Kernelized Few-shot Object Detection with Efficient Integral Aggregation

Shan Zhang*,†, Lei Wang*, Naila Murray♣, Piotr Koniusz*§†
†Australian National University  ♦University of Wollongong  §Data61/CSIRO  ♣Meta AI Research
†firstname.lastname@anu.edu.au, ♦leiw@uow.edu.au, ♣murrayn@fb.com

Abstract

We design a Kernelized Few-shot Object Detector by leveraging kernelized matrices computed over multiple proposal regions, which yield expressive non-linear representations whose model complexity is learned on the fly. Our pipeline contains several modules. An Encoding Network encodes support and query images. Our Kernelized Auto-correlation unit forms the linear, polynomial and RBF kernelized representations from features extracted within support regions of support images. These features are then cross-correlated against features of a query image to obtain attention weights, and generate query proposal regions via an Attention Region Proposal Net. As the query proposal regions are many, each described by the linear, polynomial and RBF kernelized matrices, their formation is costly but that cost is reduced by our proposed Integral Region-of-Interest Aggregation unit. Finally, the Multi-head Relation Net combines all kernelized (second-order) representations with the first-order feature maps to learn support-query class relations and locations. We outperform the state of the art on novel classes by 3.8%, 5.4% and 5.7% mAP on PASCAL VOC 2007, FSOD, and COCO.

1. Introduction

CNN object detectors [8, 29–31] require thousands of manually annotated images for training. Their performance drops during adaptation to novel classes if samples are few.

In contrast, Few-shot Learning (FSL) methods rapidly adapt to new visual concepts [39, 41, 43] but off-the-shelf FSL methods perform classification rather than Few-shot Object Detection (FSOD). As queries in FSOD contain multiple objects of various categories and FSOD detectors have to predict class labels and locations of objects in a query image, effective techniques capturing query-support similarities across multiple Regions-of-Interest (RoI) are required.

FSOD models [2, 6, 11, 12, 52, 58] are trained with so-called training episodes containing samples of common objects (i.e. base classes). Testing episodes contain support images of rare objects (i.e. novel classes) and query images in which these rare objects must be recognized/localized. Fan et al. [6] introduced into FSOD a Region Proposal Network (RPN), termed Attention RPN (ARPN). ARPN cross-correlates average-pooled features from support regions with features of the query image, which produces an attention map over the feature tensor of query image. However, average pooling (a first-order statistic) retains less information compared to higher-order statistics. PNSD [58] improves [6] by second-order pooling but is limited to so-called linear correlations (autocorrelation matrix). To address this limitation, we use kernelized covariance matrices [57] and Reproducing Kernel Hilbert Space (RKHS) kernels [38] which capture non-linear patterns. Kernels induce regularization e.g., an RBF kernel with a small (resp. large) radius captures a complex (resp. simple) decision boundary. However, as generating kernel matrices is computationally expensive, they are rarely used in detection.

We propose a novel feature representation which leverages the expressiveness and regularization capabilities of kernels, while enjoying an efficient implementation. The key to this efficiency is a novel Integral Region-of-Interest Aggregation (IRA) scheme for fast kernelization. We further accelerate IRA by count sketching [47], an unsupervised dimensionality reduction technique with a favourable property of implicitly performing feature augmentations. As the variance introduced by sketching is inversely-proportional to its size, it boosts the accuracy and computational speed, as described in Section 4. Our pipeline is shown in Figure 1. Our contributions are listed below:

i. We propose two types of kernelized representations used conjointly for FSOD that capture non-linear correlation patterns, obtained from candidate regions by Few-shot Object Detection (FSOD). As queries in FSOD contain multiple objects of various categories and FSOD detectors have to predict class labels and locations of objects in a query image, effective techniques capturing query-support similarities across multiple Regions-of-Interest (RoI) are required.

FSOD models [2, 6, 11, 12, 52, 58] are trained with so-called training episodes containing samples of common objects (i.e. base classes). Testing episodes contain support images of rare objects (i.e. novel classes) and query images in which these rare objects must be recognized/localized. Fan et al. [6] introduced into FSOD a Region Proposal Network (RPN), termed Attention RPN (ARPN). ARPN cross-correlates average-pooled features from support regions with features of the query image, which produces an attention map over the feature tensor of query image. However, average pooling (a first-order statistic) retains less information compared to higher-order statistics. PNSD [58] improves [6] by second-order pooling but is limited to so-called linear correlations (autocorrelation matrix). To address this limitation, we use kernelized covariance matrices [57] and Reproducing Kernel Hilbert Space (RKHS) kernels [38] which capture non-linear patterns. Kernels induce regularization e.g., an RBF kernel with a small (resp. large) radius captures a complex (resp. simple) decision boundary. However, as generating kernel matrices is computationally expensive, they are rarely used in detection.

We propose a novel feature representation which leverages the expressiveness and regularization capabilities of kernels, while enjoying an efficient implementation. The key to this efficiency is a novel Integral Region-of-Interest Aggregation (IRA) scheme for fast kernelization. We further accelerate IRA by count sketching [47], an unsupervised dimensionality reduction technique with a favourable property of implicitly performing feature augmentations. As the variance introduced by sketching is inversely-proportional to its size, it boosts the accuracy and computational speed, as described in Section 4. Our pipeline is shown in Figure 1. Our contributions are listed below:

i. We propose two types of kernelized representations used conjointly for FSOD that capture non-linear correlation patterns, obtained from candidate regions by Few-shot Object Detection (FSOD). As queries in FSOD contain multiple objects of various categories and FSOD detectors have to predict class labels and locations of objects in a query image, effective techniques capturing query-support similarities across multiple Regions-of-Interest (RoI) are required.

PSND [58] calls this module Hyper Attention RPN (HARPN) but FSOD-ARPN [6] calls it ARPN. We adopt the ARPN name for brevity.
ii. We equip our network with MLP units which learn the kernel hyper-parameters on the fly to adjust the learning complexity of kernelized representations to the data. We partially whiten matrices using Spectral Power Normalization [16] whose hyper-parameters are learnt via another MLP to extract the most informative features that concentrate along diagonals of kernel matrices.

iii. We redesign a Multi-head Relation Network to combine the first-order spatially-ordered features of support and query regions with the spatially orderless kernelized representations that contain higher-order statistics.

Advantages of RKHS kernelization in FSOD. We note that (i) kernels are very good at capturing non-linear relationships between feature channels of each candidate bounding box, (ii) kernels factor out spatial order while keeping rich statistics about each region, thus matching similar objects that vary in physical location, orientation, viewpoint is easy due to the shift-invariance, (iii) kernels let control the model complexity w.r.t. the region size and visual complexity, (iv) typical FSOD head uses either shift-variant or average pooled representations (we combine both).

2. Related Works

Below, we describe popular object detection and FSL algorithms and prerequisites such as second-order pooling.

Object Detection. One-stage detectors perform a regression to bounding box annotations [25, 29, 30]. Two-stage detectors, e.g., by R-CNN [31], generate class-agnostic region proposals which are then classified by a classification head [8, 31]. SNIPER [37] uses multi-scale training. DETR [1] and its variants [3, 59, 61] are anchor-free pipelines. Object detectors use large-scale datasets and fixed classes.


Few-shot Object Detection. In [11], a single-stage FSOD detector reweights base model features to adapt to new classes. Meta R-CNN [52], a two-stage detector, reweights RoI features in the detection head. Based on a balanced dataset, TFA [45] fine-tunes a two-stage model. MPSR [49] improves TFA by training over multiple scales of positive samples. NP-RepMet [53] uses negative- and positive-representative learning via triplet losses that bootstrap the classifier. FSOD-ARPN [6] proposed a FSOD network with attention and a multi-relation head to score pairwise object similarity. PNSD [58], inspired by FSOD-ARPN [6], uses second-order representations to describe proposal regions.

on the $\ell_1$-norm normalized spectrum from SVD ($\lambda_i := \lambda_i / (\sum_{\ell'} \lambda_{\ell'} + \varepsilon)$), and on symmetric positive semi-definite matrices (PSD) as
\[
\hat{G}_{\text{MaxExp}} (K; \eta) = I - (I - K)^{\eta}. \tag{2}
\]
Here, $K$ is a trace-normalized SPD matrix, $\varepsilon \geq 0$ is a small constant, and $\eta \geq 1$ is used to adjust the degree to which features are decorrelated. Larger values of $\eta$ result in greater decorrelation. As a result, rarer and more unique visual features are less overshadowed by large areas of visually-repetitive stimuli. $\hat{G}_{\text{MaxExp}}$ is followed by the element-wise PN, called SigmE [18]:
\[
\hat{G}_{\text{SigmE}} (p; \eta') = 2/(1 + e^{-\eta'p}) - 1, \tag{3}
\]
where $p$ takes each output entry of Eq. (2), $\eta' \geq 1$ controls detecting feature occurrence vs. feature counting trade-off.

**Count Sketches.** Count sketching [47] is an unsupervised dimensionality reduction technique which comes handy in reducing the size of our kernelized representations, described in Section 4. Let $K$ and $K'$ be the sizes of the input and sketched output. Let vector $h \in \mathcal{H}_K$, contain $K$ uniformly drawn integer numbers from $\{1, \cdots, K\}$ and vector $s \in \{-1, 0, 1\}^K$ contain $K$ uniformly drawn numbers from $\{-1, 0, 1\}$. The sketch projection matrix $P \in \{-1, 0, 1\}^{K' \times K}$ is given as $P_{sj}(h, s) = s_j \cdot (h_j - i)$ and the sketch projection $\text{Proj} : \mathbb{R}^K \rightarrow \mathbb{R}^{K'}$ is a linear operation $\text{Proj}_{h,s}(\phi) = P(h, s)\phi$ (or $\text{Proj}(\phi) = P\phi$). Weinberger et al. [47] showed that count sketches are unbiased estimators of the inner product $E_{h,s}(\langle \text{Proj}_{h,s}(\phi), \text{Proj}_{h,s}(\phi') \rangle) = 0$ with the variance bounded by $\frac{1}{2\pi} (\langle \phi, \phi' \rangle)^2 + \| \phi \|^2_2 \| \phi' \|^2_2$.

### 4. Proposed Approach

**Overview.** Kernelized Few-shot Object Detector (KFSOD) is trained with a set of $L$-way $Z$-shot episodes. Each episode contains a query image with objects, and $Z$ support regions (object crops) for each of $L$ sampled classes. The training protocol ensures that query objects match some support objects by label. During testing, KFSOD localizes and classifies objects in the query image given annotated support crops of novel classes. KFSOD in Fig. 1 contains:

1. Fig. 2a: Encoding Network (EN) yields conv. feature maps (stride along the channel mode is a feature vector).
2. Fig 3a & 4: Kernelized Autocorrelation (KA) unit forms two types of kernelized representations: (i) RKHS kernels and (ii) so-called kernelized autocorrelation matrices called k-autocorrelations. In practice, KA computes both types of kernelization for the linear, polynomial and RBF kernelized non-linearities from support crops and query RoIs. As there are only a few support regions per episode, we crop support regions and directly kernelize them. We use a different approach (point 4) for query images which have many more RoIs.
3. Fig. 3a: Attention Region Proposal Network (ARPN) takes kernelized feature vectors per support region to cross-correlate them against the image-wise query convolutional feature map to produce a query attention map. Region Proposal Network outputs query RoIs.
4. Fig. 3a, 3b & 4: Integral RoI Aggregation (IRA) rapidly forms inner-product matrices required to obtain RKHS kernels and k-autocorrelations for each query RoI.
5. Fig. 2b: Multi-head Relation Network (MRN) combines first-order representations (spatial-wise cues) with spatially-invariant kernelized representations to learn relations between support-query region pairs, and predict classes and locations of objects in query images.

We now describe these components in detail.

**Encoding Network (Fig. 2a).** The support crop and the query image are denoted as $X \in \mathbb{R}^{W \times H}$ and $X^* \in \mathbb{R}^{W' \times H'}$. Let $\Phi^+ \in \mathbb{R}^{K \times N}$ and $\Phi^+ \in \mathbb{R}^{K \times N'}$ be support and query maps from layer 4. Feature map $\Phi^+ \in \mathbb{R}^{K \times N}$ is used by ARPN. Let $\Phi^+ \in \mathbb{R}^{K \times 4N}$ and $\Phi^+ \in \mathbb{R}^{K \times 4N'}$ be feature maps with twice the resolution of $\Phi^+$ and $\Phi^+$. They are used for forming RKHS kernels and k-autocorrelations. (See §A of Suppl. Material for details about EN.)

**RKHS Kernels and K-autocorrelations.** Before we generate our representations, we perform the $\ell_2$-norm normalization on feature vectors $\Phi^+ \in \mathbb{R}^{K \times 4N}$. An autocorrelation matrix on $\Phi^+$ could be then computed as $K^{\text{lin}} = \frac{1}{T} \Phi^+ \Phi^T$. Let $\Phi = \Phi^T \equiv \left[ \tilde{\phi}_1, \cdots, \tilde{\phi}_K \right]$, then one could also write $K^{\text{lin}}$ as an inner-product kernel $k^{\text{lin}}(\tilde{\phi}_i, \tilde{\phi}_j)$ in practice, we firstly generate RKHS kernels listed in Table 1, e.g., $k^{\text{poly}}$ and $k^{\text{rbf}}$, by substituting the inner product $\langle \tilde{\phi}_i, \tilde{\phi}_j \rangle$ into a non-linearity $\rho : \mathbb{R} \rightarrow \mathbb{R}$ in Table 2. In addition, for the RBF kernel, one should decompose the Euclidean distance $\| \tilde{\phi}_i - \tilde{\phi}_j \|^2_2 = \| \tilde{\phi}_i \|^2_2 + \| \tilde{\phi}_j \|^2_2 - 2\langle \tilde{\phi}_i, \tilde{\phi}_j \rangle$, which becomes handy during IRA-based computations.

---

3Detailed KFSOD pipeline (all modules) is in §B of Suppl. Material.
we can define a family of kernelized autocorrelations. Let
The integral tensor in Eq. (5). Fibers (purple arrows) are multiplied
generates the so-called integral tensors for RKHS kernels and k-
are multiplied with each other. By analogy, blue fibers are
and k-autocorrelations. ARPN passes region candidates to IRA, where
produces the integral tensor in Eq. (5). Fibers (purple arrows) are multiplied
feature map to obtain the query attention map and generate
features (proposed by RPN) from $\Phi^*$ and then forming hundreds of kernel ma-
trices individually (e.g. 256 RoIs $\times$ 5 kernelized representations), we propose an efficient IRA (Fig. 4: blue block). We
form a correlation feature map by the Kronecker product along the first mode of $\Phi^*$, extracting the upper triangular$^3$
and reshaping the matrix into a three-mode feature map:

$$\mathcal{M} = \text{Reshape} \left( \text{Upper}(\Phi^* \otimes \Phi^*) \in \mathbb{R}^{\frac{1}{2}K(K+1) \times 1 \times N^v} \right),$$

where $\mathcal{M} \in \mathbb{R}^{\frac{1}{2}K(K+1) \times 2N^v_0 \times 2N^v_0}$ due to Reshape$(\cdot)$ reshaping mode 2 (size $4N^v$) into modes 2 and 3 (sizes $2N^v_0$
and $2N^v_0$). Then the so-called integral tensor is formed.

**Integral Rol Aggregation$^4$ (IRA).** For query proposal regions, we have access to a feature map $\Phi^*$ representing an entire query image. Instead of extracting RoIs (proposed by RPN) from $\Phi^*$ and then forming hundreds of kernel ma-
tices individually (e.g. 256 RoIs $\times$ 5 kernelized representations), we propose an efficient IRA (Fig. 4: blue block). We
form a correlation feature map by the Kronecker product along the first mode of $\Phi^*$, extracting the upper triangular$^3$
and reshaping the matrix into a three-mode feature map:

$$\mathcal{M} = \text{Reshape} \left( \text{Upper}(\Phi^* \otimes \Phi^*) \in \mathbb{R}^{\frac{1}{2}K(K+1) \times 1 \times N^v} \right),$$

where $\mathcal{M} \in \mathbb{R}^{\frac{1}{2}K(K+1) \times 2N^v_0 \times 2N^v_0}$ due to Reshape$(\cdot)$ reshaping mode 2 (size $4N^v$) into modes 2 and 3 (sizes $2N^v_0$
and $2N^v_0$). Then the so-called integral tensor is formed.

**Integral tensor** is obtained by multiplying $\mathcal{M}$ along modes 2 and 3 by the lower and upper triangular ma-
trices, $R_{\Delta}, i_j = 1$ if $i \geq j$ (0 otherwise) and $R_{\Delta}^\text{c} = 1$ if $i \leq j$ (0 otherwise), in order to obtain the integral tensor

$$\mathcal{M}^\Delta = \mathcal{M} \times_2 R_{\Delta} \times_3 R_{\Delta}^\text{c}, \quad (5)$$

where symbols $\times_2$ and $\times_3$ are tensor-tensor multiplications along modes 2 and 3 [14].

Figure 3b illustrates the formation of integral tensor by considering each channel of $\mathcal{M}$ separately. Let $\mathcal{M} \equiv [M_{i\ell}]_{\ell \in \mathbb{I}_{K(K+1)/2}}$ and $\mathcal{M}^\Delta \equiv [M^\Delta_{i\ell}]_{\ell \in \mathbb{I}_{K(K+1)/2}}$. Then

$$\forall \ell, \quad M^\Delta_{i\ell} = R_{\Delta} M_{i\ell} R_{\Delta}^\text{c}, \quad \text{or simply,} \quad M^\Delta_{i\ell} = \sum_{i',j' \leq \ell} M_{i'i'} R_{i'i'}. \quad (6)$$

**Kernelize region representation** with top-left $(x, y)$ and bottom-right $(x', y')$ locations, $x \leq x', y \leq y'$, is extracted with a negligible cost (one addition/two sub-
tractions) by:

$$\mathbf{K}^{(\rho)}((x, y), (x', y')) = \zeta(\rho(\mathbf{K})) \quad \text{where} \quad (6)$$

$$\mathbf{K} = \text{Deploy}(\mathcal{M}^\Delta_{i'i'}, y_{i'y}, -\mathcal{M}_{i'i'-x} - \mathcal{M}_{i'i'-y} + \mathcal{M}_{i'i'-x'-y} + \mathcal{M}^\Delta_{i'i'-x'-y'})$$

Deploy$(\cdot)$ in Eq. (6) deploys the vector containing the upper
upper triangular (plus diagonal) back into the corresponding

---

$^4$Viola and Jones [44] explain basics of integral images.

$^3$Note that $\phi \otimes \phi = \text{Vec}(\phi \phi^T) \in \mathbb{R}^{K^2}$ where Vec$(\cdot)$ vectorizes the ma-
trix. We apply $\otimes$ to feature vectors $\phi$ of $\Phi^*$ in Eq. (4). We discard redundant
diagonal lower triangular by Upper$(\phi \otimes \phi) \equiv \text{Upper}(\phi \phi^T) \in \mathbb{R}^{(K+1)/2}$
that extracts the upper triangular+diagonal from the matrix & vectorizes it.
Integral RoI Aggregation with Count Sketching. Eq. (4) and (8) apply the costly Kronecker product on $K$ dimensional feature vectors. To reduce its $O(K^2)$ complexity to $O(K'^2)$, $K' \ll K$, we apply count sketching [47] via the unitary projection matrix $P \in \{−1, 0, 1\}^{K' \times K}$ which enjoys a simple pseudoinverse $P^\dagger = \frac{K'}{K} P^T$. Eq. (4) and (8) become

$$M = \text{Reshape} \left( \text{Upper} \left( \left( \rho (\Phi^T) \otimes (\Phi^*) \right) \right) \right) \quad \text{and} \quad (10)$$

$$M' = \text{Reshape} \left( \text{Upper} \left( \sum_{i \in \mathcal{I}_r} (P \omega_i (\Phi^*)) \otimes (P \omega_i (\Phi^*)) \right) \right),$$

with the kernelized region representations recovered by

$$K^{(r)}(x, y, (x', y')) \approx \zeta (P^\dagger K P^T), \quad (11)$$

$$K'^{(r)}(x, y, (x', y')) \approx \zeta (P^\dagger K' P^T). \quad (12)$$

The above is true as $F \Phi^T \approx P \Phi^T = P^T \Phi^*$ so inverting sketching yields $P \Phi^T (P^T)^\dagger \Phi^* \approx P^T \Phi^*$.

Count sketching implicitly performs a feature-level augmentation akin to injecting the Gaussian noise into features [46]. (Proof is in §C of Suppl. Material.)

- Notice $\langle P \Phi, P \phi' \rangle = \langle P^T P \Phi, \phi' \rangle = \langle \frac{K'}{K} P^T P \Phi, \phi' \rangle$. If $\|\phi\|_2 = \|\phi'\|_2 = 1$ then based on properties of count sketches in §3, we have

$$\langle \frac{K}{K'} P^T P \Phi, \phi' \rangle = \langle P \Phi, P \phi' \rangle \sim N\left( (\langle \phi, \phi' \rangle, \sigma^2) \right), \quad (13)$$

where $\langle \phi, \phi' \rangle$ is a point-wise convolution of feature vector $\phi$ with convolutional filter $\phi'$. It follows that $\langle \frac{K}{K'} P^T P \Phi, \phi' \rangle$ realizes a point-wise noisy convolution whose variance $\sigma^2 = \frac{1}{K'} (\langle \phi, \phi' \rangle + 1)^2 = \frac{2}{K'}$.

- Injecting the Gaussian noise [46] can be characterized as $\langle \phi + \Delta \phi, \phi' \rangle$ where $\Delta \phi \sim N(0, \sigma^2)$ which leads to

$$\langle \phi + \Delta \phi, \phi' \rangle \sim N\left( (\langle \phi, \phi' \rangle, \sigma^2) \right).$$

Computational Complexity of IRA. Computing dot-product based kernels naively (from $\Phi^*$) has the complexity $O(K^2 N^* B)$, where $K$ is the number of features (channels) and $N^*$ is the average area of 256 query proposals. Computing these kernels via IRA has the complexity $O(K^2 N^* \frac{2}{K} + K^2 B)$ (the first and second terms concern forming the integral tensor and extracting $B$ kernels from it). If $N^* \ll (N^* B)^\frac{1}{2}$, computing kernels via IRA is faster.

The cost of computing the Kronecker product in Eq. (4) is $O(K^2 N^*)$ which reduces to $O(KK'N^* + K^2 N^*)$ for the sketching variant in Eq. (10) (top) and $O(KK'N^* + r'K'K^2 N^*)$ (bottom). If $K' = 0.5 K$, the cost is reduced by $4x$.

Pooling Kernelized Representations. Using matrices $K$ of dimensions $K \times K$ as feature representations is prohibitively expensive given hundreds of RoIs. Thus, we pool $K$ into a $K$-dimensional representation. Table 3 lists three pooling operators we consider: first-order (mean) pooling with PN,
denoted as FO+PN, Second-order Spectral Diagonal Correlation with PN (SOSD+PN), and Second-order Self Correlation with PN (SOSC+PN). We use SOSD+PN (standard second-order pooling) and ablate other operators in Sec. 5.

**Extraction of RKHS Kernels and K-autocorrelations.** As shown in Figure 4, given B query RoIs and the feature maps for a support crop (Φ) and query image (Φ*), we generate 5 kernelized matrices from Φ, and B × 5 kernelized matrices from Φ*. Each set of 5 kernelized matrices correspond to (linear, polynomial and RBF) RKHS kernels, and 2 (polynomial and RBF) k-autocorrelation matrices. Each kernelized matrix is computed on 1 of 5 groups of channel features, where the groups were created by splitting (denoted by ⊙ in Figure 4) the channel with size K = 1024 into 1 group with size 256 and 4 groups each with size 192.

To set λ and σ−2, the hyper-parameters of polynomial and RBF RKHS kernels, we predict them using a trained layer MLP[K(Φ)lin · 1], where MLP contains an FC layer followed by the sigmoid function. For k-autocorrelations, we first take the mean over the spatial modes of Φ and Φ*. We split the resulting vectors μ and μ* into groups, as explained above, and feed them into an MLP to generate σ−2 and λ*. Parameter η′ of PN in Table 3 is predicted by a different MLP that uses the output of OSOD as its input. As sigmoid outputs are in range (0, 1), for RBF kernels, polynomial kernels and PN, we scale them into (0, 1) ranges. Figure 4 shows the extraction procedure.

**Multi-head Relation Network.** MRN learns the similarity score, top-left and bottom-right bounding box coordinates between support-query pairs represented by (i) first-order spatially-aware (Ψ, {Ψb}b∈IB) which are feature maps in R(1024:16:14×14), and (ii) second-order spatially-invariant pooled kernelized representations (Ψ′, {Ψ′b}b∈IB) which are vectors in R(1024). Notice we have B query candidate regions.

The Multi-head Relation Net in Figure 2b contains 3 subheads: (i) global head, (ii) local head and (iii) patch head. Global and local heads combine first-order spatially-aware and second-order spatially-invariant representations to learn the similarity. The patch head takes first-order spatially-aware representations to perform the bounding box regression. See §D of Suppl. Material for more details.

Let s{(Ψ, {ψb}b∈IB), (Ψ′, {ψ′b}b∈IB); S}→{(y, x)b}b∈IB. S are network parameters, (y, x) ∈ Y contains similarity prediction, and top-left/bottom-right coordinates of candidate regions b ∈ IB. For the L-way Z-shot problem, we have L × Z support image regions {Xn}n∈U from set U and their corresponding descriptors {(Φn, Φ)n∈U} from the Encoding Network (Fig. 2a). We compute L representations Ψ and Ψ′. Let X* be a query image with its query feature maps {(Φ+b, Φ*)n∈IB} obtained from B proposal regions from ARPN (Fig. 3a). Representations (Φ+b, Φ) and (Φ+b, Φ*) are passed to the Kernelized Block (Fig. 3) whose output representations (Ψ, Ψ′) and (Ψ+b, Ψ′+b) are passed to Multi-head Relation Net to minimize the loss:

\[ \sum_{(l,b)\in IB} l_{\text{sim}}(y^l_b, y^l_b) + l_{\text{box}}(x^l_b, x^l_b) + l_{rpn}^t(Ψ'_{n}×Φ+b, Ψ'_{n}×Φ^+\cdot Φ^+) \] (14)

where query-support pairs belong to L classes in the subset \( C^+ \equiv \{c_1, \ldots, c_L\} \subset IC \equiv C \). Loss functions \( l_{\text{box}} \) and \( l_{rpn}^t \) follow [31], and \( l_{\text{sim}} \) is the binary cross-entropy, \( \mathcal{H} \) are parameters of ARPN (same with [31]) and \( \cdot \times \cdot \) performs channel-wise cross-correlation in \( \Psi'_{n}×Φ^+\cdot Φ^+ \in \mathbb{R}^{1024×16×14} \).

5. Experiments

**Datasets and settings.** For PASCAL VOC 2007/12 [5], we adopt the 15/5 base/novel category split setting and use training/validation sets from PASCAL VOC 2007 and 2012 for training, and the testing set from PASCAL VOC 2007 for testing [11]. For MS COCO [24], we follow [52], and adopt the 20 categories that overlap with PASCAL VOC as the novel categories (testing). The remaining 60 categories are used for training. For the FSOD dataset [6], we split 1000 categories into 800/200 for training/testing. We report standard FSOD metrics: \( mAP, AP, AP_{50} \) and \( AP_{75} \).

**Implementation details** are in §H and hyper-parameters for each dataset are in §I of Suppl. Material.

5.1. Comparisons with the State of the Art

**PASCAL VOC 2007/12.** We compare KFSOD to FSOD [48], CGDP+FRCN [23], TIP [21], FSCE [40], TFA [45], Feature Reweighting (FR) [11], LSTD [2], FRNC [31], NP-RepMet [53], MPSR [49], PSND [58] and FSOD [6]. Table 4 shows that KFSOD outperforms FSOD by a 6.3–10% margin. For the 1- and 10-shot regime, we outperform FSOD by ~2.2%. Table 9 (§E of Suppl. Material) shows class-wise results (5-shot protocol): KFSOD gains 11.2% and 6.5% mAP (novel and base classes) over FSOD.

**MS COCO.** Table 5a compares KFSOD vs. FSOD [48], CGDP+FRCN [23], TIP [21], FSCE [40], TFA [45], FR [11], Meta R-CNN [52], FSOD [6] and PNSD [6] on MS COCO minival set (20 novel cats., 10-shot). KFSOD outperforms FSOD by 6.9%, 2.4%, 8.9% (AP, AP_{50}, AP_{75}).

**FSOD.** In Table 5b we compare KFSOD (5-shot protocol) with PNSD [58], FSOD [6], LSTD [2] and LSTD (FRN [31]). We re-implement BD&TK, modules of LSTD, based on Faster-RCNN for a fair comparison. KFSOD gives SOTA results of 33.4% AP_{50} and 29.6% AP_{75}.

5.2. Ablation studies

Below we use PASCAL VOC (novel classes, split 1, 5-shot setting, hyper-parameters selected on the val. split).

<table>
<thead>
<tr>
<th>FO+PN</th>
<th>SOSD+PN</th>
<th>SOSC+PN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ψ_{\text{signal}}(\Phi_{\text{lin}}; \eta')</td>
<td>Ψ_{\text{signal}}(\text{Diag}(\tilde{g}_{\text{maxpool}}(K; \eta)); \eta')</td>
<td>Ψ_{\text{signal}}(\tilde{K}_{\text{lin}}; \eta')</td>
</tr>
</tbody>
</table>

Table 3. Pooling operators are applied (i) prior to cross-correlation in ARPN and (ii) to represent support and query RoIs.
Table 4. Comparison of different methods in terms of mAP (%) on three splits on the VOC 2007 testing set.

Table 5. Comparison with SOTA on the MS COCO minival set and FSOD test set, given in Tables 5a and 5b.

Table 6. Results with combinations of kernelized representations on FSOD and COCO dataset (5/10-shot protocol) are in Table 6a. Table 6b shows mAP on PASCAL VOC 2007 (5-shot, novel classes) of the linear kernel formed from low resolution $\Phi^+$ vs. high-resolution $\Phi$ feature maps (Fig. 2a).

Different backbones. See §G of Suppl. Material.

Performance of individual kernels. Firstly, we compare RKHS kernels with k-autocorrelations in a manual hyper-parameter setting. We set $\eta = 5$ and $\eta' = 100$ of SOSD+PN kernel pooling. We denote linear, RBF and polynomial RKHS kernels as Lin/k, RBF/k and Poly/k. We refer to RBF and polynomial k-autocorrelations as RBF/a and Poly/a. Fig. 5a shows that RBF/k ($\sigma = 2.0$) outperforms Lin/k by 2.1% on the validation split. However, RBF/k+MLP ($\kappa = 1.8$) outperforms RBF/k by 2% in both validation and test splits. Fig. 5b shows that RBF/a ($\sigma' = 0.2$) outperforms Lin/k by $\sim$0.2%. However, RBF/a+MLP with the sigmoid scaling parameter $\kappa' = 1.3$ outperforms Lin/k by $\sim$2% on both validation and test splits. Fig. 5c shows that Poly/k ($r = 5$) outperforms Link/k on both validation and test splits. Finally, Fig. 5d shows that Poly/a+MLP ($\kappa' = 8$) outperforms Poly/a ($r' = 5$) and Link/k. Fig. 5b & 5c show that MLP adjusts parameters of kernels on the fly in a very stable way (maxima match on val. and testing splits). On FSOD and COCO (Table 6a), combining kernels improves results over Link/k by $\sim$3%. See §F of Suppl. Material for ablations on combinations of kernels.

Kernel pooling. We evaluate pooling from Table 3 w.r.t. $\eta$ of Spectral Power Normalization (SPN) and $\eta'$ of elementwise PN, which we learn by MLP with sigmoid scaling $\kappa''$.
Table 7. In Tab. 7a is the runtime (PASCAL VOC 2007) in seconds per 1000 images (training vs. inference time) w.r.t. B (the number of proposal regions). No count sketching was used. In Tab. 7b is the runtime in seconds per 1000 images. Count sketching was used by IRA+KA. Compression ratio $\frac{B}{\eta}$ that gave best results was set to $2\times, 4\times$ and $8\times$ on PASCAL VOC 2007, FSOD, and COCO.

of matrices which discards off-diagonal correlations.

Fig. 6a shows that the best combined validation performance is attained by both RKHS-based RBF/k+MLP and k-autocorrelation RBF/a+MLP for either $\eta = 5$ or $\eta = 9$. Fig. 6b also shows that if we combine the validation performance of Poly/k+MLP and Poly/a+MLP, either $\eta = 5$ or $\eta = 9$ are good choices. Thus we set $\eta = 9$ in all our experiments. Fig. 6c verifies the choice of $\eta = 9$ on Lin/k and the combination of all 5 kernelized representations.

Fig. 6d evaluates pooling operators from Table 3. Manually setting $\eta'$ of FO+PN outperforms FO by $\sim 2\%$. Using all kernels with SODS+PN pooling outperforms FO by $\sim 5\%$. SODS+PN+MLP with $\eta'$ adjusted by the MLP with $\kappa''$ outperforms FO by $\sim 8\%$. Table 8 ablates first-order vs. kernelized features. ARPN uses first-order+PN (FO) or all kernels (All). MRN head uses only first-order inputs $\Psi, \Psi^*$ (FO), kernelized inputs $\Psi^*, \Psi^{**}$ (All), linear kernel (Lin/k) or PSND [58] (second-order matrix).

Performance vs. speed (IRA+KA). In Fig. 7a, for KSOD and the injection of Gaussian noise, we chose the best $\sigma^2$ (defined below Eq. (13)) in range $\langle 0.001, 0.1 \rangle$, and we reduce the feature map size by the factor of $K'/K$ indicated on x-axis ($K'$ is not used). For KSOD+Sketching, we set $K' = 8$ (Section 3 defines $K'$). We indicate time in hours (all methods achieve similar speed-up) on x-axis. As is clear, KSOD+Sketching achieves gain of $\sim 1.8\%$ over KSOD. Fig. 7b shows that the best performance of KSOD+Sketching on COCO is achieved for drawing a new sketch matrix $P$ every 0.4 epoch, and the best compression ratio is $\frac{K}{K'} = 8$. Table 7a shows that IRA is very beneficial as $B$ (the number of candidate regions) grows. Table 7b shows almost $3\times$ speed-up compared to naive kernel computations on COCO.

6. Conclusions

We have proposed RKHS kernels and k-autocorrelations into FSOD. In order to accelerate the computation of region representations (the number of query candidate regions is large), we propose a novel Integral Region-of-Interest Aggregation (IRA) scheme combined with count sketching. On COCO, KSOD with IRA and count sketching takes 38.5h instead of 85.2h. In addition to accelerating the computation of region representations, count sketching performs controlled feature augmentation due to its bounded noise, akin to feature augmentation by noise injection, leading to improved detection accuracy.

Figure 7. Performance (mAP%) of KFSOD, KFSOD+Gaussian noise and KFSOD+Sketching w.r.t. the compression ratio is in Fig. 7a (PASCAL VOC 2007, split 1, novel classes, 5-shot). mAP% of KFSOD with count sketching w.r.t. the frequency of drawing of sketch matrix $P$ is in Fig. 7b (COCO, novel classes, 10-shot).

\[\tau = (\text{defined below Eq. (13)}) \in \langle 0.001, 0.1 \rangle, \text{and we reduce the feature map size by the factor of } \frac{K'}{K} \text{indicated on x-axis (}K'\text{ is not used). For KFSOD+Sketching, we set } K' = 8 \text{(Section 3 defines } K') \text{. We indicate time in hours (all methods achieve similar speed-up) on x-axis. As is clear, KFSOD+Sketching achieves gain of } \sim 1.8\% \text{ over KFSOD. Fig. 7b shows that the best performance of KFSOD+Sketching on COCO is achieved for drawing a new sketch matrix } P \text{ every 0.4 epoch, and the best compression ratio is } \frac{K}{K'} = 8. \text{Table 7a shows that IRA is very beneficial as } B \text{ (the number of candidate regions) grows. Table 7b shows almost } 3\times \text{ speed-up compared to naive kernel computations on COCO.} \]
References


