Object-Level Targeted Selection via Deep Template Matching

Suraj Kothawade ∗
University of Texas at Dallas
suraj.kothawade@utdallas.edu

Donna Roy
NVIDIA
donnar@nvidia.com

Michele Fenzi
NVIDIA
mfenzi@nvidia.com

Elmar Haussmann
NVIDIA
ehaussmann@nvidia.com

Jose M. Alvarez
NVIDIA
josea@nvidia.com

Christoph Angerer
NVIDIA
cangerer@nvidia.com

Abstract
Retrieving images with objects that are semantically similar to objects of interest (OOI) in a query image has many practical use cases. A few examples include fixing failures like false negatives/positives of a learned model or mitigating class imbalance in a dataset. The targeted selection task requires finding the relevant data from a large-scale pool of unlabeled data. Manual mining at this scale is infeasible. Further, the OOI are often small and occupy less than 1% of image area, are occluded, and co-exist with many semantically different objects in cluttered scenes. Existing semantic image retrieval methods often focus on mining for larger sized geographical landmarks, and/or require extra labeled data, such as images/image-pairs with similar objects, for mining images with generic objects. We propose a fast and robust template matching algorithm in the DNN feature space, that retrieves semantically similar images at the object-level from a large unlabeled pool of data. We project the region(s) around the OOI in the query image to the DNN feature space for use as the template. This enables our method to focus on the semantics of the OOI without requiring extra labeled data. In the context of autonomous driving, we evaluate our system for targeted selection by using failure cases of object detectors as OOI. We demonstrate its efficacy on a large unlabeled dataset with 2.2M images and show high recall in mining for images with small-sized OOI. We compare our method against a well-known semantic image retrieval method, which also does not require extra labeled data. Lastly, we show that our method is flexible and retrieves images with one or more semantically different co-occurring OOI seamlessly.

1 Introduction
Retrieving images with objects that are semantically similar to objects of interest (OOI) in a query image, at a large-scale, is a fundamental task in computer vision. It has many practical applications, e.g. i) targeting consistent failure cases of trained models, ii) targeting class imbalance in object detection datasets where the imbalance can be due to various attributes like class, object size, etc. Such problems can be solved by adding relevant samples to training data. Our inspiring motivation was driven by targeted selection at an object-level, where the failure cases like false negatives/positives in production level object detectors for autonomous driving vehicles (AV) are used as OOI. These

∗Work done while the author was at NVIDIA.

failure cases occur across different driving scenarios and conditions, and are often due to under-
representation of such data in the training dataset. On the other hand, collecting unlabeled AV data at
a large-scale is easily accomplished even with a small fleet of cars. But, manual mining for relevant
data at this scale is not feasible.

Thus, the task of object-level targeted selection involves automatically finding the relevant data
with objects that are semantically similar to the OOI from a large-scale pool of unlabeled data, as shown in Fig. 1. Further, these OOI are often small-sized and occupy less than 1% of image area, are occluded, and co-exist with many semantically different objects in real-world cluttered scenes which makes the mining task chal-
lenging.

We formulate the object-level targeted selection task as a semantic sub-image retrieval problem
which takes as input a small set of query images (e.g., a few tens of query images) with bounding box annotations around one or more OOI in the scene and finds images from a large pool of unlabeled data with semantically similar objects. These annotations form the region of interest (ROI) in the query images. Typically, examples of these ROIs are objects which the object detector systematically failed to detect, e.g., motorcycles at night, bicycles mounted on cars, etc. Since the ROI is used as a template for search, and we use deep features to represent the template and image search space, we name our approach Deep Template Matching (DTM).

In the last decade, significant advances have been made in large-scale semantic image retrieval
systems - from handcrafted features and indexing algorithms [20, 27, 33, 38] to more recent methods
based on DNNs for global descriptor learning [2, 3, 4, 14, 16, 22, 34, 35, 41, 44, 43].

The recent advances in DNN-based global descriptors for image retrieval tend to focus on mining for
larger-sized geographical landmarks. Their performance is hindered in some challenging conditions
observed in real-world AV datasets, such as small-sized objects of interest, occlusion, and heavy scene clutter.

Various methods have been proposed in the literature to overcome this limitation, such as instance-
level image retrieval methods in [41, 16, 15, 26, 44, 38] which tend to be computationally expensive,
making DNNs with attention modules in [29, 6, 24, 28] and deep metric learning [11, 1, 18, 46, 50]
and neural graph [21] based methods which require extra labeled data to explicitly train the mining
DNNs. Another line of related work is template matching techniques in [10, 11, 13, 23, 25, 30, 40, 45]
which solve the problem of finding a template patch in a sample image. However, these methods
typically use low-level image features and are usually susceptible to geometric transformations
(like rotation, translation, scale, etc.), illumination changes, occlusions, and background clutter. We
describe related work in more details in section 2.

**Our main contributions:** In this paper, we propose Deep Template Matching (DTM) - a fast and
robust template matching algorithm in the DNN feature space, that retrieves semantically similar
images at the object-level from a large unlabeled pool of data. We solve this by projecting the region(s)
around the objects of interest in the query image to the DNN feature space for use as the template,
and then computing a linear-time one-shot similarity score in the deep feature space. This enables
our method to focus on the semantics of the objects of interest, even when the object is small-sized,
and occlusion and heavy scene clutter, without requiring extra labeled data, and is computationally
cheap. We demonstrate the effectiveness of our method on a labeled dataset with 465k images and a
large unlabeled dataset with 2.2M images. We compare DTM against a well-known semantic image
retrieval method [3] which also does not require extra labeled data. Lastly, we illustrate the flexibility
of DTM for multiple queries without increasing computational complexity.
**DTM has the following characteristics:** 1) **Accuracy**: High recall values for queries with one or more objects of interest (i.e., templates) of any size and multiple semantic categories in real-world scenes with occlusion and heavy clutter. 2) **Computational efficiency**: Quick mining time on the GPU which is essential for fast experimental turnaround time in production-level pipelines. DTM allows precomputing the image embeddings in DNN feature space offline, to speed up the scoring and search later. The computational complexity of DTM does not depend on the number or semantic category of the objects of interest in the query image. 3) **Multi-template search in one shot**: Mining using multiple objects of interest belonging to one or more semantic categories co-occurring in the sample image is done seamlessly in one shot for the same computational complexity as a single object of interest. 4) **Ability to preserve object size semantic information**: DTM mines for semantically similar samples of the same size as the query image ROI by zeroing out irrelevant features and maintaining the dimensionality of the original feature map (see Fig. 6a). Mining small objects from unlabeled data is critical, since many object detection datasets lack labels for small objects.

Our experiments show that DTM can successfully find images with semantically similar objects even when the objects of interest in the query image are quite small in size (e.g., occupies less than 0.3% of area in input 2MP image), and there is occlusion and heavy scene clutter with multiple kinds of other objects, e.g., busy streets with many cars, pedestrians, etc., in the query and unlabeled pool of images. As discussed in section 2 existing methods in the literature to solve this are either computationally expensive or require additional labeled training data. DTM is computationally cheap, works seamlessly for multiple co-occurring objects of the same or different semantic category, and does not require additional labeled training data.

### 2 Related Work

The following two problems are closely related to object-level targeted selection.

**Large-scale semantic image retrieval systems.** Early retrieval systems used handcrafted features and indexing algorithms [20, 27, 33, 38]. More recently, methods based on DNNs for global descriptor learning [2, 3, 4, 14, 16, 22, 34, 35, 41, 44, 43] have performed quite well in the literature. Typically, they use DNNs pre-trained on ImageNet as deep feature extractors, and focus on designing image representations suited for image retrieval on top of these features. These approaches fail when (i) the object of interest in the query image is small, or (ii) the scenes are heavily cluttered with multiple objects. Since these are very typical in real world highway and urban driving scenarios, such approaches did not work well for our AV use case. We implement the approach described in [3] as baseline and demonstrate better performance of DTM over [3] on our AV data.

Some recent DNN based local features have been proposed for patch-level matching [17, 47, 49]. However, these techniques focus on low-level image features like texture, and geometric information. They do not detect semantically meaningful features at the object-level.

In another line of work, [41, 16, 15, 25, 44, 48] apply an instance-level image retrieval scheme to find object/landmark instances among cluttered scenes. These methods focus on finding multiple viable region proposals in a single image, extracting deep local feature descriptors for each proposed region, and then aggregating these region-based local descriptors into one global descriptor for the entire image. Searching for viable region proposals and computing deep feature descriptors for each proposed region within a single sample image is non-trivial and computationally expensive.

Approaches in [29, 6, 24, 28], solve the problem of finding object/landmark instances in occluded/cluttered scenes by explicitly training the image retrieval DNN with labeled data to learn an attention module. These methods require extra labeled data for training the mining DNN, ranging from image-level annotations in [29, 6] to bounding box level annotations in [28]. Recent advances in deep metric learning [1, 5, 7, 8, 46, 50] and neural graphs [21] also focus on this problem, but they also require extra labeled data either in terms of pairs or triplets of similar and dissimilar scenes or to populate the neural graph. DTM does not need extra labeled data, which is advantageous, since labeling is an expensive human-in-the-loop operation.

**Template Matching.** Another line of related work is template matching, which solves the problem of finding a template patch in a candidate window in the sample image.

Classical template matching techniques [9, 12, 18, 31] use sum-of-squared-differences (SSD) or normalized cross correlation (NCC) as the similarity measure in the input image feature space,
and are quite sensitive to any variation in illumination and noise. More robust measures such as M-estimators\cite{9,37} or hamming based distance \cite{36,32} have been studied. However, all of these methods are not robust to real-world scenes since they account for only a strict rigid geometric transformation (only translation) during the measure computation. Several works in the parametric family attempt to overcome this problem. The approach in \cite{25} uses 2D affine transformation to account for geometric differences between the template and the sample image. \cite{42} addresses non-rigid transformations by parametric estimation of the distortion. These methods are susceptible to noise, occlusion and clutter. Moreover, they use a parametric approach, which is not required by DTM.

More recently, several robust non-parametric template matching approaches were suggested to mitigate noise, occlusions and degradation. \cite{11,40} use Nearest-Neighbor (NN) matches between features of the template and a sample image. The Best Buddies-Similarity measure in \cite{11} focuses on nearest-neighbor matches to remove bad matches due to background pixels. The Deformable Diversity Similarity (DDS) in \cite{40} considers possible template deformations and uses the diversity of nearest neighbors feature matches. These methods are susceptible to noise, occlusion and clutter. Moreover, they use a parametric approach, which is not required by DTM.

3 Deep Template Matching

In this section, we present and discuss the architecture and scoring mechanism of DTM in detail.

3.1 Architecture

Figure 2 describes the end-to-end architecture of DTM. It takes as input a query image $I_Q$ with a bounding box around object(s) of interest (i.e., region of interest) and a set $A$ of unlabeled sample images. Let $I_Q'$ be the ROI in the query image. The goal is to find images $I_S$ from the set $A$ of unlabeled images which have regions semantically similar to the region of interest $I_Q'$. For simplicity, let $I_Q$ be a single query image with a single ROI $I_Q'$ and $I_S$ be a single sample image. We extract the features $F_Q, F_S \in \mathbb{R}^{w \times h \times c}$ for $I_Q$ and $I_S$ respectively, using a pre-trained object detector DNN, where the dimensions are denoted as, $w$: width, $h$: height and $c$: number of channels. Next, we project $I_Q'$ onto $F_Q$ to obtain $F_Q' \in \mathbb{R}^{w \times h \times c}$. We do so by linearly mapping the ROI $I_Q'$ onto $F_Q$ and zeroing out features that do not have a one-to-one correspondence with $I_Q'$ (see Fig. 2). Note that $F_Q'$ is the final representation of the template, which is used to compute similarity with all $F_S \in A$. Before using these features for computing similarity, we L2-normalize them along the channel dimension $c$. We discuss the benefits of doing this normalization in the next section.

3.2 Patchwise Similarity Computation

In this section, we focus on computing the similarity matrix $S$ in Fig. 3. Consider feature tensors $F_Q'$ and $F_S$, where each spatial location is denoted by $F_Q'[i, j]$ and $F_S[x, y]$ respectively (see Fig. 3). We
define each of these tensors of depth \( c \) as a *patch* because they map to a certain region (patch) in the original image. Hence, both \( F_Q' \) and \( F_S \) have \( w \times h \) *patchwise* feature vectors, each of length \( c \).

Next, we compute patchwise cosine similarity scores between \( F_Q' \) and \( F_S \) by computing the similarity between \( F_Q'[i, j] \) and \( F_S[x, y] \) for all values of \( x, i \in [0, w - 1] \) and \( y, j \in [0, h - 1] \). Note that \( F_Q'[i, j], F_S[x, y] \in \mathbb{R}^c \). Strictly, the similarity score between the zeroed patches in \( F_Q' \) and \( F_S \) need not be computed, as it does not affect the final score in Equ. (4).

Figure 3: Patchwise Cosine Similarity.

Intuitively, the goal is to find the score of the best matching sample patch feature for each query patch feature \( F_Q'[i, j] \), at spatial location \((i, j)\). To compute this score, we use cosine similarity due to its computational efficiency. We require a score between every patch in the query image to every patch in the sample image, and compute the cosine similarity along the channel dimension. This results in a 4-D patchwise cosine similarity tensor \( S \in \mathbb{R}^{w \times h \times w \times h} \) which is computed according to

\[
S(F_S, F_Q') = (F_S \cdot F_Q', \text{axis} = c) \tag{1}
\]

At a patch level, each element in \( S \) at location \([x, y, i, j]\) is computed as

\[
S(F_S, F_Q')[x, y, i, j] = \frac{F_S[x, y] \cdot F_Q'[i, j]}{|F_S[x, y]|_2|F_Q'[i, j]|_2} \tag{2}
\]

A larger \( S(F_S, F_Q')[x, y, i, j] \) value indicates that sample patch feature \( F_S[x, y] \) and query patch feature \( F_Q'[i, j] \) are more similar. Note that the tensor \( S[x, y, :,:] \) stores similarity scores between sample patch feature \( F_S[x, y] \) and all query patch features, and \( S[::, i, j] \) stores similarity scores between query patch feature \( F_Q'[i, j] \) and all sample patch features.

Computing \( S \) is highly efficient due to the following reasons: 1) Both \( F_Q' \) and \( F_S \) are L2 normalized which means that \(|F_S[x, y]|_2 \cdot |F_Q'[i, j]|_2 = 1\). Hence, the cosine similarity in Equ. (2) boils down to computing a dot product along the channel dimension which is highly parallelizable on the GPU/CPU using off the shelf operations\(^2\) and 2) \( F_Q' \) has all zeros except the ROI which makes it sparse thereby enabling the advantages of sparse matrix multiplications.

3.3 Scoring Method

For computing the score of the best matching sample patch feature for each query patch feature, we use a max over the cosine similarity scores stored in \( S \). This gives a score map \( M_Q' \in \mathbb{R}^{w \times h} \) which can be computed with

\[
M_Q' = \max_{\text{axis}=(0, 1)} S(F_S, F_Q') \tag{3}
\]

\(^2\)See tensorflow.tensordot, tensorflow.math.reduce_max, tensorflow.math.reduce_mean

Figure 4: Heat map visualization of \( M_Q' \) score map. Left: Input query image with region of interest Right: Top-1 semantically similar image using DTM with overlaid heat map.
where, $A'_Q \in \mathbb{R}^{w \times h}$ is a normalization constant that is proportional to the area of the individual projected ROI(s), i.e., each spatial location in $A'_Q$ has a positive value equal to the area of its corresponding ROI in $F'_Q$ (if any). $M'_Q$ can be biased towards objects with larger ROIs in cases where the query image template has multiple ROIs with objects of different sizes occupying different ROI areas. We counter this with the normalization constant $A'_Q$. Each element of $M'_Q[i, j]$, where $i \in [0, w - 1]$ and $j \in [0, h - 1]$, indicates the best matching score found between the patch feature $F'_Q[i, j]$ and any patch feature in $F_S$, normalized by the area of the ROI at spatial location $(i, j)$, which is denoted by $A'_Q[i, j]$.

Using the score map $M'_Q$, we can compute the final score between query image $I_Q$ and sample image $I_S$ by averaging over the best patchwise similarity scores as

$$Score(I_Q, I_S) = Mean(M'_Q)$$

(4)

Note that in $Score(I_Q, I_S)$ in Equ. (4), we are not exploiting spatial relationships across features within the template. Computing scores that exploit spatial relationships explicitly tend to have higher runtime complexity \[39, 40, 11\] while not yielding significant retrieval recall gains. Our score map $M'_Q$ accurately matches the semantics of the ROI in the query image to the object in the sample image. Figure 4 depicts patches that are being matched from the query image to the sample image.

4 Experiments

4.1 Experimental Setup

We use internal research datasets for our experiments - one with 465k labeled images and another with 2.2M unlabeled images. We use a small representative set of 36 query images where the ROI in each query image is a motorcycle. These query motorcycles are chosen to have diverse characteristics like pose, size, orientation, lane location etc.. For fairness, we ensure disjoint driving sessions of the dataset and the query images. We use a one-stage object detector based on a UNet-backbone that was pre-trained on 900k labeled images to detect the classes: 'car', 'truck', 'pedestrian', 'bicycle' and 'motorcycle'. In our method, we represent an image by extracting features $F \in \mathbb{R}^{w \times h \times c}$ from the penultimate layer of this pre-trained object detector DNN. To reduce storage costs, we downsize the embedding by adding a maxpool layer. For comparison with the baseline approach \[3\], we represent an image by flattening $F$ using global average pooling such that $F \in \mathbb{R}^c$. In the following sections, we evaluate the performance of DTM on various settings.

4.2 DTM Evaluation on Labeled Data

In this set of experiments, we evaluate our DTM approach and compare it with the methodology proposed in \[3\] (called baseline in Fig. 5). The task is to find semantically similar motorcycles in a labeled dataset with 465k images. This dataset has 10.28% images with at least one motorcycle, which we use as an upper bound for random selection. We evaluate both the methods using Top-N recall where we count a mined image as a true positive if it has at least one motorcycle.

Figure 5 shows the recall scores on Top-3500 mined images. For fine-grained analysis, we divide the query set into 5 bins with respect to the area of the ROI in the query image. DTM (gray) outperforms baseline (yellow) and random by a significant margin. We can see that DTM has good recall across the entire range of the query ROI area. Whereas the recall for the baseline approach degrades significantly for $\leq 5\%$ ROI area (small objects, e.g., ROI in top row query image in Fig. 6). This is due to the fact that DTM does a patchwise similarity only using the ROI area from the query image. Figure 6 demonstrates the qualitative results of DTM and the baseline. Our method
Figure 6: Results on our AV data: (a) Query image with region of interest (motorcycle). (b-c) DTM retrieved images containing motorcycles with similar semantics. (d-e) Baseline [3] retrieved images that use semantics of the whole query image. Note how accurate the DTM heap maps are even for small objects with ≤ 1% ROI area, e.g., the query image ROI in the top row.

produces sharp and localized heatmaps denoting its ability to accurately mine semantically similar objects. Since the baseline method uses a globally averaged flat embedding it cannot focus on the ROI. Hence, it retrieves images that represent the dominant objects/scenes in the query (e.g., urban scenes in the fourth row query, truck in the third row query).

4.3 DTM Evaluation on Unlabeled Data

We use the same experimental setup described in section 4.1 and 4.2 for evaluating DTM on a large unlabeled dataset with 2.2M images. We further shortlist a pool of 239k night-time images from the 2.2M images using metadata that indicates high likelihood for presence of at least one pedestrian or bicycle or motorcycle object in the image.

We first evaluate DTM and baseline on the 239k unlabeled dataset, and next DTM on the 2.2M unlabeled dataset. For evaluation, we compute recall by visually inspecting the top 100 images mined for each of the 36 queries using the DTM and baseline method. A mined image is counted as a true positive if there exists at least one motorcycle in it. We observe consistent results and similar conclusions for unlabeled data as in section 4.1 (see Fig. 7).

4.4 Hard In-Distribution Queries

In this experiment, we evaluate the performance of DTM in mining for hard queries. These hard queries typically have objects with unusual pose, size, occlusion or position in the image and tend to be under-represented or absent in the training dataset. Figure 8 shows an example of hard queries where a bicycle is mounted on the top/back/front of a vehicle. These variations of mounted bicycles were under-represented in the training dataset and were not detected by our object detector (false negatives).
Figure 9: Comparison of DI (a) with DTM (b-d) for retrieving semantically similar motorcy-
ecles/bicycles. The DI method retrieves images independent of the query image. Notice DTM
retrieving very similar images for hard failure cases like (b) side pose motorcycle (c) motorcycle
covered in cloth (d) bicycle mounted on the back of a car.

For these hard queries, we compare the performance of
DTM with an image retrieval strategy using the inference
of our object detector model. For ranking the images using
the object detector model inference, we score each image
as max of confidence values across all detected motorcycle
objects in that image. The score is 0 if no motorcycle is
detected. We refer to this as the detection inference (DI)
method. We use max so that the top retrievals using DI
contain a motorcycle with high probability. Note that the
features used in DTM are derived from the same trained
model used in DI.

Figure 9 shows the visual comparison between DTM and
DI. The DI method fails to mine for the semantics of the
query ROI beyond the object category (see row (a) in
Fig. 9). This is undesirable for our task where images
semantically similar to the failure case need to be mined.
To validate this, we use DTM to retrieve images similar
to a motorcycle/bicycle that the object detection model was not able to detect (false negative). We
consider 19 motorcycle/bicycle hard query images and retrieve 20 images for each query. 51% of
these retrieved images turned out to be false negatives when inference was run on them, indicating
that the retrieved images were also hard examples for the object detector model. Rows (b-d) in Fig. 9
show the top retrievals where we indeed see that DTM retrieves images with objects semantically
similar to the failure case. In (c), DTM retrieves motorcycles covered in cloth and in (d), bicycles
mounted on the back of a car.

4.5 Out-of-Distribution Queries

We test our DTM algorithm for retrieving images that are semantically similar to objects that the
underlying DNN was *not* trained for. We observe that DTM performs well when the query is spatially
co-located with objects that the object detector was trained for. For instance, in Fig. 10(a-b), the
query image has a stroller and luggage, both being dragged by a pedestrian. The object detector
was trained on pedestrian but not on stroller or luggage. For totally unrelated objects, DTM fails to
retrieve images with the object(s) of interest (see Fig. 10(c-d) where the query is a stop sign and
a traffic cone). However this is expected since the object detector from which the features were
extracted was not trained on these objects (stop sign and traffic cone). Many possible directions can
Figure 10: DTM retrieved images for out-of-distribution queries. (a) Stroller (b) Luggage (c) Stop sign (d) Traffic cone. In (a-b), DTM is able to retrieve semantically similar images from the same scene as the query as well as other scenes. For (c-d), DTM only retrieves images from the same scene since features for those objects are not modeled in the underlying DNN feature representation.

Figure 11: DTM retrieved images with multiple objects belonging to different semantic categories (motorcycle and bicycle) co-occurring in the same sample image.

be tried, like extracting the low-level features from an earlier layer of the DNN but we leave this for future research.

4.6 DTM for Multiple Regions of Interest

DTM can be easily extended to accommodate queries with multiple regions of interest at no extra cost. This can be done by projecting all the ROIs in the query image onto the query feature map (as described in Fig. 2). We normalize the score maps using the area of each object so that the retrieved images are not biased towards larger objects (see Equ. 3). For our AV use case, we evaluate the performance of DTM for retrieving semantically similar images with motorcycle and bicycle. We achieve an average Top-100 recall score of 88.5% across 4 queries, each with a pair of motorcycle and bicycle as ROIs. Figure 11 illustrates the qualitative performance of DTM for this task.
5 Conclusion

We propose a novel approach for object level targeted selection by using deep template matching. We study this problem in the context of mining images that are semantically similar to failure cases, like false negatives/positives of object detectors deployed in autonomous vehicles. These failure cases typically have unusual characteristics in terms of scale, pose, occlusion and tend to be absent or under-represented in the training dataset. Fixing these failure cases often involve mining for semantically similar objects from a large pool of unlabeled data. Our method focuses on the semantics of the objects of interest by projecting it onto the feature space and has high recall even when the object is small-sized, amid occlusion and heavy clutter. Our method works for multiple co-occurring objects in one or more semantic categories for object-level retrieval. Unlike other methods, it does not require extra labeled training data.

References


