feature extractor. Our experiments on a range of diverse datasets (PASCAL3D+, MPII, ObjectNet3D) show that our approach works as well, or better than, alternative methods for keypoint detection, even for human keypoints, for which the literature is vast. Moreover, we observe that CoKe is exceptionally robust to partial occlusion and previously unseen object poses.

1. Introduction

Semantic keypoints, such as the joints of a human body, provide concise abstractions of visual objects in terms of their shape and pose. Accurate keypoint detections are of central importance for many visual understanding tasks, including viewpoint estimation [30], human pose estimation [5], action recognition [26], feature matching [24], image classification [46], and 3D reconstruction [20]. There are

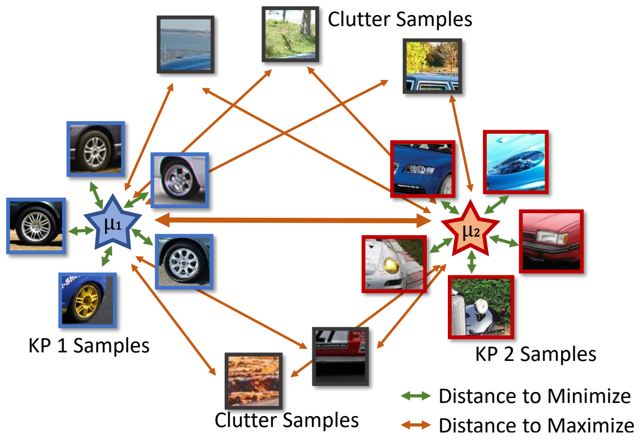

Figure 1. Intuition behind our approach. Image patches depict feature representations of two different keypoints (blue and red border) and background clutter (grey border). The star shapes illustrate the average representations μ_1 and μ_2 of the corresponding keypoint features. Our approach learns a representation space such that the following three distances are optimized: (1) The distance between features of the same keypoint is small, i.e. they cluster tightly around their mean. (2) The distance between the keypoint clusters is maximized. (3) The distance between the clutter features and the keypoint centers is maximized.

many diverse approaches for keypoint detection. Common approaches include the application of a regression loss [28, 21], a classification loss [14], or combinations of either of those with a 3D geometric model of the object [48]. In recent years, work in contrastive learning has led to major advances in representation learning [7, 13, 27] demonstrating benefits over classical losses, such as cross-entropy, e.g. in terms of robustness and data efficiency [22]. However, most works on contrastive learning in computer vision focus on the task of image classification, and it remains unclear how contrastive learning can be applied for keypoint detection.

In this paper, we introduce a contrastive learning framework for keypoint detection (CoKe). Keypoint detection differs from other visual tasks where contrastive learning has been applied, such as face recognition [33] or unsupervised learning [13], because the input is a set of images

*Joint first authors
†Joint senior authors

in which multiple keypoints are annotated. To enable the contrastive learning of keypoint detectors, we need to represented and detected keypoints independently, such that a contrastive loss can make the keypoint features different from each other and from the background. This is very different from current popular approaches to keypoint detection, such as stacked hourglass networks [**?**], which attempt to detect all keypoints jointly. Our approach has two benefits: It enables us to exploit contrastive learning for keypoint detection, and by detecting each keypoint independently the detection can become more robust to occlusion compared to holistic methods (as shown in our experiments).

Our CoKe framework introduces several technical innovations. Specifically, we found that the contrastive learning of keypoint representations requires the optimization of three types of distances in the feature space (Figure 1):

(1) The distance between features of the same keypoint should be small. But the computational cost to calculate the distance between all features of the same keypoint is quadratic in their number. We instead reduce the computational cost to be linear by introducing an average prototypical representation for each keypoint (stars in Figure 1).

(2) The distance between features of different keypoints should be large. The number of distance comparisons between features of different keypoints is combinatorial in the number of keypoints and training images. To manage this computational burden, we introduce a *keypoint bank* which stores the prototypical representations of all keypoints and allows for an efficient computation of the distance between the prototypes of different keypoints (Figure 1 bold orange arrow).

(3) The distance between keypoint features and features from the background clutter (Figure 1 grey squares) should be large to reduce false-positive detections. However, most of the features in an image are clutter features and it is not feasible to compute the distance to all of them. Therefore, we introduce a *clutter bank* that keeps track of clutter features that are spatially close to keypoint features and hence are most difficult to be distinguished from.

The proposed approximations enable an efficient contrastive learning of keypoint detectors. We evaluate CoKe on several datasets including PASCAL 3D+, MPII, and ObjectNet3D. We observe that CoKe performs on par, and often even better, compared to SOTA related work (Stacked Hourglass Networks, MSS-Net [21]) and to approaches that use additional supervision in terms of detailed 3D object geometries (StarMap [48]). These results are remarkable as CoKe works well on all these datasets while, for example, the best results on MPII are often achieved by architectures that are specialized for human keypoint detection. We also observe that when compared to related work, CoKe is exceptionally robust to partial occlusion and unseen object poses. Our main contributions are:

1. We introduce a contrastive learning framework for keypoint detection.
2. CoKe performs very well on various keypoint detection datasets for rigid and articulated objects.
3. CoKe achieves exceptional robustness to partial occlusion and previously unseen object poses.

2. Related Work

Keypoint Detection. Keypoint detection, is a widely studied problem in computer vision. Popular applications are, e.g., the detection of human joints [5, 28, 36, 37] or distinct locations on rigid objects [42, 38, 30, 48]. Early approaches relied on local descriptors [11, 32, 45, 8, 10] that are distinctive and invariant [25]. While approaches using local descriptors have proven to be robust to occlusion and background clutter, they were outperformed by deep learning approaches that were trained end-to-end [28]. Toshev et al. [37] first trained a deep neural network for 2D human pose regression and Li et al. [23] extended this approach to 3D. Starting from the work of Tompson et al. [36], regression-based approaches to keypoint detection became very popular. They perform keypoint detection by regressing a heatmap representation. These approaches achieve a particularly good performance at detecting the joints of both articulated and rigid objects [28], because they can implicitly leverage the structural information between keypoint to resolve locally ambiguous keypoint detections. Tulsiani et al. [38], proposed to integrate the structural information between keypoints explicitly by integrating 2D and 3D models, which inspired a number of follow-up works, in particular for rigid objects [48, 38, 30].

Supervised Contrastive Learning. Contrastive learning, originates from Metric Learning [6, 41, 31] and involves the learning of a representation space by optimizing the similarities of sample pairs in this space. Intuitively, supervised contrastive learning aims to reduce the distance of feature representations of the same class, while increasing the distance between samples from different classes. Popular examples use pairs of samples for loss computation [12], triplets [33], or N-Pair tuples [34].

Recently, contrastive learning has attracted attention from the research community in self-supervised learning [7, 13, 43, 27, 16]. The main difference in the self-supervised learning setting is that positive examples are usually generated using data augmentations[9] or co-occurrence [18, 35] of a query sample, whereas negative examples are chosen as other images in the same mini-batch.

While most of the supervised contrastive learning [22] focuses on learning a holistic representation of the complete image, in this paper, we target a more fine-grained task - keypoint detection. Keypoints are localized image patterns and therefore require the learning of local feature embeddings. The main challenge is that local image patterns can be highly ambiguous (e.g. the front and back tire of a car) and therefore require a contrastive learning framework that can learn to disambiguate local representations, while at the same time being able to learn a distinct representation that can be localized accurately.

3. CoKe: Contrastive Keypoint Learning

In this section, we present our framework for contrastive keypoint learning, then discuss the intuition underlying our approach and the training pipeline and how to perform keypoint detection during inference.

3.1. Training CoKe

We use Φ to denote the feature extractor. Given an input image $\mathbf{I}^i$, it computes the feature map $\Phi(\mathbf{I}^i) = \mathbf{F}^i \in \mathbb{R}^{H\times W\times D}$, where i is the image index in the training data $\{\mathbf{I}^i | i \in \{1,\dots,N\}\}$. Using the keypoint annotation we retrieve the corresponding keypoint features $\mathbf{f}_k^i \in \mathbb{R}^D$ for the keypoints $k \in \mathbb{K}$ from the feature map $\mathbf{F}^i$. Similarly, we can (randomly) select a non-keypoint location as clutter point and retrieve a set of clutter features $\{\mathbf{f}_c^i \in \mathbb{R}^D | c \in \{1,\dots,C\}\}$. We define the distance between two features as $d(\cdot,\cdot)$. During training we learn a feature extractor that optimizes the following distances in the feature space:

(1) One objective for the feature extractor during training is to **minimize the distance between features of the same keypoint** (i.e. the intra-keypoint distance) across all training images, by minimizing the objective:

$$D_{intra}(\mathbf{f}_k^i) = \sum_{j=1}^{N} d(\mathbf{f}_k^i, \mathbf{f}_k^j) \approx d(\mathbf{f}_k^i, \mu_k). \quad (1)$$

However, as we described in the introduction it is unpractical to compute the distance of a keypoint's feature vector from one image $\mathbf{f}_k^i$ to the corresponding vectors in all other training images $j \in \{1,\dots,N\}$. To resolve this computational problem, we define a prototypical keypoint feature μ_k, that represents the average feature of keypoint k. Instead of computing the full objective, we approximate it as simply the distance to the corresponding average representation $d(\mathbf{f}_k^i, \mu_k)$. We store the prototypical features of all keypoints $\{\mu_k\}$ in a *keypoint bank* and update them during training.

(2) The second objective for the feature extractor is to **maximize the distance between features of different keypoints** (i.e. the inter-keypoint distance). This requires computing the distance between the feature representations of one particular keypoint k and all other keypoints $k^{'}$ over all training images:

$$D_{inter}(\mathbf{f}_k^i) = \sum_{k' \in K\setminus\{k\}} \sum_{j=1}^{N} d(\mathbf{f}_k^i, \mathbf{f}_{k'}^j) \approx \sum_{k' \in K\setminus\{k\}} d(\mathbf{f}_k^i, \mu_{k'}). \quad (2)$$

We approximate this objective by instead computing the distance to the prototypes of the respective keypoint features from the keypoint bank.

(3) The third objective for the feature extractor is to **maximize the distance between the keypoint features and all clutter features**. In the best case, this involves computing the distance between a keypoint feature $\mathbf{f}_k^i$ to every clutter feature in all training images. To avoid the computation of this large amount of distances, we instead approximate this objective by storing a subset of all clutter features $\{\theta_c, c \in \mathbb{C}\}$ in a *clutter bank*. Which allows us to approximate the full objective with

$$D_{clutter}(\mathbf{f}_k^i) \approx \sum_{c \in \mathbb{C}} d(\mathbf{f}_k^i, \theta_c). \quad (3)$$

These three approximations make it feasible to optimize the overall objective with a feasible computational load. As the parameters of the feature extractor will change during learning, the clutter features $\mathbf{f}_c$ in the clutter bank as well as the prototypes μ_k need to be updated. To achieve this, we follow an EM-type optimization process. First, we initialize the clutter bank by randomly sampling clutter features from the training data, and initialize the prototypes by computing the average feature for every keypoint across the training data $\mu_k = \frac{\sum_{i=1}^{N} \mathbf{f}_k^i}{N}$. Using these initial estimates, we can compute the overall objective and train the feature extractor. While training the feature extractor, we update the clutter and keypoint bank. We perform those updates in an alternating manner.

3.1.1 Keypoint and Clutter Bank Update

Figure 2, illustrates the process of updating the keypoint prototypes and the clutter bank during training.

Keypoint Bank Update. Computing the prototypical keypoint features μ_k while learning the feature extractor is challenging, because we want to avoid to re-compute the prototypes over all training images $\mu_k = \frac{\sum_{i=1}^{N} \mathbf{f}_k^i}{N}$ after every gradient step. Instead, we approximate the sample mean via a cumulative moving average. Specifically, we update μ_k with a training batch of size m using:

$$\mu_k \leftarrow \mu_k * \alpha + \frac{\sum_{i=0}^{m} \mathbf{f}_k^i}{m} * (1-\alpha). \quad (4)$$

Clutter Bank Update. The clutter bank contains a limited number C of clutter features $\{\theta_c, c \in \mathbb{C}\}$. In practice the size of the clutter bank depends on the availability of

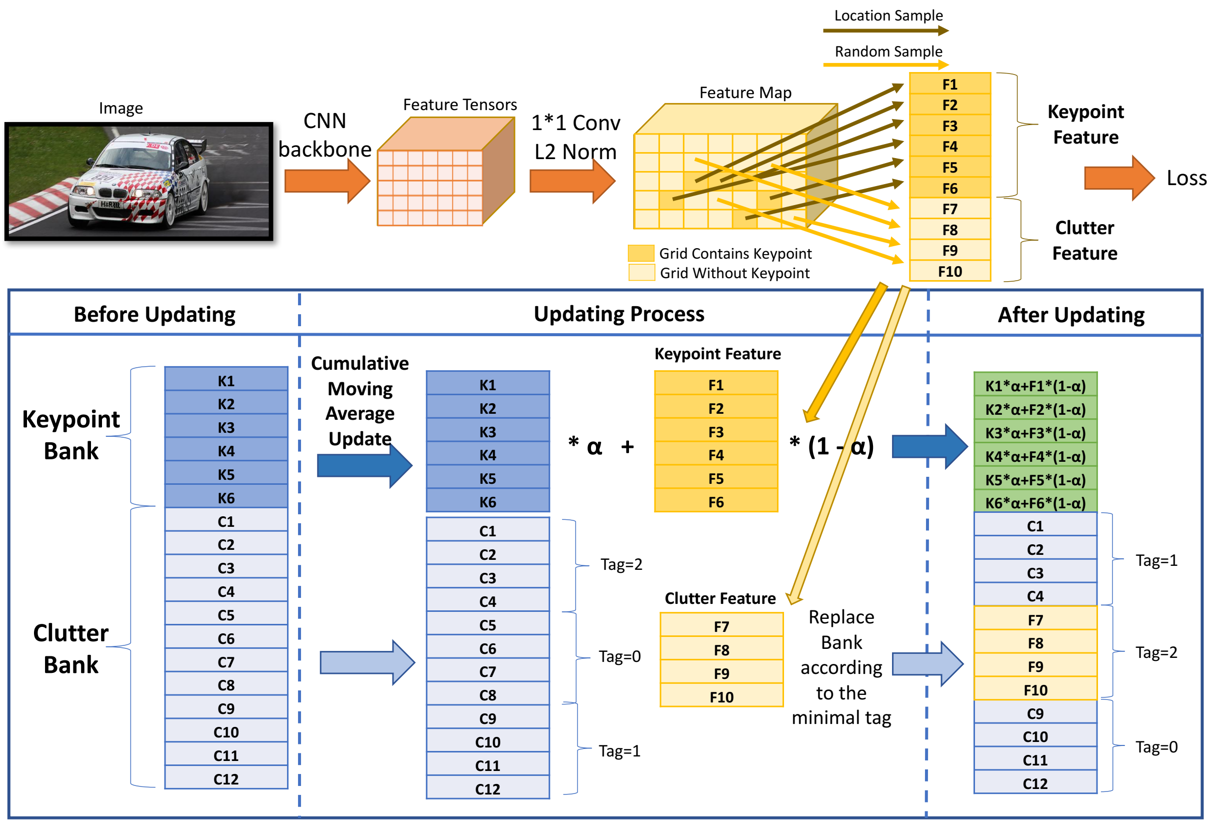

Figure 2. Illustration the Keypoint and Clutter Bank update process. First, a feature map for the input image is extracted. After a dimensionality reduction and L_2 normalization, we retrieve the keypoint features (F1-F6) and randomly selected clutter features (F7-F10). The Keypoint Bank is updated using a cumulative moving average update. The Clutter Bank is updated by replacing the oldest features in the Clutter Bank with (F7-F10) based on the time tag.

GPU memory and we observe that the larger the bank, the better the training performance (see experiments section). We update the clutter bank during training by replacing the oldest clutter features with newly extracted features from the current training batch based on a time tag that indicates how long features were stored in the bank.

3.1.2 Feature Extractor Training

When training the feature extractor with SGD, we freeze the keypoint bank and clutter bank in every gradient step and use them to compute the gradient update on the weights of the feature extractor. To compute the distance between two feature vectors, we use the L2 distance:

$$d(\mathbf{f}_a, \mathbf{f}_b) = (\mathbf{f}_a - \mathbf{f}_b)^2 = 2 * (1 - \mathbf{f}_a \cdot \mathbf{f}_b). \qquad (5)$$

The last step in above equation leverages that all features in our model are L2 normalized. From Eq. 5 observe that we can minimize $D_{intra}(\mathbf{f}_k^i)$ by maximizing $\mathbf{f}_k^i \cdot \mu_k$. Similarly, we can maximize $D_{inter}(\mathbf{f}_k^i)$ and $D_{clutter}(\mathbf{f}_k^i)$ by minimizing $\{\mathbf{f}_k^i \cdot \mu_{k'} | \forall k' \in K \setminus \{k\}\}$ and $\{\mathbf{f}_k^i \cdot \theta_c | \forall c \in \mathbb{C}\}$ respectively. To optimize those terms simultaneously, we use a non-parametric softmax as our loss function. Thus, the loss for each keypoint feature is calculated as:

$$\mathcal{L}(\mathbf{f}_k^i, \{\mu_k\}, \{\theta_c\}) = \frac{e^{\mathbf{f}_k^i \cdot \mu_k}}{\sum_{k' \in K} e^{\mathbf{f}_k^i \cdot \mu_{k'}} + \sum_{c' \in C} e^{\mathbf{f}_k^i \cdot \theta_{c'}}}, \qquad (6)$$

where $\{\mu_k\}$ is the keypoint bank and $\{\theta_c\}$ is the clutter bank.

Clutter Sampling Loss. A technical problem is that the features in the clutter bank are copied from a large number of training images, and therefore it is not practical to calculate gradient directly w.r.t. $\{\theta_c\}$, and therefore the feature extractor is not optimized w.r.t. the clutter features. This technical limitation makes the optimization w.r.t. the clutter distance $D_{clutter}(\mathbf{f}_k^i)$ converge slowly, especial when the clutter bank is large. To make the training more efficient, we propose a clutter sampling loss, which uses the sampled clutter features $\mathbf{f}_c^i$ from the current training batch and directly maximizes the clutter to keypoint distance:

$$\mathcal{L}(\mathbf{f}_c^i, \{\mu_k\}) = \sum_{k' \in K} \mathbf{f}_c^i \cdot \mu_{k'}. \qquad (7)$$

Thus, the final loss for training the feature extractor becomes:

$$\mathcal{L}(\mathbf{F}^i,\{\mu_k\},\{\theta_c\})=\sum_{k\in\mathbb{K}}\mathcal{L}(\mathbf{f}_k^i,\{\mu_k\},\{\theta_c\})+\sum_{c\in\mathbb{C}}\mathcal{L}(\mathbf{f}_c^i,\{\mu_k\}). \quad (8)$$

3.2. Inference with the CoKe Model

In the following, we describe the detection process for keypoints of a single category, but note that this can be trivially extended to multiple categories. Figure 3 illustrates the inference process with the CoKe. To locate the predicted the keypoint position p_k for keypoint k on the feature map of a test image $\mathbf{I}^i$, we apply the following steps:

Extract a feature map $\Phi(\mathbf{I}^i) = \mathbf{F}^i$ using the trained feature extractor Φ. Use the prototypes $\mu_k, k \in \mathbb{K}$ to compute the per pixel feature distance $d(\mathbf{f}, \mu_k)$ for all feature vectors $\mathbf{f} \in \mathbf{F^i}$ and store the detection scores in the output $\mathbf{S} \in \mathbb{R}^{H\times W\times K}$. For each keypoint, select the position p with the highest detection score in $\mathbf{S}$. Project p back to the original image coordinates.

4. Experiments

In this section, we experimentally evaluate CoKe and compare it to related works. We first describe the experimental setup, compare CoKe to related work on several diverse datasets, and study their robustness to partial occlusion. Then discuss qualitative results with ablation studies.

4.1. Experimental Setup

Evaluation Protocol. Evaluation is done using the standard Percentage of Correct Keypoints (PCK) metric which reports the percentage of detections that fall within a normalized distance of the ground truth. we use PCK=0.1 for PASCAL3D+, ObjectNet3D and OccludedPASCAL3D+ datasets following the standard experimental protocol. For MPII, we use PCKh=0.5 as the evaluation metric. Distance is normalized by a fraction of the head size (PCKh). We follow the common protocol of evaluating each object category by computing the average accuracy on the visible keypoints over all the test images.

Training Setup. We use the standard train-val-test split for all the datasets. We use a batch size of 64 for training. For each image, we randomly select a group of 20 clutter points. The full clutter bank consists of 1024 such groups. We choose the clutter features to be within two pixels distance to the keypoint annotation in the feature map. We use the non-parametric softmax [43] to calculate the similarity between features and banks. The temperature parameter[17] that controls the concentration level of the distribution is set to $\tau = 0.7$.

PASCAL3D+ Dataset. The dataset contains 12 man-made object categories with totally training 11045 images and 10812 evaluation images. Different from previous works [48, 38], we use all images for evaluation, including occluded and truncated ones.

MPII Dataset. MPII Human Pose [1] consists of images taken from a wide range of human activities with a challenging array of articulated poses. The keypoint visibility is annotated, enabling us to report numbers for the full dataset as well as partially occluded humans.

ObjectNet3D Dataset. ObjectNet3D consists of common daily life objects and is notably more difficult compared to PASCAL3D+ as it contains more rare viewpoints, shapes and truncated objects with occlusion. We test on nine categories, chosen based on their high annotation accuracy, as some categories in ObjectNet3D have low quality annotations due to the complexity of this dataset.

OccludedPASCAL3D+ Dataset. While it is important to evaluate algorithms on real images of partially occluded objects, simulating occlusion enables us to quantify the effects of partial occlusion more accurately. We use an analogous dataset with artificial occlusion for keypoint detection proposed in [39] for object detection. It contains all 12 classes of the PASCAL3D+ dataset at various levels of occlusion. The dataset has a total of 3 occlusion levels, with Lv.1: 20-40%, Lv.2: 40-60% and Lv.3: 60-80% of the object area being occluded.

4.2. Performance on Various Datasets

PASCAL3D+ and OccludedPASCAL3D+. Table 1 shows the keypoint detection results on the PASCAL3D+ dataset for CoKe models learned from three different backbones: ResNet-50 [15], Stacked-Hourglass-Network and Res-UNet [47]. We also show the performance of StarMap [48] as reported in the original paper. Note that StarMap uses 3D models as additional supervision to jointly reason about the relative position of the keypoints. The performance of CoKe with all backbones is constantly high. The highest performance is achieved with the most recently developed architecture Res-UNet. When compared to the original Stacked-Hourglass-Network trained with a regression loss (SHG) we can clearly observe a large gain in performance. Most notably, the performance difference is very prominent for strong occlusion. We believe CoKe is more robust to occlusion, because of two reasons: 1) It is actively optimized to discriminate between keypoint features and clutter features based on their local representations only, which might help to reduce the effective receptive field size and hence reduces the negative effects of occlusions. 2) CoKe stores the clutter representation from a large number of images, and hence can better learn to distinguish keypoints from clutter. Overall, our results clearly highlight that CoKe is very competitive with related work, while being highly robust to partial occlusion.

In Table 2, we demonstrate an additional robustness to

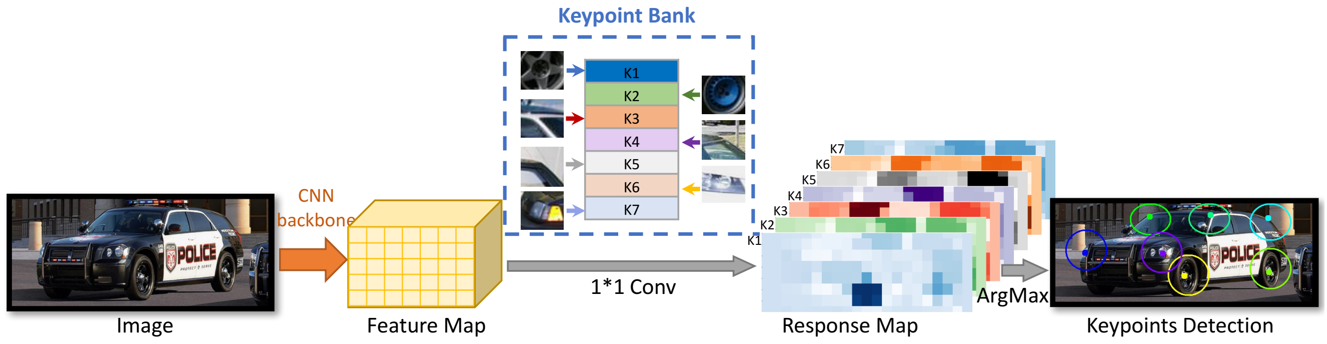

Figure 3. Keypoint detection with CoKe. We first extract feature representation at different positions of input image. Each keypoint in the Keypoint Bank has an individual representation used as a convolution kernel to compute a response map. The location of maximum response is used as prediction result. Colored boxes show ground truth and dots the prediction result.

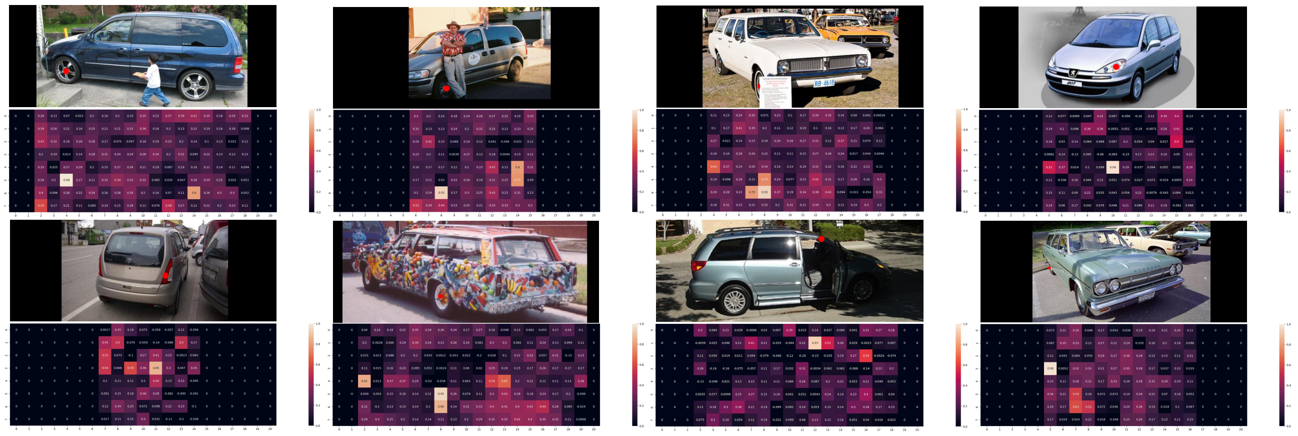
Figure 4. Eight examples of CoKe-Res50's representation visualization. For each sub-figure, top is the original image with keypoint annotation, labeled with red dot. Bottom is the response map, predicted by CoKe-Res50. It is worth noting that how all keypoints are detected accurately despite the difficulty from false positives, occlusions, rare viewpoints, different domain, irregular appearance and irregular state and rare viewpoints.

unseen object poses. We divide the azimuth pose in the *car* category of Pascal3D+ into 4 bins, front, back, left and right. We train CoKe on the front and back subsets, and test on seen (7305 images) and unseen (3507 images) poses separately. Table 2 shows that CoKe is much more robust to unseen poses compared to SHG.

PASCAL3D+					
Occlusion Level	Lv.0	Lv.1	Lv.2	Lv.3	Avg
SHG	68.0	46.5	43.2	39.9	49.4
MSS-Net	68.9	46.6	42.9	39.6	49.5
StarMap	78.6	-	-	-	-
CoKe-Res50	77.0	67.6	**59.9**	53.4	64.4
CoKe-SHG	78.3	66.3	58.4	52.3	63.8
CoKe-ResUnet	**80.3**	**68.5**	59.1	**54.0**	**65.5**

Table 1. Keypoint detection results on PASCAL3D+ under different levels of partial occlusion (Lv.0:0%,Lv.1:20-40%,Lv.2:40-60%,Lv.3:60-80% of objects are occluded, L0 is the original dataset). CoKe outperforms baseline models and is highly robust to partial occlusion.

PASCAL3D+		
	Seen-Pose	Unseen-Pose
CoKe-SHG	94.0	**84.0**
SHG	**94.2**	73.6

Table 2. Robustness of keypoint detectors to unseen poses. For the *car* category, we separate PASCAL3D+ training and testing set into 4 bins using azimuth annotations. We train CoKe on the front and back subsets, and test under both seen and unseen poses.

MPII. We compare CoKe learned from the SHG backbone to a number of related works on MPII in Table 3. CoKe-SHG is again highly competitive and outperforms related work by a small but significant margin. Table 4 compares CoKe-SHG and SHG for both occluded and non-occluded keypoint detection on the MPII dataset. We can observe that CoKe achieves significantly higher results on occlusion scenarios compared with SHG. When compared to a recent and highly competitive baseline such as RSNs [4], we can observe that RSN achieves a higher performance on the full dataset, due to its advanced architecture that has more than *four times* more parameters compared to our model. However, we observe that the model suffers from a relatively higher performance decrease when the humans are occluded, hence it is not as robust as our model that uses a standard Stacked-Hourglass backbone. We also see the potential for the performance of our model to further increase with more advanced backbones.

MPII	
RecurrentPose[2]	88.1
PoseMachines[40]	88.5
DeeperCut[19]	88.5
PartHeatmap[3]	89.7
SHG[28]	90.9
DualPathNetworks[29]	91.2
PoseRegression[3]	91.2
CoKe-SHG	**91.4**

Table 3. Keypoint detection results on MPII compared with regression based algorithms.

MPII		
	Full	Occluded
SHGs	90.1	84.3
RSN [4]	92.7	86.9
CoKe-SHG	91.4	86.7

Table 4. Keypoint detection results on MPII of CoKe-SHG and SHG on both original and challenging scenarios.

ObjectNet3D. We compare SHG and a CoKe-SHG on ObjectNet3D in Table 5. CoKe-SHG outperforms SHG by a large margin on every category. Since ObjectNet3D dataset contains much more challenging scenarios, we claim that CoKe is more robust.

ObjectNet3D					
	coffee	dryer	kettle	jar	wash
SHG	31.0	35.1	32.2	41.9	33.9
CoKe-SHG	**36.4**	**37.6**	**37.8**	**45.2**	**36.1**
	can	calc	eyegls	guitar	mean
SHG	66.8	52.3	44.6	45.8	42.62
CoKe-SHG	**70.2**	**57.7**	**51.3**	**49.6**	**46.88**

Table 5. Keypoint Detection results on ObjectNet3D+ [44].

In summary, we observe that CoKe is a general purpose framework that constantly achieves a very high performance across a wide range of backbone architectures and for a range of datasets with very different characteristics, while also being highly robust to occlusion.

4.3. Qualitative Results

Visualization of Keypoint Detection Maps. We visualize the part detection maps from CoKe-Res50 in Figure 4. Images are selected from the *car* category in PASCAL3D+ which has complex changes in pose, illumination and object structure. Note how all keypoints are detected accurately despite the difficulty from false positives, occlusions, rare viewpoints, different domain, irregular appearance. We can observe that CoKe is robust in these challenging scenarios.

Visualization of Detection Results under Occlusion. We visualize qualitative results in Figure 5. Overall, the illustrations demonstrate the robustness of CoKe to partial occlusions. Any keypoints that are not in the vicinity of occluders are correctly detected and not affected by the occlusion. Furthermore, keypoints that are partially occluded (e.g. the wheel) can still be located robustly, although the detections tend to move away from the occluder. Importantly, we do not observe false positive detections at locally ambiguous keypoints. This demonstrates that CoKe leverages the receptive field to disambiguate keypoints, while still being able to localize individual keypoints accurately.

Inference Time and Memory Consumption. During inference, CoKe-Res50 (params: 23M, acc: 77%) takes 0.01s per image, while SHGs (params: 25M, acc: 68%) needs 0.06s. For the memory consumption, CoKe-Res50 needs 715MB, SHGs 786MB when batch size equals to 1. CoKe has an advantage of the inference time while maintaining the competence of the memory consumption.

PASCAL3D+			
Backbone	Reg	Class	CoKe
SHGs	68.0	67.8	**78.3**
Res50	68.9	69.3	**77.0**
ResUnet	69.7	70.2	**80.3**

Table 6. Ablation study of CoKe training strategy in comparison of standard regression or classification training loss.

4.4. Ablation Study

Comparison with Other Losses. In Table 6, we ablate the effect of the contrastive learning compared to other common losses on different backbone architectures. Specifically, we compare to a regression loss as used in SHGs and a supervised classification loss. The classification baseline enables us to compare the effect of the contrastive learning with a cross-entropy loss. We implement keypoint detection as a classification problem by using the features at the ground truth keypoint locations in the training data as instances of different classes to train the backbone with the cross-entropy objective. During testing, we use the average representations from the training set as part detector, similar as in the CoKe pipeline. The main difference to the CoKe training is hence that the contrastive loss and the clutter bank are not used. From the results, we observe that our representation learning formulation of keypoint detection constantly outperforms other losses by a large margin.

Clutter Bank Mechanism In Table 7, we study the influence of the clutter features and the clutter sampling

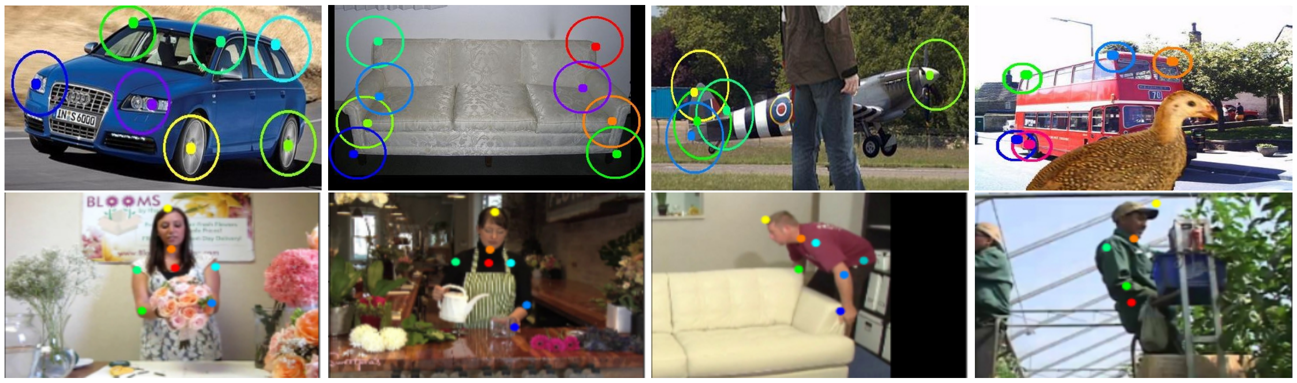
Figure 5. Qualitative detection results under different levels of partial occlusion for artificially occluded objects from PASCAL3D+ and humans from MPII. The dots visualize the detection result of CoKe. The colored circles in indicate the ground-truth position within PCK=0.1, and indicate ground-truth position within PCKh=0.5 for MPII. Note how CoKe is very robust even under strong occlusion.

Occlusion Level	Lv.0	Lv.1	Lv.2	Lv.3
No clutter	79.3	75.4	71.8	65.8
Image-specific clutter	92.8	82.7	76.5	69.2
Clutter Bank (64 groups)	93.0	83.6	**80.1**	**73.3**
Clutter Bank (256 groups)	94.3	84.3	77.7	71.0
Clutter Bank (1024 groups)	**95.5**	**85.9**	79.0	70.6
Clutter Bank w/o clutter sampling loss	94.2	83.1	76.8	68.0

Table 7. Ablation study on PASCAL3D+ with different settings: no clutter features, image-specific clutter features (using 20 features from the same image as clutter examples), our proposed Clutter Bank with different number of groups (each group contains 20 features) and deactivating clutter loss. Note the benefit of using clutter features in general, and in particular using a large clutter bank, as well as the importance of the clutter sampling loss.

loss on the contrastive learning result. In particular, the table shows the keypoint detection results for CoKe-Res-UNet on the car category of the PASCAL3D+ dataset. We observe that the performance decreases significantly when no clutter features are used during training. An extension of this basic setup is to use image-specific clutter features but without maintaining the features in a Clutter Bank. In particular, we select the clutter features from the same image from which the keypoint features are sampled using the same hard negative sampling mechanism as in our standard setup. From the results we observe that using the image-specific clutter features increases the performance significantly. However, the best performance is achieved using our proposed Clutter Bank mechanism. In particular, the results show a general trend that the more features we store in the bank, the higher the performance becomes. Notably, lower occlusion scenarios benefit from a larger clutter bank while for stronger occlusion a smaller clutter bank is more beneficial. Finally, our ablation shows that explicitly regularizing the clutter to be distinct from the Keypoint Bank using the clutter sampling loss is highly beneficial.

Cumulative Moving Average Update We also study the importance of the cumulative moving average update. Here we provide another possible method: average approximation. In particular, we calculate an average over the whole dataset for $\{\theta\}$ each 10 epochs. We report the result using CoKe-Res-UNet on the car category of the PASCAL3D+ dataset. We observe a deduction of keypoint detection accuracy as L0: 94.2 $\rightarrow$ 88.7, L1: 83.1 $\rightarrow$ 80.0, L0: 76.8 $\rightarrow$ 75.0, L0: 68.0 $\rightarrow$ 68.9. Conclusively, the best performance is achieved using our proposed cumulative moving average update mechanism.

5. Conclusion

In this paper, we study keypoint detection from the perspective of contrastive learning. Our contrastive keypoint learning framework (CoKe) makes several efficient approximations to enable the contrastive representation learning for keypoint detection: (i) A clutter bank to approximate non-keypoint features; (ii) a keypoint bank that stores prototypical representations of keypoints, to approximate the within-keypoint distance; and (iii) a cumulative moving average update to learn the keypoint prototypes while training the feature extractor. Our experiments on several diverse datasets (PASCAL3D+, MPII, ObjectNet3D) show that CoKe is very general and performs well for rigid as well as articulated objects. Compared to related work, CoKe achieves higher performance, while also being much more robust to partial occlusion and unseen object poses.

Acknowledgements. AK acknowledges support via his Emmy Noether Research Group funded by the German Science Foundation (DFG) under Grant No. 468670075.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 5

[2] Vasileios Belagiannis and Andrew Zisserman. Recurrent human pose estimation. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 468–475. IEEE, 2017. 7

[3] Adrian Bulat and Georgios Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In *European Conference on Computer Vision*, pages 717–732. Springer, 2016. 7

[4] Yuanhao Cai, Zhicheng Wang, Zhengxiong Luo, Binyi Yin, Angang Du, Haoqian Wang, Xiangyu Zhang, Xinyu Zhou, Erjin Zhou, and Jian Sun. Learning delicate local representations for multi-person pose estimation. In *European Conference on Computer Vision*, pages 455–472. Springer, 2020. 6, 7

[5] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299, 2017. 1, 2

[6] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 11–14. Springer, 2009. 2

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 1, 2

[8] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. *arXiv preprint arXiv:1606.03558*, 2016. 2

[9] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018. 2

[10] Peter R Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. *arXiv preprint arXiv:1806.08756*, 2018. 2

[11] Nicolas Gourier, Daniela Hall, and James L Crowley. Facial features detection robust to pose, illumination and identity. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, volume 1, pages 617–622. IEEE, 2004. 2

[12] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006. 2

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019. 1, 2

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[16] Olivier J Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019. 2

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 5

[18] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 2

[19] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *European Conference on Computer Vision*, pages 34–50. Springer, 2016. 7

[20] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018. 1

[21] Lipeng Ke, Ming-Ching Chang, Honggang Qi, and Siwei Lyu. Multi-scale structure-aware network for human pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 713–728, 2018. 1, 2

[22] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020. 1, 2

[23] Sijin Li and Antoni B Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision*, pages 332–347. Springer, 2014. 2

[24] Jonathan L Long, Ning Zhang, and Trevor Darrell. Do convnets learn correspondence? In *Advances in neural information processing systems*, pages 1601–1609, 2014. 1

[25] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 2

[26] Ross Messing, Chris Pal, and Henry Kautz. Activity recognition using the velocity histories of tracked keypoints. In *2009 IEEE 12th international conference on computer vision*, pages 104–111. IEEE, 2009. 1

[27] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020. 1, 2

[28] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European*

conference on computer vision, pages 483–499. Springer, 2016. 1, 2, 7

[29] Guanghan Ning and Zhihai He. Dual path networks for multi-person human pose estimation. *arXiv preprint arXiv:1710.10192*, 2017. 7

[30] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE, 2017. 1, 2

[31] Oren Rippel, Manohar Paluri, Piotr Dollar, and Lubomir Bourdev. Metric learning with adaptive density discrimination. *arXiv preprint arXiv:1511.05939*, 2015. 2

[32] Tanner Schmidt, Richard Newcombe, and Dieter Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, 2016. 2

[33] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015. 1, 2

[34] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016. 2

[35] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 2

[36] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656, 2015. 2

[37] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014. 2

[38] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1519, 2015. 2, 5

[39] Angtian Wang, Yihong Sun, Adam Kortylewski, and Alan Yuille. Robust object detection under occlusion with context-aware compositionalnets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 5

[40] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016. 7

[41] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(2), 2009. 2

[42] Jiajun Wu, Tianfan Xue, Joseph J Lim, Yuandong Tian, Joshua B Tenenbaum, Antonio Torralba, and William T Freeman. Single image 3d interpreter network. In *European Conference on Computer Vision*, pages 365–382. Springer, 2016. 2

[43] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018. 2, 5

[44] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3d object recognition. In *European Conference on Computer Vision*, pages 160–176. Springer, 2016. 7

[45] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *European conference on computer vision*, pages 467–483. Springer, 2016. 2

[46] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *European conference on computer vision*, pages 834–849. Springer, 2014. 1

[47] Zhengxin Zhang, Qingjie Liu, and Yunhong Wang. Road extraction by deep residual u-net. *IEEE Geoscience and Remote Sensing Letters*, 15(5):749–753, 2018. 5

[48] Xingyi Zhou, Arjun Karpur, Linjie Luo, and Qixing Huang. Starmap for category-agnostic keypoint and viewpoint estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 318–334, 2018. 1, 2, 5