Visual instance search from one example

Tao, R.

Publication date
2017
Document Version
Final published version
License
Other

A visual instance is a visually unique entity or a set of entities that have identical visual appearance and hence are not visually distinguishable.

This thesis is dedicated to visual instance search from one example. Given an image of a visual instance as query, the goal is to retrieve all the images of the same instance from a large image collection.

The thesis starts with developing methods for particular types of visual instances, continues with designing generic algorithms for a much broader set of instances, and ends on connecting visual instance search and visual object tracking.

ISBN: 978-94-6182-749-4

Visual Instance Search from One Example

Ran Tao

Visual Instance Search from One Example

Ran Tao

Visual Instance Search from One Example

Ran Tao

This book was typeset by the author using LaTeX 2_ε .

Printing: Off Page, Amsterdam

ISBN 978-94-6182-749-4

Visual Instance Search from One Example

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. K. I. J. Maex
ten overstaan van een door het college voor promoties
ingestelde commissie,
in het openbaar te verdedigen in de Agnietenkapel
op dinsdag 10 januari 2017, te 14:00 uur

door

Ran Tao

geboren te Jiangsu, China

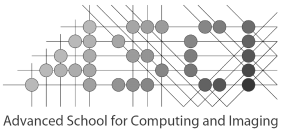
Promotiecommissie

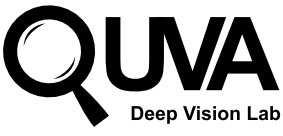
Promotor:	Prof. dr. ir. A. W. M. Smeulders	University of Amsterdam
Co-promotor:	Prof. dr. T. Gevers	University of Amsterdam
Overige leden:	Prof. dr. R. Cucchiara	University of Modena and Reggio Emilia
	Prof. dr. M. Welling	University of Amsterdam
	Prof. dr. ir. F. C. A. Groen	University of Amsterdam
	Dr. C. G. M. Snoek	University of Amsterdam
	Dr. E. Gavves	University of Amsterdam

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

UNIVERSITEIT VAN AMSTERDAM

This work described in this thesis has been carried out within the graduate school ASCI, dissertation number 359, at the lab of Intelligent Sensory Information Systems, and partially at QUVA Lab, University of Amsterdam. This research is partially supported by the Dutch national program COMMIT/.





COMMIT/

CONTENTS

1

INTRODUCTION

According to the Oxford dictionary, *instance*, when used as a noun, means *an example or a single occurrence of something*. This is a sharp definition as it reflects two essences of an instance, namely *generality* and *specificity*. "An example of something" emphasizes the generality. Instances or examples are often used to describe the kind, and they do so much better as instances are more concrete than abstract references. "A single occurrence of something", on the other hand, emphasizes the specificity of an instance, close to the original meaning of the word[1].

In Chinese, there is no such a single word that has the exact meaning of the English word instance. Rather, in Chinese, there are two separate words. One is 实例 which refers to *an example of something*, and the other one is 个例 which refers to *a single occurrence of something*. Other than the division over two words, the two meanings of *instance* are the same in both languages.

The generality of an instance is the property inherited from the kind of which it is an example. Hence, in one aspect, the generality of an instance does not exist without being a member of a kind. And, the generality of an instance varies when the kind under consideration changes[2]. Since the generality is a group property, all instances will inherit group identification aspects from the group. Hence, one can predict the generality of an instance without seeing it by transferring aspects from other instances of the same kind. For example, without seeing your friend's newly bought car, you can already predict confidently that it has wheels, doors and alike.

Specificity is the exclusive property of an instance. One cannot predict the specificity of an instance without seeing it. You cannot tell the color of your friend's newly bought car and the size of the doors without seeing it.

A way of approaching the specificity of an instance could be to derive from adding a modifier to the generality. For example, having door is the generality of an instance of car. Adding a modifier, *e.g.*, having the name of the owner engraved on the door, results in the specificity of the instance. In the sequel, we will explore this property.

Likewise, specificity can be divided into *relative specificity* and *absolute specificity*. *Relative specificity* is what makes the instance distinct from other instances of the same kind. *Relative specificity*, same as *generality*, varies when the kind under consideration changes. *Absolute specificity*, on the other hand, is what makes the instance distinct from anything else in the world, regardless of the kind.

This point of view suggests two tactics to explain what makes an instance unique. One tactic is to first describe what makes the instance being an example of a kind, *i.e.*,

1 According to the Oxford dictionary, in the late 16th century, the word *instance* denoted a particular case used to disprove a general assertion.

2 An entity can be an instance of multiple kinds.

Figure 1: Following the two-step procedure to explain the uniqueness of the instance in the left picture (the car), one first explains what makes the instance distinct from instances which are not cars, namely the instance is 2-3 meters wide, with four wheels, windshield, head lamps, radiator grille, able to carry a couple of people, etc. *Then one explains what makes the instance different from other instances of car, namely it has an exotic camouflage-like pattern, black wheels with ten spokes, four round lamps in the front,* etc. *As humans share a good understanding of car, we simplify the first step by just mentioning that the instance is a car. Employing the second tactic which directly describes what makes the instance distinct from anything else, one needs to describe the identifying aspects as many as possible to ensure that the description does not apply to any other things such as the instance in the right picture.*

its generality, and then to describe what makes it different from other instances of the same kind, *i.e.*, its relative specificity. As an instance can be an example of multiple kinds, there exist multiple combinations of generality and relative specificity to explain the uniqueness of an instance. This tactic will be referred to as the two-step identification procedure later. The other tactic is to directly describe what makes the instance distinct from anything else, *i.e.*, its absolute specificity. This may also be considered as the extreme combination of generality and relative specificity, where the kind is the one that contains everything in the world. Regarding the second tactic, often one cannot be certain whether everything has been taken into account, *i.e.*, one cannot be absolutely sure whether the description of the instance indeed does not apply to any other instance in the world. As humans we almost always use the first tactic, since we share a good understanding of common kinds, like *car*, and hence can skip the step of describing the generality by simply mentioning the instance is an example of that kind. See Figure 1 an example of explaining the uniqueness of an instance of car using the two tactics.

The question can be raised: *can an instance be a kind?* Following the meaning "an example of something", it seems plausible to say *human is an instance of mammal*. However, the meaning "a single occurrence of something" implies an instance is an individual. Following this meaning, it seems not proper to say *human is an instance of mammal* as human is not an individual but a kind. It is perhaps debatable whether an instance can be a kind. Here we divide instances into *primary instance*, *i.e.*, individuals, and *secondary instances*, *i.e.*, kinds, as Aristotle distinguished *primary substance* from *secondary substance*. And we focus on *primary instances* in this thesis.

And, *can an instance be abstract?* For example, is *an embarrassing moment at Arnold's office at 5:27 pm, 26 May, 2016* an instance of *embarrassing moment*? It is

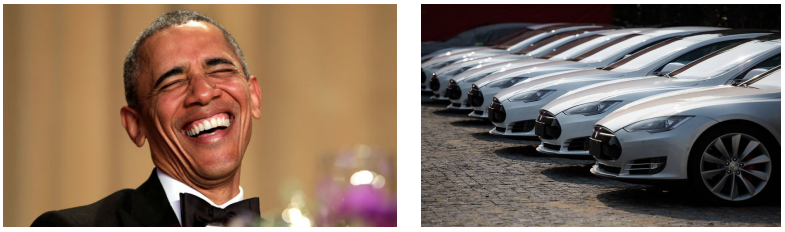


Figure 2: Barack Obama has unique visual characteristics which make him distinct from anything else, whereas a set of cars may have identical visual appearance and hence are not visually distinguishable.

perhaps also debatable. Within this thesis, as we study instances that can be perceived through visual sense, we only consider physical instances.

We departed from two essences of an instance, *generality* and *specificity*. Thus far, the definition of instance was discussed as an abstract notion. In reality, both the generality and the specificity of an instance can take different forms: visual, acoustic, tactile and others. As this thesis is about machine vision, only visual properties are considered. With this consideration, a more precise definition of *visual instance* is needed.

Certain entities in the world are visually unique, such as the Brooklyn Bridge and Barack Obama (Figure 2: left). They have unique visual characteristics based on which humans can distinguish them from anything else in the world. Some are not visually unique, like pencils, clothes and cars (Figure 2: right). Manufacturers often produce thousands of or even more visually identical copies for one model. For this type of entities, other non-visual information is required to uniquely identify individuals. You know the TV set in your living room is yours because it is right there in your living room. When it is placed in a street, you cannot tell whether or not it is your TV. Having that discussed, we give the definition of visual instance as follows. A *visual instance* is a visually unique entity or a set of entities that have identical visual appearance and hence are not visually distinguishable. In other words, two things are deemed different visual instances if and only if they can be differentiated solely based on visual signals.

Any visual instance has a finite spatial extent. Some instances are small, like an instance of *ant* or an instance of *button*. Some instances are big, like an instance of *dam* or an instance of *mountain*. Recreated in pictures, different instances cover image regions of different portions of the images. Of course, the spatial extent of an instance in images does not only depend on its spatial extent in the physical world, but also depends on the camera settings and the intent of the photographer. However, what is generally true is that, due to the finite spatial extent and common aesthetic sense, an instance, in images, often covers a part of the image instead of the whole image, like an instance of *car*. Exceptions are instances of *landscape* and instances with interior space where the camera can be positioned such as instances of *room*. See Figure 3.

A visual instance can have many and perhaps infinite number of pictorial instantiations. Photos of the same instance may look very different. Such variations come from two sources. One is that the instance can have appearance changes. Barack Obama surely will wear different clothes at different occasions. The same dog can be running, or being curled on a cushion. The other source is that the imaging conditions can be

Figure 3: On the left, an instance of landscape covers the entire image. On the right, an instance of car covers a portion of the image. The part of the image covered by the car has the identifying information of the instance, while the rest of the image is in general uninformative unless this car often comes to this place at sunset.

different. The Brooklyn Bridge can be recorded in rainy days or sunny days. It can be shot from a helicopter or by a person standing on the bridge. From one point of view, a particular picture of an instance is considered to be an instance of the instance. In this thesis, we do not consider a picture as an instance. And *visual instance search* is the task of retrieving all images of a target visual instance specified by a query, regardless the appearance variations in different recordings.

As many other search problems, in visual instance search, a query can be given in different forms. A query can be a textual description, *e.g.*, 'Brooklyn Bridge', known as query-by-text. Query-by-text allows one to search images from nothing. However, with a textual query, the images in the collection to be searched through need to have textual labels obtained manually or automatically [91], or the textual description needs to be transformed to some meta-representation, which allows for straightforward comparison with the image data [21]. An obvious limitation of query-by-text is that many visual instances cannot be specified precisely in textual descriptions. Alternatively, a query can also be specified by providing example images in what is known as query-by-example. Giving examples is equivalent to telling the machine that 'I want to search this visual instance'. The images in the collection do not need to be labeled. Query-by-example allows one to search any visual instances, including those that cannot be precisely described verbally. In this thesis, we focus on visual instance search from example images. In particular, we consider the extreme case where only 1 example is available.

Visual instance search has strong connections with several other fields in computer vision. It is important to 3D reconstruction from 2D images. Reconstructing a 3D object requires images of the object captured from different angles to have a good cover of the object. A powerful instance search algorithm can help find a diverse set of images of the target to facilitate the reconstruction [145]. Tasks like video description, generating a story automatically for a video, can also benefit from a good instance recognition and search algorithm, as often what is interesting in a video is something happening to a particular instance like this person and that car. Tracking may be considered as an instance search problem where the search set is composed of a set of images ordered by time. We will return to tracking later.

There are also many practical applications that motivate visual instance search. In the search for a suspect, footage from surveillance cameras in streets can be used to find the suspect. The same scenario can be generalized to sending a fleet of drones to locate one suspect. The core technology here is visual instance search. Or, imagine you are

Figure 4: The left and middle images in each row depict the same instance while the right image shows a different instance. Images of the same instance can look very different while images of different instances can sometimes look very similar.

visiting a museum and you are very interested in one piece of art. You can simply take a photo of the art and the instance search algorithm can help find it automatically on the Internet with all affiliated information.

As humans we can instantaneously recognize visual instances with almost perfect recognition accuracy. We are amazingly good at searching visual instances. For machines, however, it is a challenging task. Although compared to humans, machines have the advantage of being capable of efficiently searching through millions of images, so far the searching accuracy of an automatic system has been nowhere near human performance. On the one hand, the same instance can vary tremendously in appearance in different recordings due to scale change, rotation, illumination variation, viewpoint change, occlusion, self-deformation and other factors. As a consequence, the visual appearance lacks the visual invariant characteristics ascribed verbally to typical examples. On the other hand, although a visual instance has its unique visual characteristics, different instances, especially those belonging to the same or nearby classes of objects, share similar aspects of appearances and therefore cause the hardness of distinction. See Figure 4.

The powerful human vision is at least partially the consequence of a lifelong process of seeing and learning. In image categorization, the current best algorithms [60, 150, 153] can perform as well as humans under clear circumstances via learning from hundreds of or even more examples per category. However, the fact that one wants to search for images of a visual instance implies that the requestor does not have many images of that instance. In the query-by-example instance search problem, the number of example images that the machine can learn from is usually very limited. It is an extremely challenging case when there is only 1 example available as we consider here. One example can only show one side of an instance while the instance can have several sides. See Figure 5 an example.

As the main question for this thesis, given 1 image of a visual instance, how to find all the examples of the instance automatically from a collection despite all appearance variations it may have and despite the confusion with other similar instances? The main question generates special cases. A relatively easy case is where there are only one-sided views of an instance. In such case, matching the query image and the target suffices. This is still a formidable problem considering that there is no clue in what image and where

Figure 5: One image can only show one side of an instance while the instance can have several sides. It is a very challenging case when there is only 1 query example available.

in the image the target instance would appear. See Figure 6a. A particularly hard case is a query specified in frontal view while the relevant images in the search set show a view from the back which has never been seen before. See Figure 6b.

Since the introduction of the bag of visual words (BoW) formulation in 2003 [151], BoW and its improved variants have become the most popular paradigm to address instance search from one example [8, 73, 76, 133, 136, 158]. Approaches belonging to this paradigm match the appearance of local image patches in the potential image to the query image. In other words, this paradigm relies on gathering in the potential image local evidence of the presence of the target instance. Existing approaches search for the evidence over the entire image, ignoring an important fact that the target instance often occupies a (small) portion of the image. When the entire image is considered, the supportive evidence might drown in the sea of disturbing information from the background. With this in mind, we pose our first research question:

Can we exploit locality for better instance search accuracy?

This research question is addressed in Chapter 2. Instead of searching globally over the entire image, we propose to search locally in the image by evaluating many bounding boxes holding candidates for the target instance. An efficient storage and evaluation mechanism is proposed to efficiently evaluate hundreds of or even thousands of boxes per image. Furthermore, in Chapter 2, we also bring locality in the feature space, by efficiently employing a large visual vocabulary and an exponential similarity metric, to better measure the local evidence. This line of approaches resembles the tactic of directly identifying the 'absolute specificity', where the risk, as discussed, is that it is hard to ensure what is thought to be the absolute specificity of an instance is indeed so. Therefore, we introduce locality in the feature space to impose a strict matching criterion so that only very precise matches of local patches between the target image and the query count, as a way to reduce confusion from other, similar instances.

A type of instances that has received much attention are logos and other iconic visual symbols [80, 139–141]. Companies, organizations and even individuals use logos to promote public recognition. An accurate logo search system is useful as it can help measure the exposure by for example searching through the images uploaded to the social websites. Chapter 3 puts an emphasis on logos. When restricting the search to certain types of instances, as humans we often use specific domain knowledge. For example, when searching for a specific bird, we would focus on the beak, belly and tail

(a)

(b)

Figure 6: (a) Two images of the postnl logo. Logo is an example of 2D objects with only one-sided views. Instance search of 2D objects is relatively easy since the viewpoint difference is often limited as a consequence of being one-sided. However, this is still a formidable problem as there is no clue in what image and where in the image the target instance would appear. (b) Three images of an instance of shoe. Shoe is an example of 3D objects with views from multiple sides. A hard case is that the left image showing the frontal view of the instance is given as the query and the goal is to find the middle image showing the back view of the shoe.

as we know it is the unique detail on these local parts that makes a specific bird unique in appearance [45, 181, 190]. We pose the research question:

Can we exploit domain knowledge for better search accuracy on logos?

But what is the domain knowledge for logos? Logos are a special type of instances. Text is often a part of a logo. Companies and organizations usually put their names in the logo for better public recognition. In Chapter 3, we exploit the recognized text in the image to improve logo search.

In the first two chapters, we focus on particular types of visual instances, mainly buildings and logos. In the next chapters, we pursue instance search on a much broader set of visual instances. The ultimate goal is arbitrary instance search where any visual instance is searchable. We phrase the research question

Can we design a generic method capable of searching for an arbitrary visual instance?

This research question is initially addressed in Chapter 4. Here we first investigate how the state-of-the-art methods perform on generic instance search from 1 example where the query instance can be an arbitrary object. Can we search for other objects like shoes using the same method that has been shown promising for buildings? To that end, we evaluate several existing instance search algorithms [76, 77, 120, 138, 156] on both buildings and shoes, two very different types of objects. The conclusion is that none of the existing methods work well on both buildings and shoes. Interestingly, the method proposed in Chapter 2 achieves the best performance on buildings, but loses its generality on shoes, performing worse than all other methods. And a method that works best on shoes performs worst on buildings.

Why is it so difficult to perform well on both buildings and shoes? The root is the different characteristics of buildings and shoes. Buildings, especially the famous ones, like those in the Oxford dataset [133], usually have one main side where people often take photos. Therefore, buildings are approximately 2D and one-sided objects. The consequence of being one-sided is that the viewpoint variations of these instances in the images are limited. And, instances like building often have rich textures. The limited viewpoint variations and the rich visual details render methods which rely on matching unique local details suited, as the local details can be reliable matched across different images under limited viewpoint variations [109]. To the contrary, objects like shoes are real 3D objects, and when photographed from every possible viewing angles, have large viewpoint variations. Moreover, shoes usually do not have rich textured patterns. These properties make methods based on matching local details inferior. Rather, certain methods that capture general information how such objects in general look from all sides are desired.

A generic method for instance search has to be able to extract different levels of information when dealing with different types of instances. When searching for instances like buildings, the extracted information has to capture the specifically identifying details for that building. When searching for instances like shoes, however, the extracted

information needs to capture the general view of the shoe informative from all sides, as it is yet unknown which of the views will be present in the dataset. In Chapter 4, we present a generic, data-driven, method, aiming to handle various types of instances. The proposed method learns, from a set of instances of the same category, a group of visual aspects. These visual aspects are learned to be invariant to occasional recording factors like viewpoint change. And, these aspects are generalizable to new, previously unseen, instances of the same category. The aspects are a useful basis to derive the relative specificity of an instance in the two-step identification procedure to make within-category distinction. In fact, these aspects are category-specific attributes. For example, in the case of shoe search where shoe is the category, the attributes roughly coincide with what humans would call *high-heel*, *boot* and *openness*, to name a few. Given the aspects as learned automatically, the specificity of an instance is derived by precisely quantifying the visual aspects of the example image, *e.g.*, *heel of this height* and *openness to this extent*. This is a direct consequence of the discussion above that specificity of an instance can be seen as a modifier to general aspects.

In the fifth chapter of the thesis, we make a connection between generic instance search from 1 example and visual object tracking. In tracking, the goal is to follow an instance throughout the video by predicting its locations in frames, starting from one observation of the target instance, usually provided in the initial frame of the sequence. The main challenge is to cope with the appearance variations the target may undergo over time due to scale changes, in and out-of-plane rotation, camera motion, uneven illumination, deformation, occlusion and other factors. As mentioned earlier, one way of viewing tracking is to think tracking as an instance search problem where the dataset to search through contains a set of images ordered by time. The temporal coherence in tracking videos has motivated many tracking algorithms with a focus on motion [19, 66] and sequential modeling [57, 61, 189]. As a result, the connection between visual instance search and tracking has been obfuscated. Tracking and instance search have been two independent research topics for a long time without interaction. This brings us to the next research question of the thesis.

Can we address tracking as an instance search problem (over the video at hand)? That is, can we handle tracking without taking the temporal coherence into account?

The question is addressed in Chapter 5. From the standpoint of the conventional standard tracking literature the proposed tracker is simple: it tracks the target instance simply by retrieving in each incoming frame the patch that is most similar in appearance to the initial patch of the target. This simple way of tracking is similar to the simplicity of the normalized cross-correlation (NCC) tracker which was proposed 40 years ago [17, 34]. However, from the point of view of answering what is really relevant to detect the same instance in several different images, the proposed algorithm is highly sophisticated and flexible, as it externally learns all the possible visual variations of any object, even if this object has never been seen before by the tracker. The proposed tracker only has an instance search core with a powerful similarity metric which is learned in an end-to-end manner using a Siamese deep convolutional neural network [18, 25]. The tracker does not apply on-the-fly sequential learning [57, 61, 189], occlusion detection [66, 127, 129],

combination of trackers [65, 173], geometric matching [65, 129] and alike.

This thesis is dedicated to visual instance search from 1 example. The thesis starts with developing methods for particular types of visual instances, continues with designing generic algorithms for a much broader set of instances, and ends on connecting visual instance search and tracking. Findings of this thesis may lead to a better understanding of what makes an instance *an example of something* and *a single occurrence of something* in the visual scope.

1.1 MATERIALS FOR THE REMAINING CHAPTERS

- **Chapter 2** is based on "Locality in Generic Instance Search from One Example", published in IEEE Conference on Computer Vision and Pattern Recognition, 2014, by Ran Tao, Efstratios Gavves, Cees Snoek and Arnold Smeulders [156].

  Contribution of authors
  Ran Tao: all aspects
  Efstratios Gavves: helped with designing the method
  Cees Snoek: supervised the research
  Arnold Smeulders: supervised the research

- **Chapter 3** is based on "Words Matter: Scene Text for Image Classification and Retrieval", under review for publication in IEEE Transactions on Multimedia, by Sezer karaoglu, Ran Tao, Theo Gevers and Arnold Smeulders [83].

  Contribution of authors
  Sezer Karaoglu and Ran Tao equally contributed to this work. Theories and algorithms were developed together. Sezer Karaoglu focused on implementing textual cue extraction, whereas Ran Tao focused on implementing visual cue extraction for fine-grained classification and logo retrieval tasks. The experiments, the analysis and paper writing were performed by Sezer Karaoglu and Ran Tao.
  Theo Gevers and Arnold Smeulders supervised the research.

- **Chapter 4** is based on "Attributes and Categories for Generic Instance Search from One Example", published in IEEE Conference on Computer Vision and Pattern Recognition, 2015, by Ran Tao, Arnold Smeulders and Shih-Fu Chang [157].

  Contribution of authors
  Ran Tao: all aspects
  Arnold Smeulders: supervised the research
  Shih-Fu Chang: supervised the research

- **Chapter 5** is based on "Siamese Instance Search for Tracking", published in IEEE Conference on Computer Vision and Pattern Recognition, 2016, by Ran Tao, Efstratios Gavves and Arnold Smeulders [155].

  Contribution of authors
  Ran Tao: all aspects
  Efstratios Gavves: supervised the research
  Arnold Smeulders: supervised the research

2

LOCALITY IN INSTANCE SEARCH FROM ONE EXAMPLE

2.1 INTRODUCTION

[1] In instance search the ideal is to retrieve all pictures of an object given a set of query images of that object [7, 73, 125, 135]. Similar to [8, 26, 133, 139, 160], we focus on instance search on the basis of only one example. Different from the references, we focus on *generic* instance search, like [9, 76, 131], in that the method will not be optimized for buildings, logos or another specific class of objects.

The challenge in instance search is to be invariant to appearance variations of the instance while ignoring other instances from the same type of object. With only one example, generic instance search will profit from finding relevant unique details, more than in object categorization, which searches for identifying features shared in the class of objects. The chances of finding relevant unique details will increase when their representation is invariant and the search space is reduced to local and promising areas. From this observation, we investigate ways to improve locality in instance search at two different levels: locality in the picture *and* locality in the feature space.

In the picture, we concentrate the search for relevant unique details to reasonable candidate localizations of the object. Spatial locality has been successfully applied in image categorization [59, 161]. It is likely to be even more successful in instance search considering that there is only one training example and the distinctions to the members of the negative class are smaller. The big challenge here is to keep the number of candidate boxes low while retaining the chance of having the appropriate box. The successful selective search [162] is still evaluating thousands of candidate boxes. Straightforward local picture search requires a demanding 1,000s-fold increase in memory to store the box features. We propose efficient storage and evaluation of boxes in generic instance search. We consider this as the most important contribution of this work.

In the feature space, local concentration of the search is achieved in two ways. The first tactic is using large visual vocabularies as they divide the feature space in small patches. In instance search, large vocabularies have been successfully applied in combination with Bag of Words (BoW), particularly to building search [110, 133, 134]. Without further optimizations to buildings [27, 133], BoW was shown inferior in performance in instance search to VLAD and Fisher vector [76]. Therefore, we focus on the latter two for generic instance search. Yet the use of large vocabularies with these methods is prohibited by the memory it requires. We propose the use of large vocabularies with these modern methods.

1 Published in *IEEE Conference on Computer Vision and Pattern Recognition, 2014* [156].

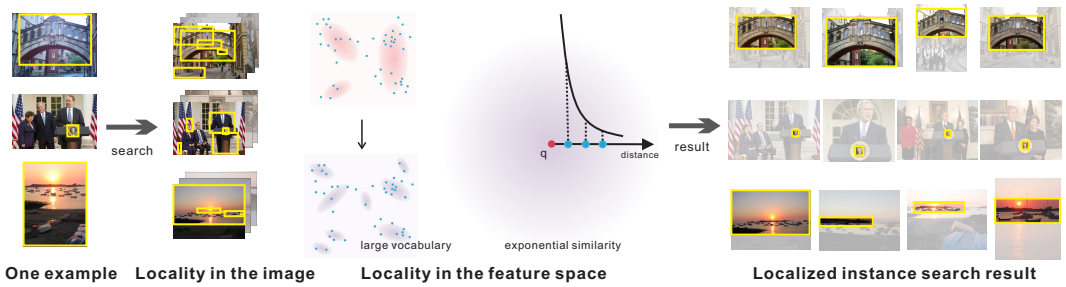


Figure 7: We propose locality in generic instance search from one example. As the first novelty, we consider many boxes as candidate targets to search locally in the picture by an efficient point-indexed representation. The same representation allows, as the second novelty, the application of very large vocabularies in Fisher vector and VLAD to search locally in the feature space. As the third novelty, we propose the exponential similarity to emphasize local matches in feature space. The method does not only improve the accuracy but also delivers a reliable localization.

As a second tactic in the feature space, we propose a new similarity function, named *exponential similarity*, measuring the relevance of two local descriptors. The exponential similarity enhances locality in the feature space in that the remote correspondences are punished much more than the closer ones. Hence this similarity function emphasizes local search in the feature space.

As the first novelty in this work, we aim for an efficient evaluation of many boxes holding candidates for the target by a point-indexed representation independent of their number. The representation allows, as the second novelty, the application of very large vocabularies in Fisher vector and VLAD in such a way that the memory use is independent of the vocabulary size. The large vocabulary enables the distinction of local details in the feature space. Thirdly, we propose the exponential similarity function which emphasizes local matches in the feature space. We summarize our novelties in Figure 7. We demonstrate a drastic increase in performance in generic instance search, enabled by an emphasis on locality in the feature space and the image.

2.2 RELATED WORK

Most of the literature on instance search, also known as object retrieval, focuses on a particular type of object. In [8, 133, 134] the search is focused on buildings, for which vocabularies of 1M visual words successfully identify tiny details of individual buildings. For the same purpose, building search, geometrical verification in [133], improves the precision further, and query expansion in [26, 27] with geometrically verified examples further improves recall. For the topic of logos specifically, in [139], a method is introduced by utilizing the correlation between incorrect keypoint matches to suppress false retrievals. We cover these hard problems on buildings and logos, but at the same time consider the retrieval of arbitrary scenes. To that end, we consider the three standard datasets, Oxford5k [133], BelgaLogos [80] and the Holidays dataset [73] holding 5,062, 10,000 and 1,491 samples each. We do the analysis to evaluate one and the same generic method. Besides, we define a new dataset, TRECVID50k, which is a 50,000 sample of the diverse TRECVID dataset [125] for generic instance search.

BoW quantizes local descriptors to closest words in a visual vocabulary and produces a histogram counting the occurrences of each visual word. VLAD [76] and Fisher vector [130] improve over the performance of BoW by difference encoding, subtracting the mean of the word or a Gaussian fit to all observations respectively. As VLAD and Fisher vector focus on differences in the feature space, their performance is expected to be better in instance search, especially when the dataset grows big. We take the recent application to instance search of VLAD [9, 76] and Fisher vector [76, 131] as our point of reference.

In [110,123,133], the feature space is quantized with a large BoW-vocabulary leading to a dramatic improvement in retrieval quality. In VLAD and Fisher vector, storing the local descriptors in a single feature vector has the advantage that the similarity between two examples can readily be compared with standard distance measures. However, such a one-vector-representation stands against the use of large vocabularies in these methods, as the feature dimensionality, and hence the memory footprint, grows linearly with the vocabulary size. Using a vocabulary with 20k visual clusters will produce a vector with 2.56M dimensions for VLAD [9]. In this study, we present a novel representation independent of the vocabulary size in memory usage, effectively enabling large vocabularies.

Spatial locality in the picture has shown a positive performance effect in image categorization [59, 161]. Recent work [5, 35, 162] focuses on generating candidate object locations under a low miss rate. Selective search [162] oversegments the image and hierarchically groups the segments with multiple complementary grouping criteria to generate object hypotheses, achieving a high recall with a reasonable number of boxes. We adopt selective search for instance search, but the method we propose will function for any other location selection method.

Spatial locality has been applied in retrieval [79, 88, 96]. [88] applies BoW on very, very many boxes inserted in a branch and bound algorithm to reduce the number of visits. We reduce their number from the start [162], and we adopt the superior VLAD and Fisher vector representations rather than BoW. [79] randomly splits the image into cells and applies BoW model. [96] proposes a greedy search method for a near-optimal box and uses the score of the box to re-rank the initial list generated based on global BoW histograms. The reference applies locality after the analysis, relying on the quality of the initial result. The method in the reference is specifically designed for BoW, while we present a generic approach which is applicable to VLAD, Fisher vector and BoW as well. The authors in [9] study the benefits of tiling an image with VLADs when searching for buildings which cover a small portion of an image. In the reference, an image is regularly split into a 3 by 3 grid, and 14 boxes are generated, 9 small ones, 4 medium ones (2 x 2 tiles), and the one covering the entire image. A VLAD descriptor is extracted from each of the boxes and evaluated individually. In this work, we investigate the effect of spatial locality using the candidate boxes created by the state-of-the-art approach in object localization rather than tiling, and evaluate on a much broader set of visual instances.

The exponential similarity function introduced in this work is similar to the thresholded polynomial similarity function recently proposed in [158] and the query adaptive similarity in [136] in that all pose higher weights on closer matches which are more likely

to be true correspondences. However, our proposal has fewer parameters than [158] and does not need the extra learning step of [136].

2.3 LOCALITY IN THE IMAGE

Given the query instance outlined by a bounding box, relevant details in a positive database image usually occupy only a small portion of the image. Analyzing the entire database image in the search is suboptimal as the real signal on the relevant region will drown in the noise from the rest. The chance of returning an image which contains the target instance is expected to be higher if the analysis is concentrated on the relevant part of the image only. To this end, we propose to search locally in the database image by evaluating many bounding boxes holding candidates for the target and ranking the images based on the per-image maximum scored box. Generating promising object locations has been intensively researched in the field of category-level object detection [5, 35, 162]. We adopt selective search [162] to sample the bounding boxes.

Evaluating many bounding boxes per database image, however, is practically infeasible in combination with VLAD or Fisher vector, since the VLAD or Fisher representations for all the boxes are either too expensive to store or too slow to compute on-the-fly. On the 5,062 images of the Oxford5k dataset [133], selective search will generate over 6 million boxes. With VLAD encoding this will generate over 700 gigabytes even with a small vocabulary consisting of 256 clusters. We therefore propose to decompose the one-vector representations into point-indexed representations, which removes the linear dependence of the memory requirement on the number of sampled boxes. Furthermore, we decompose the similarity function accordingly for efficient evaluation, saving on an expensive online re-composition of the one-vector representation. In the following we first briefly review VLAD and Fisher vector, and then describe the decomposition of the appearance models and the similarity measure, which allows to evaluate boxes efficiently in a memory compact manner.

2.3.1 *Global appearance models*

Let $\mathcal{P} = \{\mathbf{p}_t, t = 1...T\}$ be the set of interest points and $\mathcal{X} = \{\mathbf{x}_t, t = 1...T\}$ be the d-dimensional local descriptors quantized by a visual vocabulary $C = \{\mathbf{c}_i, i = 1...k\}$ to its closest visual word $q(\mathbf{x}) = argmin_{\mathbf{c}\in C}\|\mathbf{x} - \mathbf{c}\|^2$, where $\|.\|$ is the ℓ_2 norm.

Where BoW counts the occurrences of each visual word into a histogram $\mathbf{V}_B = [w_1, ...w_k]$ with $w_i = \sum_{\mathbf{x}_t\in\mathcal{X}:q(\mathbf{x}_t)=\mathbf{c}_i} 1$, VLAD sums the difference between the local descriptor and the visual word center, which results in a d-dimensional sub-vector per word $\mathbf{v}_i = \sum_{\mathbf{x}_t\in\mathcal{X}:q(\mathbf{x}_t)=\mathbf{c}_i}(\mathbf{x}_t - \mathbf{c}_i)$, concatenated into:$\mathbf{V}_V = [\mathbf{v}_1, ..., \mathbf{v}_k]$. VLAD quantifies differentiation within the visual words and provides a joint evaluation of several local descriptors.

Fisher vector models the local descriptor space by a Gaussian Mixture Model, with parameters $\lambda = \{\omega_i, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i, i = 1, ..., k\}$ where $\omega_i, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i$ are the mixture weight, mean vector and the standard deviation vector of the i^{th} component. Fisher vector describes how a set of local descriptors deviates from the universal distribution of the local descriptor space via taking the gradient of the set's log likelihood with respect

to the parameters of the GMM, first applied to image classification by Perronnin *et al.* [130,132]. Later the gradient with respect to the mean was applied to retrieval [76,131]: $\mathbf{g}_i = \frac{1}{\sqrt{\omega_i}} \sum_{t=1}^{T} \gamma_t(i) \frac{\mathbf{x}_t - \boldsymbol{\mu}_i}{\sigma_i}$ where $\gamma_t(i)$ is the assignment weight of $\mathbf{x}_t$ to Gaussian i. We drop T from the denominator as mentioned in [76], as it will be canceled out during normalization. The Fisher vector representation $\mathbf{V}_F$ is the concatenation of $\mathbf{g}_i$ for $i = 1...k$: $\mathbf{V}_F = [\mathbf{g}_1, ..., \mathbf{g}_k]$.

2.3.2 *Decomposition of appearance models*

Decomposing a VLAD vector into point-indexed features is straightforward. The description of an interest point $\mathbf{p}_t$ with local descriptor $\mathbf{x}_t$ in VLAD is simply represented by the index of the closest visual word plus the difference vector with the word center

$$\{q_{ind}(\mathbf{x}_t); \mathbf{d}_t = \mathbf{x}_t - q(\mathbf{x}_t)\}. \tag{2.1}$$

Before we can decompose Fisher vectors, we note that in the original implementation each local descriptor contributes to all k Gaussian components, which imposes a serious memory burden as each point will produce k different representations. We thereby modify the original formulation by allowing association with the largest assignment weights only. A similar idea has been explored for object detection in [24], where only the components with assignment weights larger than a certain threshold are considered. After rewriting the above equation for $\mathbf{g}_i$ into $\mathbf{g}_i = \sum_{\mathbf{x}_t \in \mathcal{X}: \gamma_t(i) \neq 0} \frac{\gamma_t(i)}{\sqrt{\omega_i}} \frac{\mathbf{x}_t - \boldsymbol{\mu}_i}{\sigma_i}$, the description of a point in the truncated Fisher vector, tFV, is given by the index r_t^j of the Gaussian component with j^{th} largest soft assignment weight, the assignment weight divided by the square root of the mixture weight and similar to the VLAD-case, the difference to the mean. Point $\mathbf{p}_t$ is represented by

$$\{[r_t^j; \frac{\gamma_t(r_t^j)}{\sqrt{\omega_{r_t^j}}}; \mathbf{d}_{tj} = \frac{\mathbf{x}_t - \boldsymbol{\mu}_{r_t^j}}{\boldsymbol{\sigma}_{r_t^j}}], j = 1...m\}. \tag{2.2}$$

Apparently, the memory consumption of the point-indexed representations is independent of the number of boxes. However, as in VLAD and tFV the difference vectors have the same high dimensionality as the local descriptors, the memory usage of the representations is as yet too large. Hence, we propose to quantize the continuous space of the difference vectors into a discrete set of prototypic elements and store the index of the closest prototype instead of the exact difference vector to arrive at an arbitrarily close approximation of the original representation in much less memory. As in [74], the difference vectors are split into pieces with equal length and each piece is quantized separately. We randomly sample a fixed set of prototypes from real data and use the same set to encode all pieces. Denote the quantization function by $\widetilde{q}$ and the index of the assigned prototype by $\widetilde{q_{ind}}$. Each difference vector $\mathbf{d}_t$ is represented by $[\widetilde{q_{ind}}(\mathbf{d}_{t_s}), s = 1...l]$, where $\mathbf{d}_{t_s}$ is the s^{th} piece of $\mathbf{d}_t$. The quantized point-indexed representations are memory compact, and box independent. To allow the evaluation of bounding boxes, we also store the meta information of the boxes, such as the coordinates, which costs a small extra amount of space.

2.3.3 *Decomposition of similarity measure*

Cosine similarity is the de facto similarity measure for VLAD [9, 76] and Fisher vector [76, 131], and hence for tFV. We propose to decompose accordingly the similarity measure into pointwise similarities, otherwise the one-vector-representation of a box has to be re-composed before being able to measure the similarity score of the box.

To explain, first consider the decomposition of the cosine similarity for BoW histograms. Let Q be the query box with $\mathcal{X}^Q = \{\mathbf{x}_1^Q, ..., \mathbf{x}_{n_Q}^Q\}$ local descriptors and let $\mathcal{X}^R = \{\mathbf{x}_1^R, ..., \mathbf{x}_{n_R}^R\}$ be the local descriptors of a test box R. The cosine similarity between histograms $\mathbf{V}_B^Q = [w_1^Q, ..., w_k^Q]$ and $\mathbf{V}_B^R = [w_1^R, ..., w_k^R]$ is:

$$S_B^{QR} = \frac{1}{\|\mathbf{V}_B^Q\|\|\mathbf{V}_B^R\|} \sum_{i=1}^{k} w_i^Q w_i^R. \tag{2.3}$$

For the sake of clarity, we will drop the normalization term $\frac{1}{\|\mathbf{V}_B^Q\|\|\mathbf{V}_B^R\|}$ in the following elaboration. By expanding w_i^Q, w_i^R with $\sum_{z=1}^{n_Q} q_{ind}(\mathbf{x}_z^Q) == i, \sum_{j=1}^{n_R} q_{ind}(\mathbf{x}_j^R) == i$ and reordering the summations the equation turns to

$$S_B^{QR} = \sum_{j=1}^{n_R} \sum_{z=1}^{n_Q} (q_{ind}(\mathbf{x}_j^R) == q_{ind}(\mathbf{x}_z^Q)) \cdot 1. \tag{2.4}$$

We define the term $(q_{ind}(\mathbf{x}_j^R) == q_{ind}(\mathbf{x}_z^Q)) \cdot 1$ in Equation 2.4 as the pointwise similarity between $\mathbf{x}_j^R$ and $\mathbf{x}_z^Q$. Denoting $(q_{ind}(\mathbf{x}_j^R) == q_{ind}(\mathbf{x}_z^Q))$ by δ_{jz} we derive the pointwise similarity for BoW as

$$\hat{S_B}(\mathbf{x}_j^R, \mathbf{x}_z^Q) = \delta_{jz} \cdot 1. \tag{2.5}$$

The VLAD-similarity S_V^{QR} can be decomposed in a similar way into a summation of pointwise similarities, defined as

$$\hat{S_V}(\mathbf{x}_j^R, \mathbf{x}_z^Q) = \delta_{jz} < \mathbf{d}_j^R, \mathbf{d}_z^Q >, \tag{2.6}$$

where $\mathbf{d}_j^R$ and $\mathbf{d}_z^Q$ are the differences with the corresponding visual word centers. Replacing the exact difference vectors with the quantized versions, we derive

$$\hat{S_V}(\mathbf{x}_j^R, \mathbf{x}_z^Q) = \delta_{jz} \sum_{i=1}^{l} < \widetilde{q}(\mathbf{d}_{j_i}^R), \widetilde{q}(\mathbf{d}_{z_i}^Q) > . \tag{2.7}$$

As the space of the difference vectors has been reduced to a set of prototypical elements, the pairwise dot products $D(i, j)$ between prototypes can be pre-computed. Inserting the pre-computed values, we end up with

$$\hat{S_V}(\mathbf{x}_j^R, \mathbf{x}_z^Q) = \delta_{jz} \sum_{i=1}^{l} D(\widetilde{q_{ind}}(\mathbf{d}_{j_i}^R), \widetilde{q_{ind}}(\mathbf{d}_{z_i}^Q)). \tag{2.8}$$

In the same manner, the pointwise similarity measure for tFV approximated up to the m^{th} Gaussian, can be derived as follows:

$$\hat{S_A}(\mathbf{x}_j^R, \mathbf{x}_z^Q) = \sum_{f,h=1}^{m} \psi_{jz}^{fh} < \mathbf{d}_{jf}^R, \mathbf{d}_{zh}^Q >, \tag{2.9}$$

where

$$\psi_{jz}^{fh} = (r_j^f == r_z^h) \frac{\gamma_j(r_j^f)\gamma_z(r_z^h)}{\sqrt{\omega_{r_j^f}}\sqrt{\omega_{r_z^h}}}. \tag{2.10}$$

Inserting the pre-computed values, we arrive at

$$\hat{S_A}(\mathbf{x}_j^R, \mathbf{x}_z^Q) = \sum_{f,h=1}^{m} \psi_{jz}^{fh} \sum_{i=1}^{l} D(\widetilde{q_{ind}}(\mathbf{d}_{jf_i}^R), \widetilde{q_{ind}}(\mathbf{d}_{zh_i}^Q)). \tag{2.11}$$

The evaluation of sampled bounding boxes is as follows. The approach computes the score of each interest point of the database image through the pointwise similarity measure described above, and obtains the score of a certain bounding box by summing the scores over the points which locate inside the box. Considering that the pointwise scores only need to be computed once and the box scores are acquired by simple summations, the proposed paradigm is well suited for evaluating a large number of boxes.

2.4 LOCALITY IN THE FEATURE SPACE

In this section we continue on localizing the search in the feature space with two different tactics.

2.4.1 *Large vocabularies*

We employ large vocabularies in order to shrink the footprint of each word to a local comparison of close observations. This will suppress the confusion from irrelevant observations as they are less likely to reside in the same small cells as the query descriptors. Moreover, small visual clusters can better capture the details in the local feature space, enabling distinction between very similar observations.

It is practically infeasible to apply very large vocabularies directly in the standard VLAD and Fisher vector as the dimensionality of VLAD and Fisher representation grows linearly with the size of the vocabulary. However, the point-indexed representation described in the previous section allows the application of very large vocabularies in VLAD and Fisher vector effortlessly. Its memory consumption is independent of the size of the vocabularies, as for each point it only requires storing m numbers for tFV (and 1 for VLAD) to indicate the associated visual clusters.

2.4.2 *Exponential similarity*

In instance search it is reasonable to reward two descriptors with a *disproportionally* high weight when they are close, as we seek exact unique details to match with the detail of the one query example. The pointwise similarities in equations 2.6 and 2.9 do not meet this property. We enhance locality in the feature space by *exponential similarity*.

Without loss of generality, we consider the VLAD case as an example to elaborate. The exponential pointwise similarity for VLAD coding is expressed as

$$S_V^{\hat{exp}}(\mathbf{x}_j^R, \mathbf{x}_z^Q) = \delta_{jz} \cdot exp(\beta \cdot f(\mathbf{d}_j^R, \mathbf{d}_z^Q)), \tag{2.12}$$

where $f(\mathbf{d}_j^R, \mathbf{d}_z^Q)$ measures the cosine similarity of the two difference vectors, and β is a parameter which controls the shape of the exponential curve.

The rate of the change is captured by the first-order derivate. The derivate of the above exponential similarity function with respect to the cosine similarity is

$$\frac{\partial S_V^{\hat{exp}}(\mathbf{x}_j^R, \mathbf{x}_z^Q)}{\partial f(\mathbf{d}_j^R, \mathbf{d}_z^Q)} = \delta_{jz} \cdot exp(\beta \cdot f(\mathbf{d}_j^R, \mathbf{d}_z^Q)) \cdot \beta. \tag{2.13}$$

Indeed, the rate of similarity change increases as the two observations get closer.

The proposed exponential similarity emphasizes locality in the feature space, putting disproportionally high weight on close matches.

2.5 EXPERIMENTS

2.5.1 *Experimental setup*

Datasets. We evaluate the proposed methods on 3 datasets, namely Oxford buildings [133], Inria BelgaLogos [80]and Inria Holidays [73]. Oxford buildings contains 5,062 images downloaded from Flickr. 55 queries of Oxford landmarks are specified, each by a query image and a bounding box. BelgaLogos is composed of 10,000 press photographs. 55 queries are defined, each by an image from the dataset and the logo's bounding box. Holidays consists of 1,491 personal holiday pictures, 500 of them used as queries. For all datasets, the retrieval performance is measured in terms of mean average precision (mAP).

Local descriptors. We use the Hessian-Affine detector [108, 128] to extract interest points on Oxford5k and BelgaLogos while the public available descriptors are used for Holidays. The SIFT descriptors are turned into RootSIFT [8], and the full 128D descriptor is used for VLAD as in [9], while for Fisher vector and tFV, the local descriptor is reduced to 64D by PCA, as [76, 144] have shown PCA reduction on the local descriptor is important for Fisher vector, and hence also for tFV.

Vocabularies. The vocabularies for Oxford buildings are trained on Paris buildings [134], and the vocabularies for Holidays are learned from Flickr60k [73], the same as in [9]. For BelgaLogos the vocabularies are trained on a random subset of the dataset.

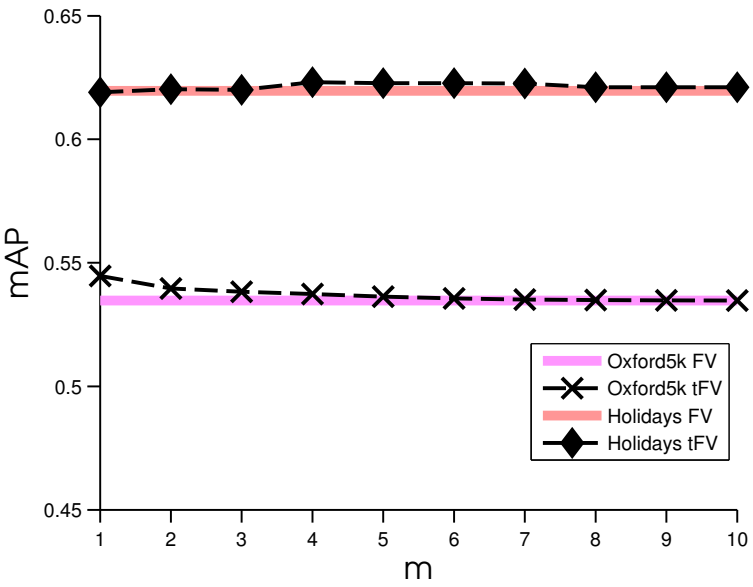


Figure 8: ***Impact of the parameter*** m ***on the performance of tFV.*** *The parameter m controls the number of Gaussian components each point is assigned to. The straight line is for $m = 256$, the standard Fisher vector implementation. It is clear that the first assignment is by far the most important one.*

2.5.2 *Truncated Fisher vector*

We first evaluate the performance of tFV with different values of m, which controls the number of Gaussian components each SIFT descriptor is associated with. We compare tFV with the original Fisher vector under the same setting, where a GMM with 256 components is learned to model the feature space and the full database image is used during the search.

As shown in Figure 8, m has little impact on the result. tFV and the original Fisher vector have close performance. In the following experiments, we set $m = 2$ for tFV.

2.5.3 *Spatial locality in the image*

In this experiment we test whether adding spatial locality by analyzing multiple bounding boxes in a test image improves the retrieval performance, as compared to the standard global retrieval paradigm where only the full image is evaluated. For the localized search, we use the highest scored box as the representative of the image to rank the test examples. We use the same vocabulary with 256 visual clusters for both global retrieval and localized retrieval. In order to ensure a fair comparison and show the influence of spatial locality, we apply ℓ_2 normalization in all cases. The results are shown in Table 1.

Localized search has a significant advantage on Oxford5k (landmarks) and BelgaLogos (small logos), in short for fixed shape things, while on the scene-oriented Holidays dataset, global search works slightly better.

When searching for an object which occupies part of the image, see Figure 9, introducing spatial locality is beneficial, as the signal to noise ratio within the bounding box is much higher than the entire image, especially for small non-conspicuous objects.

	VLAD		tFV	
	global [76]	*local*	*global* [76]	*local*
Oxford5k	0.505	0.576	0.540	**0.591**
BelgaLogos	0.107	0.205	0.120	**0.219**
Holidays	0.596	0.597	**0.620**	0.610
Generic	0.403	0.460	0.427	**0.473**

Table 1: ***The influence of spatial locality.*** *Localized search evaluates multiple locations in a database image and takes the highest scored box as the representative, while global search [76] evaluates the entire image. To ensure a fair comparison and show the influence of spatial locality, we use the same vocabularies with 256 clusters and ℓ_2 normalization for both localized search and global search. Localized search is advantageous on object-oriented datasets, namely Oxford5k and BelgaLogos, while on scene-oriented Holidays, global search works slightly better. As the average mAP over the three datasets in the last row shows, the proposed localized search is generic, working well on a broad set of instances.*

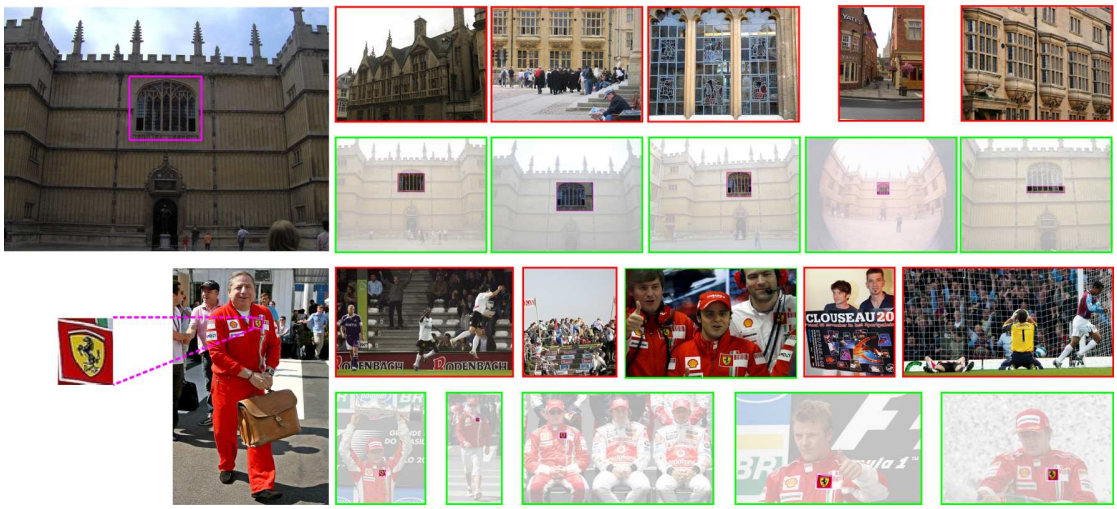


Figure 9: ***The effect of spatial locality.*** *Query instances are shown on the left, delineated by the bounding box. On the right are the top 5 retrieved examples. For each query example, the upper row and lower row are results returned by global search and localized search respectively. Positive (negative) samples are marked with green (red) borders. Focusing on local relevant information, localized search has successfully ranked and discovered the instance despite the presence of a noisy background.*

However, when looking for a specific scene which stretches over the whole picture, adding spatial locality cannot profit. As whether it is an edifice, a logo, an object or alternatively a scene is a property of the query, it can be specified with a simple question at query-time whether to use locality or globality in the search.

	VLAD			tFV		
	256	*2048*	*20k*	*256*	*2048*	*20k*
Oxford5k	0.576	0.670	0.724	0.591	0.673	**0.734**
BelgaLogos	0.205	0.246	0.271	0.219	0.241	**0.280**
Holidays	0.597	0.667	0.727	0.610	0.684	**0.737**
Generic	0.460	0.528	0.574	0.473	0.533	**0.584**

Table 2: ***The influence of vocabulary size.*** *Three sets of vocabularies are evaluated for box search, with 256, 2048 and 20k visual clusters respectively. Increasing the vocabulary size leads to better performance for all datasets.*

2.5.4 *Feature space locality by large vocabularies*

In this section we evaluate the effectiveness of large vocabularies which impose locality in feature space by creating small visual clusters. Table 2 lists the retrieval accuracy. It shows increasing the vocabulary size improves the performance in all cases.

Large vocabularies better capture the small details in the feature space, advantageous for instance search where the distinction between close instances of the same category relies on subtle details. However, there is no infinite improvement. We have also tested VLAD200k on Oxford5k and BelgaLogos, and the mAP is 0.723 and 0.266 respectively, no further increase compared to VLAD20k. Creating a GMM with 200k Gaussian components is prohibitively expensive in terms of computation, but we expect the same behavior as VLAD. The quantified differentiation within the visual clusters will be superfluous or even adverse when the visual cluster is so small that the hosted local descriptors represent the same physical region in the real world. Before reaching the gate, large vocabularies are beneficial.

2.5.5 *Feature space locality by exponential similarity*

In this experiment we quantify the add-on value of the proposed exponential similarity, see equation 2.12, which emphasizes close matches in feature space, as compared to the standard dot product similarity. We set $\beta = 10$ for all datasets without further optimization. We embed the evaluation in the box search framework using 20k-vocabularies. As shown in Table 3, the exponential similarity consistently improves over dot-product similarity by a large margin. Exploring a similar idea, the thresholded polynomial similarity in the concurrent work [158] achieves a close performance. We have also experimented with the adaptive similarity [136]. Giving much higher weights to closer matches has the most important effect on the result. Both [136] and our proposal provide this, where our proposal does not need the extra learning step. Putting *disproportionally* high weights on close matches in the feature space is advantageous for instance search, which relies on matches of exact unique details.

	VLAD			**tFV**		
	dot	*exp*	*poly*	*dot*	*exp*	*poly*
Oxford5k	0.724	0.765	0.773	0.734	0.770	**0.778**
BelgaLogos	0.271	0.291	0.296	0.280	0.302	**0.304**
Holidays	0.727	0.772	0.749	0.737	**0.787**	0.767
Generic	0.574	0.609	0.606	0.584	**0.620**	0.616

Table 3: ***The effect of exponential similarity.*** *The value of the exponential similarity, denoted by 'exp', is evaluated within the box search framework using 20k-vocabularies. As compared to the dot-product similarity, denoted by 'dot', the exponential similarity improves the search accuracy in all cases. 'poly' denotes the thresholded polynomial similarity function proposed in the recent work [158].*

	VLAD			**Fisher vector**		
	[9]	[30]	20k^{exp}	[76]	[131]	tFV20k^{exp}
Oxford5k	0.555	0.517	0.765	0.418	-	**0.770**
BelgaLogos	0.128*	-	0.291	0.132*	-	**0.302**
Holidays	0.646	0.658	0.772	0.634	0.705	**0.787**
Generic	0.443	-	0.609	0.395	-	**0.620**

Table 4: ***State-of-the-art comparison.*** *The entries indicated with a $*$ are our supplementary runs of the reported methods on that dataset. Our combined novelty, localized tFV20k with exponential similarity outperforms all other methods by a considerable margin.*

2.5.6 *State-of-the-art comparison*

To compare with the state of the art in generic instance search from one example, in Table 4 we have compiled an overview of the best results from [9, 30, 76, 131] which employ VLAD or Fisher vector. For BelgaLogos where VLAD and Fisher vector have not been applied before, we report results acquired by our implementation. The proposed localized tFV20k with exponential similarity outperforms all other methods by a significant margin. The method is followed by localized VLAD20k^{exp}.

For the newly defined TRECVID50k dataset, which is a factor of 5 to 30 bigger than the other three datasets, and covering a much larger variety, the performance improvement of our subsequent steps is indicated in the rows of Table 5.

2.6 CONCLUSION

We propose locality in generic instance search from one example. As the signal to noise ratio within the bounding box is much higher than in the entire image, localized search in the image for an instance is advantageous. It appears that continuing on the localization in the feature space by using very large vocabularies further improves the

	VLAD	**tFV**
Baseline (*global search*)	0.075	0.096
+ Spatial locality	0.084	0.116
+ 20k vocabulary	0.103	0.131
+ Exponential similarity	0.124	0.144

Table 5: The performance improvement by the three novelties on the TRECVID50k dataset. The dataset is a 50k subset of the TRECVID 2012 instance search dataset [125] with annotations for 21 queries, here applied with 1 example each.

results considerably. Finally, localizing the similarity metric by exponential weighting, improves the result significantly once more.

The combination of spatial locality and large vocabularies either will pose heavy demands on the memory or on the computation. In the standard implementation even a vocabulary of 256 clusters with box search will require a huge 777 gigabytes and over 2,000s of computation to finish one query for Oxford5k. The implementation of [76] achieves an mAP of 0.490 using PCA and product quantization on a 256 vocabulary with a memory of 1.91 gigabytes. This will explode for larger vocabularies. Our implementation with point-indexed representation requires only 0.56 gigabytes for a 20k vocabulary, achieving a vast increment to an mAP of 0.765 with a computing time of 5s. The computation time can be improved further by the use of hierarchical sampling schemes, a topic of further research.

On the newly proposed TRECVID50k dataset, we have set an mAP with one query example of 0.144. On the commonly used datasets Oxford5k, BelgaLogos, and Holidays we achieve an average performance increase from 0.395 for the recent [76], and 0.443 [9] to 0.620 for our generic approach to instance search with one example proving the value of locality in the picture and feature space for this type of search. The method does not only improve the accuracy but also delivers a reliable localization, opening other avenues, most notably complex queries asking for spatial relations between multiple instances.

3

WORDS MATTER: SCENE TEXT FOR IMAGE CLASSIFICATION AND RETRIEVAL

3.1 INTRODUCTION

[1]Fine-grained classification is the problem of assigning images to classes where instances from different classes differ slightly in the appearances *e.g.*, flower types [122], bird [177] and dog species [97], and aircraft models [103]. In contrast to coarse object category recognition *e.g.*, cars, cats and airplanes, low-level visual cues are often not sufficient to make distinction between fine-grained classes. Even for human observers, fine-grained classification tasks usually require expert and domain specific knowledge. Accordingly, most recent works also integrated such domain specific knowledge into their solutions. For instance, dogs have ears, nose, body, legs *etc.*, and the differentiation of dog species relies on the subtle differences in these parts. Different bird species have different wing and beak appearances, and such differences in local parts provide the critical information to categorize different bird types. [97, 181, 190] exploit the part information and extract features from particular parts for better birds and dogs recognition. In this work, we make use of the domain specific knowledge of *buildings*. We exploit the recognized text in images for fine-grained classification of building types. The building types studied in this work are places-of-businesses (*e.g.*, bakery, cafe, bookstore *etc.*). Automatic recognition and indexing of business places will be useful in many practical scenarios. For instance, it can be used to extract information from Google street view images and Google Map can use the information to provide recommendations of bakeries, restaurants close to the location of the user.

Most of the time, the stores use text to indicate what type of food (pizzeria, diner), drink (tea, coffee) and service (drycleaning, repair) they provide. This text information is helpful even for human observers to understand the content of the store. For instance, in Figure 10, the images of two different buildings (*pizzeria* and *bakery*) have a very similar appearance. However, they are different types of business places. It is only possible with text information to identify what type of business places these are. Moreover, text is also useful to identify similar products (logo) such as *Heineken*, *Foster* and *Carlsberg*. Therefore, we propose a multimodal approach which uses recognized text and visual cues to do better fine-grained classification and logo retrieval.

The common approach to text recognition in images is to detect text first before they can be recognized [71, 175]. The state-of-the-art word detection methods [92, 100, 119, 169, 178] focus on obtaining a high f-score by balancing precision and recall.

1 Under review for publication in *IEEE Transactions on Multimedia* [83]

Figure 10: bakery and pizzeria example images. The two buildings are visually similar. Text can be used to differentiate the two shops.

However, instead of using the f-score, our aim is obtain a high recall. A high recall is required because textual cues that are not detected will not be considered in the next (recognition) phase of the framework. Unfortunately, there exists no single best method for detecting words with high recall due to large variations in text style, size and orientation. Therefore, we propose to combine character candidates generated by different state-of-the-art detection methods. To obtain robustness against varying imaging conditions, we use color spaces containing photometric invariant properties such as robustness against shadows, highlights and specular reflections.

The proposed method computes text lines and generates word box proposals based on the character candidates. Then, word box proposals are used as input of a state-of-the-art word recognition method [70] to yield textual cues. Finally, textual cues are combined with visual cues for fine-grained classification and logo retrieval. The proposed framework is given in Figure 11.

The work has the following contributions. First, this work combines textual and visual cues for fine-grained classification and logo retrieval. In contrast to [85] which extracts textual cues at character level, the proposed method extracts textual cues at word level. The proposed method reaches state-of-the-art results on both tasks. Second, to extract the textual cues, a generic and computationally efficient word proposal algorithm which aims at high recall is proposed without any training involved. The proposed algorithm obtains state-of-the-art recall for word detection for a limited number of word box candidates. Third, contrary to what is widely acknowledged in text detection literature, we experimentally show high recall in word detection is more important than high f-score at least for both applications considered in this work. Last, this work provides a large text detection dataset (10K images with 27601 word boxes). This dataset will be made publicly available.

3.2 RELATED WORK

Word Detection. Word detection consists of computing bounding boxes of words in images. Existing word detection methods usually follow a bottom-up approach. Character candidates are computed by a connected component [36, 119] or a sliding window approach [71, 169, 175]. Candidate character regions are further verified and combined to form word candidates. This is done by using geometric, structural and appearance

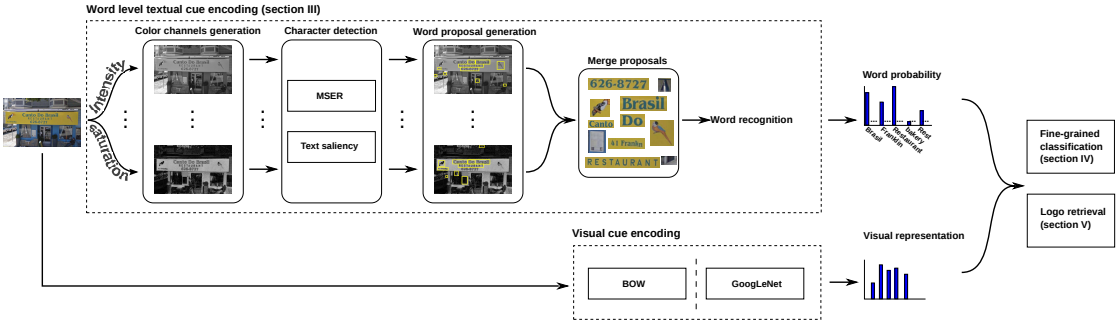


Figure 11: Pipeline of our multimodal approach. Text is encoded at a word level and utilized for fine-grained classification and logo retrieval. A generic and fully unsupervised word box proposal method is proposed to detect words in images. The method uses different color spaces and character detection algorithms (MSER [105] and text saliency [84]). The word box candidates are used as input for a state-of-the-art word recognition method [70] to perform word-level encoding. An English vocabulary consisting of around 90k words is considered [70]. For the visual cues, bag-of-words (BOW) and GoogLeNet features [153] are used. The multimodal approach combines the visual and textual cues.

properties of text and is based on hand-crafted rules [36] or learning schemes [71, 169]. State-of-the-art word detection methods [92, 119, 169] focus on high f-score by the trade-off between recall and precision. Strict rules are used in character detection and word formation to keep only boxes that most likely contain words. As a consequence, methods aiming for high f-score may miss a number of correct word boxes. In contrast, we propose to generate word boxes with the goal to include all words, *i.e.*, high recall. We use recall in text detection because our aim is not to miss correct word boxes with the cost of introducing false detections.

Our work is similar to the recent works [51, 69] in terms of providing word box proposals. [69] combines two generic object proposal outputs, namely Edge Boxes [198] and Aggregate Channel Feature Detector [32], as preliminary word box proposals. Then, these proposals are filtered using the HOG [29] feature with a Random Forest text/non-text classifier [16]. Finally, the remaining word box proposals are processed using a convolutional neural network regressor to refine the coordinates of these word boxes. [51] performs an over-segmentation using maximally stable extremal region (MSER) algorithm with flexible parameters. Then, the segments are grouped together using distance metrics related to text (*e.g.* color, stroke width *etc.*). Finally, weak classifiers are used to obtain a text-likeliness measure for these word candidates. In contrast, our word box generator is uniquely designed to detect text in images without any training involved. Moreover, [51, 69] in the end aim at high f-score word recognition whereas this work aims only at high recall. We experimentally verify high recall is more important than high f-score for the applications considered in this work. Further, different from [51, 69] which address word recognition, the aim of this work is to combine textual and visual cues for better fine-grained recognition and logo retrieval.

Text Recognition. Text recognition approaches can be categorized into two groups: character and word based methods. Character based methods first recognize single characters, then form words [112, 113, 124]. Recent work [6, 50, 70] shows that entire-

word recognition performs better than recognizing characters first and then forming words. In this work, we follow the state-of-the-art word recognition approach [70] to encode the textual cues.

Textual Cues. Mishra *et al.* [111] propose to use textual cues for query-by-text image retrieval. Given a query text, the method assigns scores to images based on the presence of the query characters. Additional pairwise spatial constraints between characters are used to refine the ranking. Karaoglu *et al.* [85] propose to use textual cues in combination with visual cues for fine-grained classification. Bi-grams are computed based on recognized characters in images. These bi-grams are used to encode the textual cues. In contrast, this work performs a word-level textual cue encoding. Moreover, the proposed method aims at high recall word detection which leads to combine state-of-the-art text detectors performed in various color spaces.

Fine-grained Classification. Many recent works in fine-grained classification exploit domain specific knowledge. Dogs and birds are composed of a number of semantic parts, such as head, body and tail. [181, 190, 192] use parts for better fine-grained recognition. [190] learns part detectors and localizes the parts to isolate the subtle differences in specific parts. [181] shows the hidden layers of a deep neural network are actually part detectors and uses the filters in the hidden layers to detect specific bird and dog parts. [192] generates multi-scale part proposals and selects useful parts. [45] presents another successful use of domain specific knowledge for bird species recognition. It exploits the fact that birds have rather fixed poses and fits an ellipse to represent the overall shape of a bird. In this work, we exploit the domain specific knowledge for building types classification. In our case, the domain knowledge is the scene text in the building images. We propose a multimodal approach to fine-grained building type classification by fusing the textual and visual cues. A recent paper from *Google* [114] also studies the classification of different business places. [114] only considers visual cues for classification while in our work we show that adding textual cues significantly outperforms methods that only use visual information.

3.3 WORD-LEVEL TEXTUAL CUE ENCODING

In order to extract the textual cues from the image, a two-step procedure is followed. In the first step, word box proposals are generated to locate the words in the image. In the second step, the word proposals are used as input to a word recognizer to form the word-level representation.

3.3.1 *Word Box Proposals*

High recall. When a word in an image is not detected or localized incorrectly, it is not possible to identify it. Our aim is to obtain high recall with the cost of false positives. To this end, the proposed method uses a complementary set of character detection algorithms and color invariant spaces.

Low computational cost. The word box proposal method needs to be efficient especially for large scale scenarios. Further, the number of possible word box candidates (*i.e.*, proposals) should be as low as possible.

Generic. We aim for a generic word proposal method. No need for tuning the method for different alphabets or datasets.

Therefore, we propose an efficient and fully unsupervised bottom-up approach. First, characters are detected by a text-independent approach. Then, these detected characters are filtered based on geometric and appearance properties. Finally, they are grouped to generate word box proposals.

Character Detection

As stated earlier, there exists no single character detection algorithm that is robust against all variations in text style, size and orientation and imaging conditions. Therefore, we propose to compute character candidates using two methods with different strengths, *i.e.*, text saliency [84] and Maximally Stable Extremal Regions(MSERs) [105].

In [84], a text saliency map is computed using scene background. It is assumed that background pixels are uniformly colored *e.g.*, windows, boards, roads, buildings, fences *etc.*, and that they contrast with text regions. Accordingly, the method uses background homogeneity to form connectivity between background pixels. The method selects initial background seeds and grows these seeds iteratively until all background pixels are covered (detected). Assuming that text regions have strong contrast with the background [15], text regions will remain uncovered by the region growing algorithm. Finally, the background image is subtracted from the original image to obtain a text saliency map, which is further binarized using [42] to obtain character candidates.

Detecting background relies heavily on correctly selecting initial seeds. As text is salient [81, 168], the background that highly contrasts with text is assumed to be non-salient. Moreover, text rarely appears at image borders. Therefore, seeds are selected from non-salient image regions which are refined by image boundary information.

Color and Curvature Saliency. Color edges are useful to detect if a region belongs to the background. Color is usually homogeneous for different backgrounds such as roads, fences and skies. Furthermore, color edge responses correlate with colorful text/background transitions. To exploit this, we use the color boosting algorithm [164] to enhance the saliency of colorful text/background transitions and to select the background seeds based on non-salient regions in the color saliency map.

The color saliency measure is inappropriate for colorless edge transitions, see the right two images in Figure 12. Therefore, in addition to color, curvature (L) is used for colorless edges. Because of the text/background contrast, text regions result in high curvature even for colorless edge transitions. Non-salient regions in the curvature saliency map are used to select the background seeds.

Saliency Refinement Using Background Priors. The image regions which do not have strong responses in color and curvature are considered to be background pixels. However, some of the regions which have high saliency response may also belong to the background. Therefore, salient regions which are unlikely to belong to text are suppressed using background priors, *i.e.*, text is mostly located in the center of the image [84]. Hence, salient regions, which are connected to the borders, are suppressed in the color and curvature saliency maps.

The refined saliency maps are used to select the background seeds. Specifically, the refined saliency maps are normalized to $[0, 1]$ and linearly combined. Regions without

Figure 12: Saliency map samples: Original images, color saliency and curvature saliency (top to down order). It is shown that text edges are detected better with color saliency for the first two images whereas curvature saliency works better for the last two images.

any response on this combined saliency map are considered as background seeds. The background of the input image is reconstructed using morphological operators [166]. The reconstruction is performed by a conditional dilation (δ). Conditional dilation is a basic dilation which is conditioned by a mask image (*i.e.*, the single-channel image I in our case). The conditioning is obtained by defining the output as the intersection of the dilation and I, formulated by:

$$\delta_I(J) = (J \oplus B) \wedge I\,, \tag{3.1}$$

where $J \oplus B$ stands for the dilation of J (the image consisting of only background seeds) and B (the structuring element), and $\wedge$ denotes the element-wise minimum.

To obtain a reconstructed background image (ρ) of image I, given the image consisting of the initial background seeds, J_0, Eqn. 3.1 is executed until stability is reached. That is, starting from the initial background seeds J_0 repeat $J_n = \delta_I(J_{n-1})$ until $J_n = J_{n-1}, (n = 1, 2, 3...)$ and obtain ρ by $\rho = J_n$.

Text saliency computation does not require any tuning for varying text size, style and orientation, and is robust to image noise. However, due to the information loss caused by the image boundary priors and the binarization, the method may miss characters. To compensate for this, we enable MSER as another character detection algorithm. MSERs define an extremal region as a connected component of which image values remain stable within the boundary and highly contrast against boundary pixels [105]. MSER regions are widely in use for character detection [23, 119]. MSER is suited for character detection because text regions are usually designed to have uniform appearance (color).

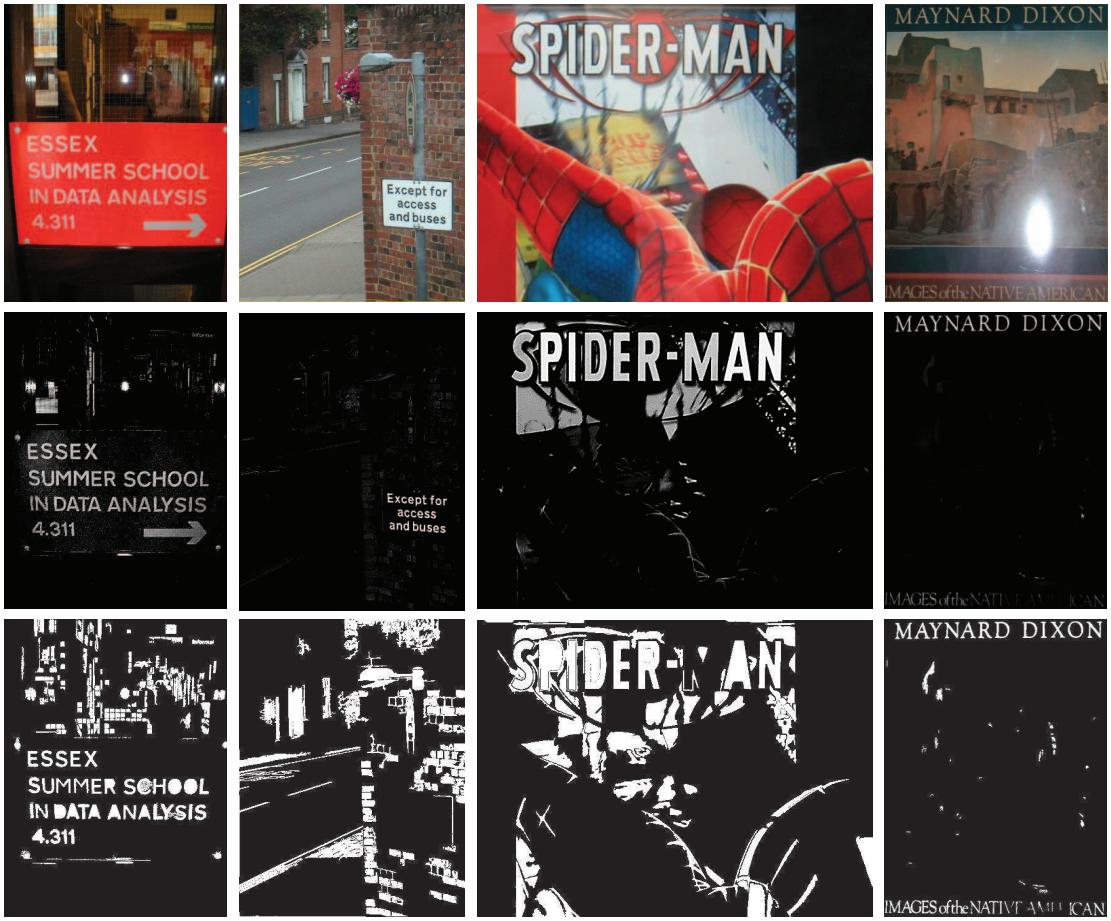


Figure 13: Original images (up), text saliency (middle) and MSER character detections (down) obtained. Text saliency method is robust against changes in text size and noise while MSER detects characters at image boundaries.

Further, they usually have high contrast with their surroundings. However, MSER has certain shortcomings for character detection such as detecting characters in blurry and noisy images [23]. Moreover, MSER is sensitive to character sizes due to the parameters used to define stable regions. In fact, the MSER and text saliency results are, to a certain extent, complementary. Figure 13 illustrates complementary properties of MSER and saliency methods.

Complementary Color Spaces

Images are captured under uncontrolled illumination conditions. Therefore, text regions may be influenced by different photometric changes such as shadows and specular reflections. A uniformly colored character may vary in intensity due to shadows or highlights. Hence, these shadows or highlights may negatively influence the pixel connectivity for a uniformly colored character.

To compensate for this, the proposed method computes the character candidates using a variety of color spaces containing a range of invariant properties. The two channels, (O_1, O_2), from the opponent color space [38], Saturation (S) and Hue (H) from *HSV* [162], and (I) from gray scale are considered in the proposed method (see

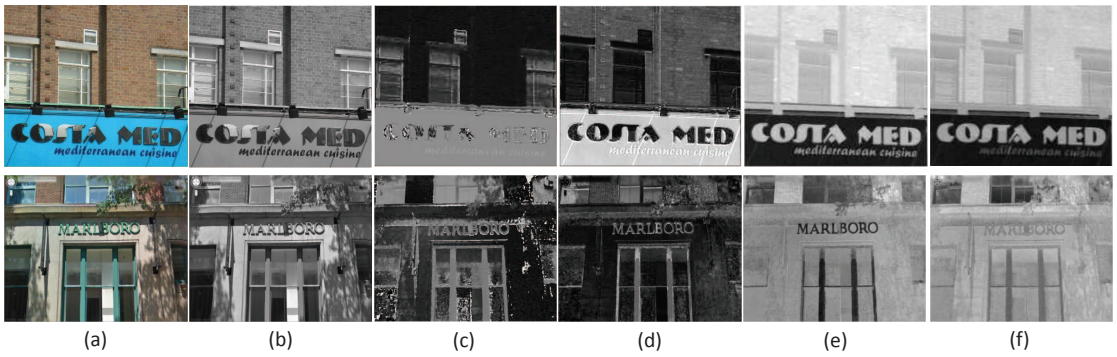


Figure 14: Examples of different color channel responses: (a) Original image, (b) Gray-scale, (c) Hue, (d) Saturation, (e) O_1 and (f) O_2. It is shown that color channels have different responses to photometric changes e.g. *shadow and highlights, based on their invariant properties.*

	I	O_1	O_2	S	H
Highlights	-	+	+	-	+
Shadows	-	-	-	+	+

Table 6: Color spaces and their invariant properties. I is the gray scale. (O_1, O_2) are the two channels from the opponent color space [38]. Saturation (S) and Hue (H) are from the HSV color space [162]. '+' means invariant. In this work, we use all these color spaces with different invariant properties to cope with the photometric changes in natural images.

Figure 14). The invariant properties are summarized in Table 6. Figure 14 illustrates the color channel responses for photometrical changes.

Character Filtering

The character candidates provided by the method described before may consist of non-character regions. Our method for character filtering is based on state-of-the-art text detection systems [23, 36] to filter out non-character regions efficiently.

Size. The proposed method limits the height of a character candidate to be greater than 5 pixels and the area to contain more than 50 pixels [36]. If the character is too small, the information it carries is limited. Therefore, it is likely that even if these regions are not eliminated, recognition on these regions would fail.

Aspect ratio. Most of the real characters have a width-height ratio close to 1 [36]. Therefore, the proposed method limits the aspect ratio of character candidates to be a value between 0.1 and 10. These values are reported in [36] to be conservative enough to still keep characters such as 'i', 'I' or '1'. This process filters out text-like items in images such as fences and branches of trees.

Solidity. The solidity is defined as the proportion of the number of character pixels to the convex area which covers the text candidate. It has been observed that text regions have low solidity [23]. Therefore, the proposed method eliminates character candidates which have high solidity (>0.95) and longer width than height. Longer width is to avoid removing characters like 'i' and 'l'. This process filters out brick-like image regions which have solidity close to 1. Solidity threshold is set to be conservative enough to keep characters like 'w' and 'm'.

Contrast. Pixels at character borders usually have high contrast and the contrast decreases with the distance to the borders. As a result, the box which neatly covers a character will have a higher average contrast than its slightly expanded version. Therefore, the proposed method eliminates the character candidates which do not meet this condition. Contrast (C) of an image pixel (p) is calculated by $C_p = \sqrt{I_x^2(p) + I_y^2(p) + I_x(p)I_y(p)}$, where I_x, I_y are the first order image derivatives (x and y dimensions) in intensity I.

A character candidate satisfying all these conditions is remained for further processing. This filtering step removes those obvious non-character candidates to reduce computational cost in following steps. Figure 15 shows filtered character candidates for each condition.

Word Box Proposal Generation

The next step is to compute word box proposals using character candidates. We consider combinations of character candidates as potential words. However, it is computationally expensive if all possible combinations are considered. And, due to the nature of text, characters within a word cannot have arbitrary positions and sizes [36, 39, 92, 104, 117, 118]. Therefore, as the first step of computing word box proposals, we generate text lines to restrict the selection of combinations by linking character candidates based on five pair-wise constraints. In Figure 16, the two boxes stand for two character candidates with (x_1, y_1), $height_1$ and $width_1$ being the coordinates

Figure 15: Samples for character candidate filtering. The green, red and yellow (different colors are used to highlight boxes) boxes represent filtered character candidates after corresponding filtering condition is applied (i.e. *size, aspect ratio, solidity and contrast).*

of the top-left corner, height and width of the box covering the first character. The box of the second character is defined likewise.

The five pairwise constraints are as follows:

(a) Distance between two character centers is smaller than 2.5 times of the longer axis of the character box [36,39,118]. $Distance < \max([height_1, width_1, height_2, width_2]) \times 2.5$ where $Distance = (x_2 + \frac{width_2}{2}) - (x_1 + \frac{width_1}{2})$. 2.5 is considered to allow one missed character in between.

(b) The ratio of the vertical displacement and horizontal offset is no greater than 0.2 [92, 118], formally expressed by $\frac{VD}{Distance} \leq 0.2$ where $VD = |(y_2 + \frac{height_2}{2}) - (y_1 + \frac{height_1}{2})|$ and $Distance$ as defined in (a). Text is mostly horizontally aligned.

(c) The height ratio of two characters is not greater than 2 [36, 39, 118], *i.e.*, $0.5 \leq \frac{height_1}{height_2} \leq 2$. Two characters of a word should have similar height and 2 is considered to allow the case of a lower-case character following a capital.

(d) Two characters must not overlap more than 0.1, formally, $\frac{Area(Char_1 \cap Char_2)}{Area(Char_1 \cup Char_2)} \leq 0.1$. Characters of a word usually do not overlap except in special cases, *e.g.*, italic.

(e) The bottom of one character is below the center of the other [118], *i.e.*, $(y_1 + height_1) \geq \frac{y_2 + height_2}{2}$ and $(y_2 + height_2) \geq \frac{y_1 + height_1}{2}$.Two consecutive characters of a word are usually well aligned for easy reading.

As the second step, we compute word box proposals by considering all possible combinations of character candidates within a text line. A combination of character candidates corresponds to the box covering the union of the character candidates. The proposed method starts with a single character candidate as a word proposal. The reason is that when the characters of a word are connected the word is covered by only one character candidate.

Word box proposals are generated from each character detection algorithm and color space independently and then combined.

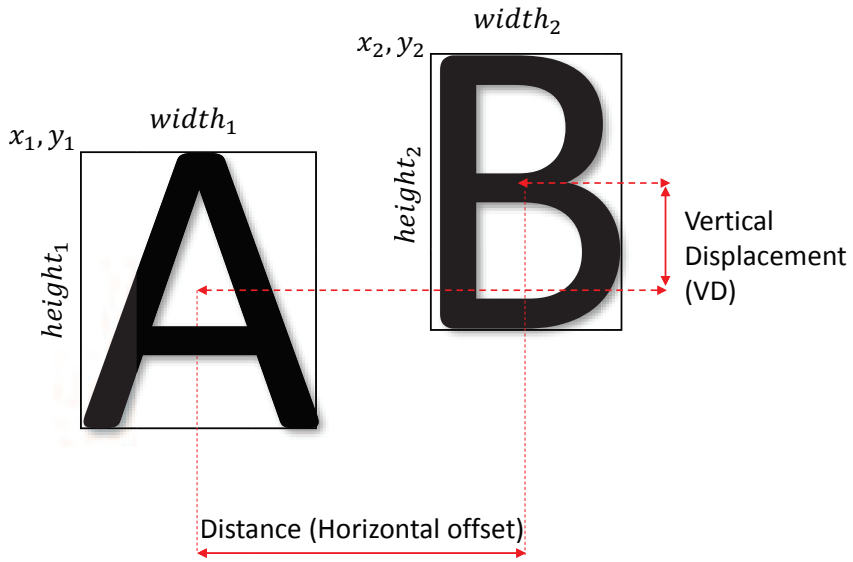


Figure 16: An illustration on the notions of two character candidates. This illustration is used to elaborate the pairwise constraints.

3.3.2 *Word Recognition and Textual Cue Encoding*

Section 3.3.1 generates word box proposals. To recognize words, we employ a state-of-the-art word recognition approach [70]. [70] formulates word recognition as a multi-class classification problem, where a word from a predefined English vocabulary is treated as one class. A convolutional neural network classifier with four convolutional layers and two fully-connected layers is used to solve the classification problem. We refer to [70] for the details of the network. The network takes a word box proposal b as input and produces for each word w a probability of the word being present in the box, $P(w|b)$. The probability is modeled by the softmax scaling of the final multi-way classification layer. As a result, each word box proposal is represented by a n-dimensional feature, where n is the number of words in the vocabulary. In this work, we use the model[2] provided by the authors of [70]. The model considers a vocabulary of 88,172 words and is trained using synthetic data. We encode the textual cues in an image by summarizing the representations of word box proposals with average pooling. Each dimension of the resulting image feature represents the probability of the corresponding word being present in the image.

3.4 FINE-GRAINED CLASSIFICATION

Fine-grained classification is the problem of the categorization of subordinate-level categories such as bird species [177], flower types [122] and building types [85]. The small inter-class visual differences and the large intra-class variations make fine-grained classification challenging. In this section, in addition to visual features, we exploit the use of textual cues in the images for fine-grained image classification.

2 http://www.robots.ox.ac.uk/~vgg/research/text

3.4.1 *Dataset and Implementation Details*

Dataset. We use the *Con-Text dataset* proposed in [85]. The dataset is for fine-grained classification of business places *e.g.*, *Cafe*, *Bookstore* and *Pharmacy*. The dataset consists of 24, 255 images from 28 categories. The dataset is divided into three folds. Experiments are repeated three times, each time using two folds as training and the other as testing. We report the mean performance over the three runs. Average precision is used to measure the performance.

Dataset Annotation. To study the influence of precision and recall for word detection in the context of fine-grained classification, we have annotated text regions for the first 10 classes of the dataset (in alphabetical order). All the text (Latin alphabet) visible and recognizable has been annotated. The annotated dataset consists of 9131 images. 5219 of these images contain at least one word box. In total there are 27601 word boxes annotated.

Implementation notes. Three visual-only classification baselines are considered. All the three visual baselines employ one-versus-rest SVM classifiers for classification, while the differences lie in the employed visual representations. First, as in [85], we use a standard bag of visual words representation with 3×1 and 2×2 spatial pyramid, denoted as *BOW*.

Second, as image representation, we use the *L*2 normalized output of the last average pooling layer of the ImageNet-pretrained GoogLeNet [153], denoted by *DEEP*. The network is pre-trained on the 1000 ImageNet categories[3] [143], available in the Caffe library [78].

Third, we fine-tune the pretrained GoogLeNet with a 28-way softmax classifier on the *Con-Text dataset*. After fine-tuning, the last average pooling layer output of network is used as the image representation. This visual baseline with features from fine-tuned GoogLeNet is denoted by *DEEP-FT*. The details of the fine-tuning are as follows. The learning rate is initially set to be 0.001, and is decreased by a factor of 10 every 5 epochs. The network is fine-tuned for 20 epochs. The weight decay parameter equals 0.0005. The network is fine-tuned using SGD with momentum which is set to be 0.9.

For text-based classification, the textual cues are extracted as described in Section 3.3.

Libsvm [20] is used for classification. The histogram intersection kernel is employed for *BOW*, following [85], while linear kernel is adopted for *DEEP*, *DEEP-FT* and the proposed textual cues. Textual and visual cues are combined by kernel fusion. Specifically, the visual-based kernel and textual-based kernel matrices are computed independently. Then the two kernel matrices are summed up with equal weights to generate the final kernel matrix. In all experiments, we use the default value for the C parameter (=1) without tuning.

3.4.2 *The Influence of Word Detection Precision and Recall on Fine-grained Classification*

We use the annotated 10 classes to analyze the effect of word detection precision and recall on fine-grained classification. Therefore, we systematically change recall or preci-

3 http://www.image-net.org/challenges/LSVRC/2012/

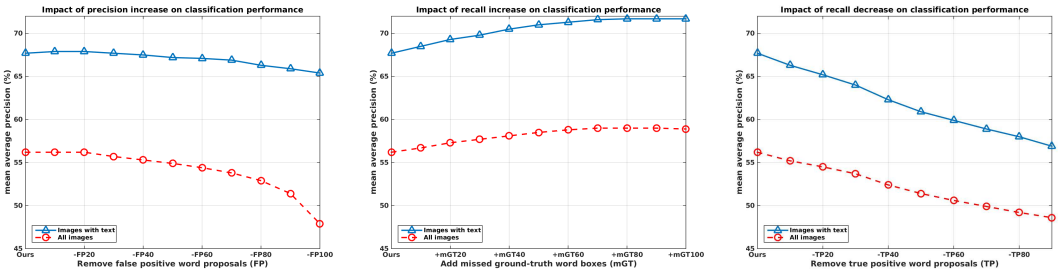


Figure 17: The influence of the precision and recall change in word detection on fine-grained classification performance (evaluated on the 10 annotated classes). Left: Increasing precision by removing the false positive detections (FP) from the automatically generated set of proposals does not improve the classification performance. '-FP20' denotes removing 20% of the false positives. Middle: We systematically increase the recall by adding the missed ground-truth word boxes (mGT) on top of the automatically generated set of proposals. The classification performance keeps increasing as the word detection recall increases before it saturates. '+mGT20' denotes adding 20% of the missed ground-truth word boxes. Right: Decreasing the word detection recall by removing the true positive detections (TP) from the automatically generated set negatively influences the classification performance. '-TP20' denotes removing 20% of the true positives. This set of experiments show that word detection recall is more crucial than precision for the classification performance.

sion and evaluate the classification performance. Within this section, only textual cues are used.

Performance on images without text

Not all images in the dataset contain text. However, the proposed method may generate candidate word proposals in non-textual regions in the image. The method uses the character candidate detector using MSER and saliency. Consequently, regions of interest, other than text, may also be detected. We have evaluated the classification performance using the 'textual cues' encoded by the proposed method on images without text. Interestingly, it achieves 28.9% in mAP, significantly better than random guessing, although the textual cues are much more effective on images with text (67.7% in mAP). This indicates some salient non-text patterns within the same class could be consistently detected and similarly encoded. In the following analysis, we consider two cases, one with images containing text and the other considering all images.

The influence of word detection precision

To study the influence of word detection precision on fine-grained classification, we increase the precision while keeping the recall unchanged by removing the false positive detections (FP) from the generated word proposals. Figure 17 (left) shows that increasing the precision does not improve the classification performance.

Interestingly, this experiment has brought the following additional insight. The classification performance actually decreases when too many false positives are removed from the generated word proposals, especially when all images are considered ('All images'). There are two reasons for this. (1) The proposed word proposal method may detect salient but non-text regions. And some salient non-text patterns within the same class could be consistently detected and similarly encoded. Consequently, some false positive word proposals may contribute positively to the classification, especially for those images without text. This has been discussed before. This is also the reason for the decrease in classification performance when removing too many false positives. This is more significant on 'all images' than 'images with text' as shown in Figure 17. (2) The boxes, that cover the text regions for less than 50% overlap with the ground-truth, are treated as false positives. These boxes may contain parts of words or contain complete words with extra background regions. Removing such boxes may have a negative influence on the classification results.

Additionally, we study the influence of precision decrease by adding the generated word proposals (*Ours*) on top of the manually annotated word boxes (*GT*). The classification performance of *GT+Ours* (with a precision of 6.2%) is 75.7% whereas *GT* (with a precision of 100%) is 76.1%. The significant drop in precision from 100% to 6.2% results in a marginal decrease in classification performance.

These experiments indicate that the false positive word proposals generated by the proposed method do not negatively influence fine-grained classification. However, it is worth to mention that it is still desirable to produce a limited number of word proposals for memory and efficiency concerns.

The influence of word detection recall

First, we evaluate the influence of a recall increase on the classification rate. We systematically increase the recall by adding the missed ground-truth word boxes (mGT) on top of the automatically generated set of proposals. As shown in Figure 17 (middle), the classification performance keeps increasing as the word detection recall increases before it saturates.

Second, we decrease the recall by removing the true positive word proposals (TP) from the automatically generated set. The results in Figure 17 (right) show that decreasing the word detection recall negatively influences the classification performance.

Note that even when 90% of the true positive word proposals are removed, the classification performance is acceptable. There are two reasons for this. (1) As discussed before, the word proposal method is able to consistently detect a number of salient but non-text patterns which are contributing positively to the classification. (2) The boxes that cover the text regions, with less than 50% overlap with the ground-truth, are treated as false positive. Therefore, even when all true positives are removed, these boxes contribute positively to the classification result.

Additionally, we evaluate the performance only using ground-truth boxes. In the case where only images containing text are considered, the performance is 76.1%. When all images are considered, the performance is 54.2%, outperformed by *Ours* (56.2%). When using the ground-truth boxes, the performance on images with no text is random, while when using our generated word proposals, the classification on images with no text

	Performance (mAP%)	
	Ours	*Characterness [92]*
Images with Text	67.7	37.8
All Images	56.2	30.6

Table 7: Comparison to state-of-the-art text detection [92]. [92] aims at a high F-score. The recall, precision and F-score values of the proposed method are 64.7%, 4.7% and 8.7% respectively while the values of [92] are 19.3%, 25.3% and 21.9%. A high recall value is more effective than a high f-score for the fine-grained classification problem.

is 28.9% in mAP, much better than random guessing. This is why when all images are considered, including both images with text and images without text, the performance of using our generated boxes is slightly better than the result of using ground-truth boxes.

Comparison to state-of-the-art text detection

We compare the proposed word detection method with a recent state-of-the-art text detection approach [92]. The textual cue encoding and the classification steps are kept same. [92] aims for a high f-score, like other state-of-the-art text detection methods [119, 169].

The recall, precision and f-score values of the proposed method are 64.7%, 4.7% and 8.7% respectively while the values of [92] are 19.3%, 25.3% and 21.9%. Compared to [92], the proposed method achieves a significantly higher recall but a lower precision and F-score. In terms of fine-grained classification performance, as shown in Table 7, the proposed method (*Ours*) significantly outperforms [92].

3.4.3 *Performance evaluation on 28 classes*

In this section, we use all 28 classes for evaluation and conduct two experiments. First, we evaluate the effectiveness of the textual cues encoded by the proposed method on the 28-class classification problem. Second, we compare the classification performance of word-level and character-level textual cue encoding.

Experiment I. Three different ways to generate word box proposals are considered: (1) the proposed method using all color channels, denoted by *full*, (2) the proposed method using only the gray scale, denoted by *gray-only*, and (3) a state-of-the-art text detection approach [92] aiming at a high f-score, denoted by *characterness*. We evaluate the sets of word box proposals generated by these three different ways separately while keeping the textual cue encoding and classification steps the same.

As shown in Table 8, *full* always outperforms *gray-only* and *characterness* thanks to a higher recall in word detection. The proposed textual-only classification method obtains a mean average precision of 38.3%, outperforming *BOW* (34.0%). The combination of textual and visual cues improves the visual-only baseline by 21.8%, 17.7% and 14.2% for *BOW*, *DEEP* and *DEEP-FT* respectively. It can be derived that recognized words in images contain discriminative information and that it is complementary to visual cues.

	Performance (mAP%)
Textual-only (full)	**38.3±0.9**
Textual-only (gray-only)	33.1±0.5
Textual-only (characterness [92])	20.2±0.6
Visual-only (BOW)	34.0±0.3
Visual-only (DEEP)	53.3±0.08
Visual-only (DEEP-FT)	**60.3±0.2**
Textual (full) + Visual (BOW)	55.8±1.0
Textual (gray-only) + Visual (BOW)	52.0±0.6
Textual ([92]) + Visual (BOW)	42.7±0.4
Textual (full) + Visual (DEEP)	71.0±0.5
Textual (gray-only) + Visual (DEEP)	68.7±0.3
Textual ([92]) + Visual (DEEP)	62.0±0.2
Textual (full) + Visual (DEEP-FT)	**74.5±0.8**
Textual (gray-only) + Visual (DEEP-FT)	72.7±0.5
Textual ([92]) + Visual (DEEP-FT)	67.5±0.6

Table 8: Fine-grained classification performance on Con-Text dataset. The textual cue encoded by the proposed method is effective. It is complementary to the visual information. Textual-only (full), Textual-only (gray-only) and Textual-only (characterness [92]) only differ in word detection. Textual cue encoding and classification steps are kept the same. 'full' outperforms 'gray-only' and 'characterness' [92] thanks to a higher recall in word detection.

	Performance (mAP%)		
	text only	*visual only*	*text and visual*
Word Level [this work]	38.3	34.0	**55.8**
Character Level [85]	15.6	32.9	39.0

Table 9: Comparison to the state-of-the-art [85] which encodes textual cues at character level. Numbers are taken from [85]. Where the visual-only performance is compatible, the proposed method outperforms [85] by a large margin. It can be derived that representing the textual information at word level is more effective than at a character level.

Figure 18 shows the per-class performance. The low performance of textual cues is due to the lack of scene text, *e.g.*, for classes as *Bistro* and *Massage Center*. However, combining visual and textual cues improves visual-only on all classes. The performance improvement is the highest on the classes where visual cues are not sufficient and textual cues are discriminative, *e.g.*, *Pawn Shop*, *Dry Cleaner* and *Steak House*.

Experiment II. We compare our approach with the state-of-the-art [85] which extracts the textual information at a character level. To ensure a fair comparison, in this experiment we use *BOW* for the visual-based classification as [85]. Table 9 summarizes the results. Our method outperforms [85] by a large margin (16.8%). It shows that representing the textual information at a word level is more effective than at a character level.

3.5 LOGO RETRIEVAL

In logo retrieval, the objective is to retrieve all images of a specific logo from an image collection, *e.g.*, *Heineken*, given one image example of that logo as query. Logo retrieval is useful for measuring brand exposure. Logo is a special type of objects where text can be part of the object. Examples are *Starbucks*, *Ford* and *Google*. Previous works [80, 139, 140, 156] do not consider the recognized text of the logo. These methods treat the text of the logos the same as other visual patterns. In contrast, we explicitly extract the word-level textual cues in the logos and utilize it for logo retrieval.

3.5.1 *Dataset and Implementation Details*

Dataset. We evaluate our approach on *FlickrLogos-32* [141]. *FlickrLogos-32* has 32 brand logos, *e.g.*, *Google*, *Coca-cola* and *DHL*. We follow the retrieval setting of [140], which defines a set of 960 queries, 30 per logo, and a search set of 4280 images in total. The search set consists of 1280 logo images, 40 per logo, and 3000 non-logo images.

Implementation notes. The common method for logo retrieval is to use low level feature matching. In line with this paradigm, two visual baselines are considered. First, we use the available BOW representations with a visual vocabulary of 1 million visual words [140], denoted by *BOW*. Second, we implement another visual baseline based on aggregated selective match kernels [158], denoted by *ASMK*. The visual vocabulary has 20000 visual words. The kernel we use is a thresholded 4-degree polynomial kernel expressed by $\sigma(\mu) = [\mu > 0]\mu^4$, where the square bracket stands for the Iverson bracket.

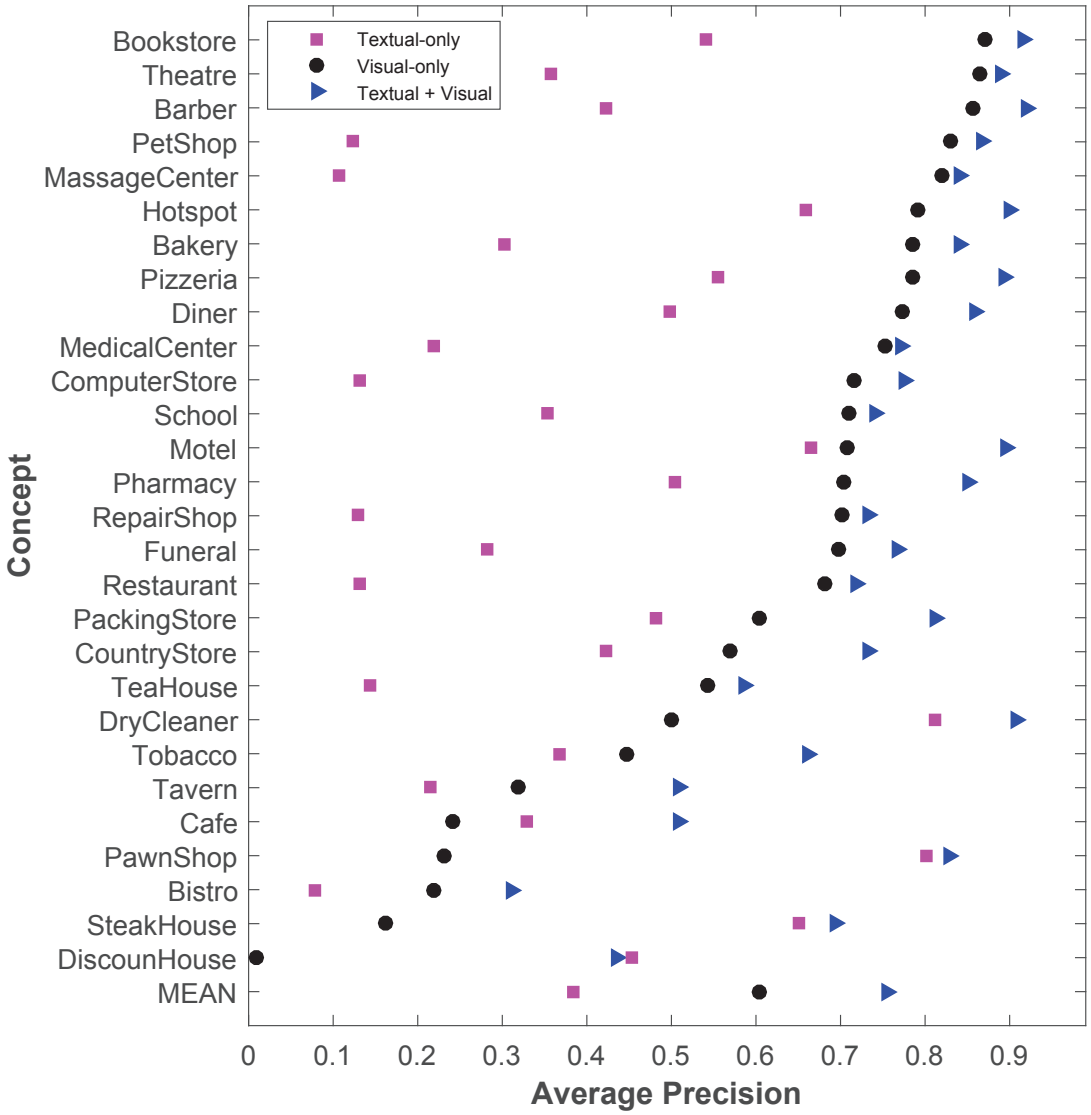


Figure 18: Fine-grained classification performance for each class. Adding textual cues improves the performance on all classes. The proposed multimodal approach improves the visual-only baseline (DEEP-FT) from 60.3% to 74.5% in mean average precision. Textual-only has average precision values from 10% to 60% and visual-only has values from 0% to 80% largely, whereas multimodal approach guarantees at least 50% except two classes (Bistro and DiscountHouse) up to 90%.

For textual cues, we encode the images in the same way as in the previous fine-grained classification application, detailed in Section 3.3. For the query images, we use the query bounding boxes to only keep the word box proposals that overlap with the query boxes. The textual representations are normalized to unit length and cosine similarity is used to rank the images.

To combine the visual and textual cues, we perform a late fusion on the similarity scores obtained from the two modalities. Both sum fusion, expressed by $S_{fusion} = S_{visual} + S_{textual}$, and product fusion, expressed by $S_{fusion} = S_{visual} * (S_{textual} + \epsilon)$ are tested. S_{fusion}, S_{visual} and $S_{textual}$ are the fused score, visual-based score and textual-based score respectively. ϵ is a small constant value added to handle cases where no text has been detected. Sum fusion requires the two scores to be roughly in the same numerical range while product fusion does not. For this reason, only the product fusion is considered for fusing with *ASMK* as the similarity scores produced by *ASMK* lie in a very different range from the scores generated based on the textual cues. The product fusion is also different from the sum fusion because the product fusion has a higher requirement than the sum fusion on the quality of both modalities to derive a decent final result. In general, the product fusion requires both modalities to be reasonably good.

3.5.2 *Experiments and Results*

This section experimentally evaluates the proposed multi-modal approach to logo retrieval. We quantify the added value of the proposed textual cues on top of the visual baselines. Moreover, we compare with several state-of-the-art text detection methods for the purpose of logo retrieval.

Table 10 summarizes the results. Adding the proposed textual cues 'Textual (full)' and 'Textual (gray-only)' always improves the visual baselines. The best performance, 62.7% in mAP, is achieved by combining the proposed textual cues (full) with the visual baseline (*ASMK*) using product fusion. Interestingly, fusing the textual cues from other text detection methods with the visual baselines using the product fusion does not improve the performance because the performance of the textual part is too modest in these cases. From the experiments, it can be concluded that the proposed textual cue extraction that focuses on high recall word detection is effective, resulting in a textual cue complementary to the visual cues for logo retrieval.

Analysis. Adding the textual cues improves the retrieval performance on 641 queries out of 960 ('Textual (full) + Visual (ASMK)'). Text is helpful when it is in standard fonts and orientations. Figure 19a shows 4 example queries where combining textual and visual cues improves the performance of visual-only. On the other hand, when text is not there or it is in exotic fonts or orientations, adding textual has a negative effect on the accuracy. Figure 19b shows 4 example queries where considering textual information decreases the performance of visual-only. For the query of *Ferrari*, considering textual information is not helpful because there is simply no text. The example of *Cocacola* is due to the exotic font style which makes it unrecognizable. For *Foster* and *Guinness*, the vertical text makes detection and recognition fail.

	mAP%
Textual-only (full)	**32.2**
Textual-only (gray-only)	28.4
Textual-only ([92])	12.3
Textual-only ([183])	13.2
Textual-only ([175])	12.7
Visual-only (BOW)	54.8
Visual-only (ASMK)	**58.4**
Textual (full) + Visual (BOW) [*sum fusion*]	59.4
Textual (gray-only) + Visual (BOW) [*sum fusion*]	57.8
Textual ([92]) + Visual (BOW) [*sum fusion*]	56.0
Textual ([183]) + Visual (BOW) [*sum fusion*]	56.2
Textual ([175]) + Visual (BOW) [*sum fusion*]	55.9
Textual (full) + Visual (BOW) [*product fusion*]	59.5
Textual (gray-only) + Visual (BOW) [*product fusion*]	56.9
Textual ([92]) + Visual (BOW) [*product fusion*]	36.2
Textual ([183]) + Visual (BOW) [*product fusion*]	34.5
Textual ([175]) + Visual (BOW) [*product fusion*]	30.8
Textual (full) + Visual (ASMK) [*product fusion*]	**62.7**
Textual (gray-only) + Visual (ASMK) [*product fusion*]	61.0
Textual ([92]) + Visual (ASMK) [*product fusion*]	41.5
Textual ([183]) + Visual (ASMK) [*product fusion*]	40.1
Textual ([175]) + Visual (ASMK) [*product fusion*]	36.5

Table 10: Logo retrieval performance on FlickrLogos-32 [141]. Adding the proposed textual cues always improves the retrieval performance. The proposed textual cues are more effective than the textual cues from other text detection methods due to the focus on high recall word detection.

(a) Improved cases

(b) Failure cases

Figure 19: (a) Example queries where adding textual cues improves the retrieval performance of visual-only. (b) Example queries where adding textual cues decreases the performance. The reasons are no text (Ferrari), exotic font style (Cocacola) and vertical text (Foster and Guinness).

3.6 WORD BOX PROPOSAL EVALUATION

Dataset. We evaluate the performance of our word box proposal method on the *SVT* dataset [169]. The dataset consists of 249 images which are downloaded from Google Street View of road-side scenes. The dataset has word-level box annotations.

Evaluation measures. The performance is measured in terms of recall, number of proposals and average maximum overlap (*AMO*) (See Table 11). We calculate the overlap between each groundtruth box and its best overlapping word box proposal. AMO is the average of these overlap values.

3.6.1 *Experiments and Results*

We conduct three experiments. First, we evaluate the effect of the color spaces and the character detection algorithms on the word detection performance. Second, we compare our method with state-of-the-art word box proposal methods [51, 69]. Third, we analyze the influence of ground-truth overlap threshold on word detection recall and word recognition accuracy.

Experiment I. The proposed method generates word box proposals using different color spaces and character detection algorithms. Word box proposals are generated for each color space independently and then combined. The same candidate regions may be detected for the different color spaces or character detection algorithms. To filter out these duplicate regions, non-maximum suppression is applied.

Table 11 shows that adding more color spaces improves the performance in terms of recall and AMO. When a single character detection algorithm is used, the recall values for MSER and text saliency are 85.47% and 90.88% respectively, whereas the recall is 96.14% when both algorithms are considered. Hence, the use of color spaces with different invariant properties, and complementary character detection algorithms results in a high recall.

Experiment II. We compare the performance of our word proposals with the state-of-the-art word proposal methods [51, 69]. [69] uses generic object proposal methods to generate preliminary word box proposals. However, the number of boxes is prohibitively large ($> 10^4$). Therefore, [69] filters out most of these boxes using a Random Forest text/non-text classifier. As their recognition step is based on the preciseness of the word boxes, a convolutional neural network regressor is learned to refine the coordinates of the remaining word boxes. [51] uses MSER with flexible parameters and a grouping strategy to generate word proposals. These proposals are also further scored by a weak classifier for word-likeliness. Table 11 shows that our method achieves a slightly higher recall than [51, 69] while requiring fewer boxes.

Experiment III. As is common practice in text detection, a candidate word-box is considered as a true positive if it overlaps more than 0.5 with the ground-truth word-box. However, a 0.5 overlap does not guarantee a correct word recognition. In particular, not all true positives are correctly recognized. We analyze the relation between the recognition accuracy and the overlap. The lexicon word with the maximum probability returned by [70] is considered as the word recognition result for each word proposal. Given the word proposals that pass the overlap threshold, the recognition accuracy is

	#proposals	recall(%)	AMO(%)
[This work] MSER+TSAL, I	338	84.23	70.40
[This work] MSER+TSAL, I+O_1, O_2+S	806	95.21	77.08
[This work] MSER+TSAL, I+O_1, O_2+S, H	968	96.14	**77.54**
[This work] MSER, I+O_1, O_2+H, S	568	85.47	70.90
[This work] TSAL, I+O_1, O_2+H, S	500	90.88	75.12
TextProposals [51]	17358	94.00	-
Jaderberg *et al.* [69] without (RF+CNN-reg)	$> 10^4$	**97.00**	77.00
Jaderberg *et al.* [69] without CNN-reg	900	94.80	-
Jaderberg *et al.* [69]	900	-	-

Table 11: Evaluation of the word box proposals on SVT dataset. MSER and TSAL are the MSER based and text saliency based character detection algorithms. I, O_1 ,O_2, H and S are the color models. The recall increases as more color invariant models are combined because of their complementary photometric invariant properties. Using both character detection algorithms results in a higher recall than using a single algorithm. RF and CNN-reg of [69] are the Random Forest classifier for non-text box filtering and the convolutional neural network regressor for box refinement. The values for [51, 69] are taken from the references, and empty blocks are not reported in the references. Different from [51, 69] the proposed method is fully unsupervised.

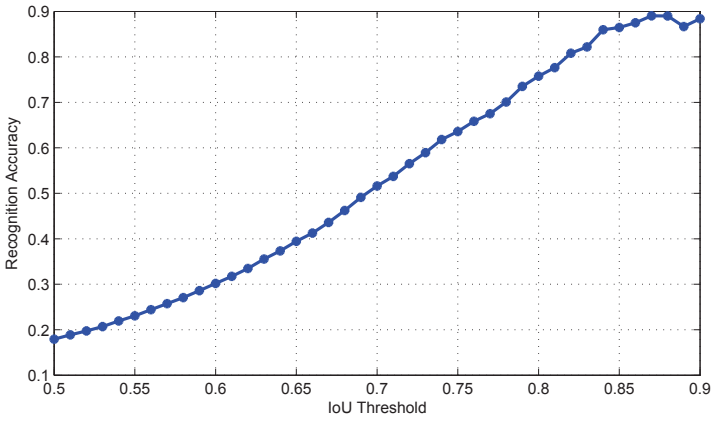


*Figure 20: The relation between the ground-truth overlap threshold (*i.e.*, the IoU threshold) and the word recognition accuracy, evaluated on SVT dataset [169]. Proposals with higher IoU values are better recognizable.*

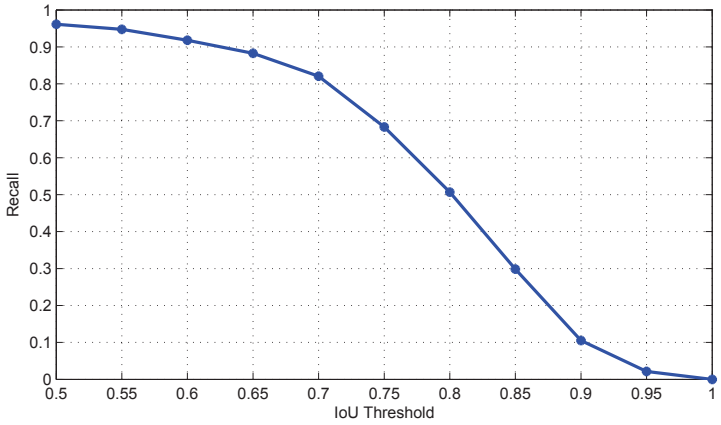


*Figure 21: The influence of the ground-truth overlap threshold (*i.e.*, the IoU threshold) on word detection recall, evaluated on SVT dataset [169]. Recall decreases as IoU threshold increases.*

computed as the percentage of correctly recognized proposals. Concretely, for a specific overlap threshold, *e.g.*, 0.7, we take all the word proposals that have at least 0.7 overlap with ground-truth, and compute how many of them are correctly recognized. The results are summarized in Figure 20. The results show that the candidate word-boxes (proposals), which have higher overlap with ground-truth, also have higher recognition accuracy. Therefore, not only higher recall but also higher *AMO* is important for accurate textual cue extraction. Further, we vary the ground-truth overlap threshold and evaluate the word detection recall. As expected, increasing the threshold has a negative effect on the recall, see Figure 21. However, the proposed method still performs well for a threshold of (> 0.75). For this threshold value, the recall and recognition accuracy is around 70%.

Efficiency. The matlab implementation of the proposed method (without optimization) takes 4s (on average) on a standard laptop to process one image from the *SVT* dataset.

3.7 CONCLUSION

We have demonstrated the effectiveness of textual cues for fine-grained (building) classification and logo retrieval. To capture textual information in images, a generic, efficient and fully unsupervised word box proposal approach which aims at high recall has been proposed. For fine-grained building classification, the proposed method outperforms the state-of-the-art [85] from 39.0% to 55.8% in mean average precision. It shows that encoding the textual cues at the word level is superior to using characters. To validate the influence of recall, precision and f-score changes on fine-grained classification, we have annotated a large set of 27601 word boxes. Furthermore, the work explores textual cues for logo retrieval. Combining the textual and visual cues improves the retrieval performance to 62.7% from 58.4% of visual-only. Moreover, we show that high recall in word detection is more relevant than high f-score for fine-grained classification and logo retrieval. The proposed unsupervised word box proposal method achieves state-of-the-art recall for word detection on *SVT* with a limited number of word box proposals (< 1000).

4

ATTRIBUTES AND CATEGORIES FOR GENERIC INSTANCE SEARCH FROM ONE EXAMPLE

4.1 INTRODUCTION

[1]In instance search, the objective is to retrieve all images of a specific object given a few query examples of that object [7, 73, 125, 135, 197]. We consider the challenging case of only 1 query image and admitting large differences in the imaging angle and other imaging conditions between the query image and the target images. A very hard case is a query specified in frontal view while the relevant images in the search set show a view from the back which has never been seen before. Humans solve the search task by employing two types of general knowledge. First, when the query instance is a certain class, say a female, answers should be restricted to be from the same class. And, queries in the frontal view showing one attribute, say brown hair, will limit answers to show the same attribute, even when the viewpoint is from the back. In this chapter, we exploit these two types of knowledge to handle a wide variety of viewpoints, illumination and other conditions for instance search.

In instance search, excellent results have been achieved by restricting the search to buildings [7, 8, 46, 133]. Searching buildings can be used in location recognition and 3D reconstruction. Another set of good results has been achieved in searching for logos [80, 139, 156] for the estimation of brand exposure. And, [176] searches for book and magazine covers. All these cases of instance search show good results for near-planar, and one-sided objects which are recorded under a limited range of imaging angles. In this work, we aim for broader classes of query instances. We aim to perform generic instance search from 1 example. *Generic* implies we consider arbitrary objects, and not just one-sided objects. And, *generic* implies we aim to use one approach not specially designed for a certain kind of instances, such as RANSAC-based geometric verification for rigid and highly textured objects. In our case, instances can be buildings and logos, but also shoes, clothes and other objects. In this work, we illustrate on a diverse set of instances, including shoe, car, building and person. In this work, we treat person re-identification [52] as a special case of generic instance search, and address the problem using the same method as for other kinds of instances.

The challenge in instance search is to represent the query image invariant to the (unknown) appearance variations of the query while maintaining a sufficiently rich representation to permit distinction from other, similar instances. To solve this, most existing approaches in instance search match the appearance of local spots [11, 99] in

1 A preliminary version of the chapter is published in *IEEE Conference on Computer Vision and Pattern Recognition, 2015* [157].

the potential target to the query [73, 77, 133, 156, 158]. The quality of match in these approaches between two images is the sum of similarities over all local descriptor pairs. The difference between the cited approaches lies in the way local descriptors are encoded and in the computation of the similarity. Good performance has been achieved by this paradigm on buildings, logos and scenes from a distance. However, when searching for an arbitrary object with a wider range of viewpoint variability, more sides, and possibly having self-occlusion and non-rigid deformation, these methods are likely to fail as local descriptor matching becomes unreliable in these cases [109].

In this work we propose to use automatically learned attributes [40, 89] to address generic instance search. Attributes, as higher level abstractions of visual properties, have been shown advantageous in classification when training examples are insufficiently covering the variations in the original feature space [4, 40, 184], surely present in the one-example challenging case. By employing attributes, we aim to be robust against intra-instance appearance variations. Further, we optimize the attributes such that they are meanwhile discriminative among different instances. Concretely, in this work, we learn a set of category-specific non-semantic attributes that are optimized to recognize different instances of a certain category, *e.g.*, shoes. With the learned attributes, an instance can be represented as a specific combination of the attributes, and instance search boils down to finding the most similar combinations of attributes.

In order to address the possible confusion of the query with instances from other categories, we further propose to supplement the learned category-specific attributes with category-level information. The category-level information are incorporated to reduce the search space by filtering instances of other categories. It is advantageous when there is only 1 query image, to use slightly more user provided information. In addition to the interactive specification of the object region in the query image, we require the specification of the category the query instance belongs to.

4.2 RELATED WORK

Most approaches in instance search rely on gathering matches of local image descriptors [73, 77, 133, 151, 156, 158], where the differences reside in the way the local descriptors are encoded and the matching score of two descriptors is evaluated. Bag-of-words (BoW) [133, 151] encodes a local descriptor by the index of the nearest visual word. Hamming embedding [73] improves upon BoW by adding an extra binary code to better describe the position of the local descriptor in space. The matching score of a pair of descriptors is 1 if they are encoded to the same word and the Hamming distance between binary signatures is smaller than a certain threshold. VLAD [75] and Fisher vector [131] improve over BoW by representing the local descriptor with an extra residual vector, obtained by subtracting the mean of the visual word or the Gaussian component respectively. In VLAD and Fisher vector, the score of two descriptors is the dot product of the residuals when they are encoded to the same word, and 0 otherwise. [156, 158] improve VLAD and Fisher vector by replacing the dot product by a thresholded polynomial similarity and an exponential similarity respectively to give disproportionally more credits to closer descriptor pairs. [77] encodes a local descriptor by only considering the directions to the visual word centers, not the magnitudes, outperforming Fisher vector on instance search.

With these methods, good performance has been achieved on buildings, logos, and scenes from a distance. These instances can be conceived as near-planar and one-sided. For buildings, logos, and scenes from a distance the variation in the viewing angle is limited to a quadrant of 90 degrees at most out of the full 360 circle. For limited variations in viewpoint, matches of local descriptors can be reliably established between the query and a relevant example. In this work, we consider generic instance search, where the instance can be an arbitrary object with a wider range of viewpoint variability and more sides. We evaluate existing methods for approximately one-sided instance search on this problem of generic instance search.

Attributes [40, 43, 89] have received much attention recently. They are used to represent common visual properties of different objects. Attribute representation has been used for image classification [4, 40, 184]. Attributes have been shown to be advantageous when the training examples are insufficiently covering the appearance variations in the original feature space [40, 184]. Inspired by this, we propose to use attribute representation to address generic instance search, where there is only 1 example available and there still exists a wide range of appearance variations.

Attributes have been used for image retrieval [86, 137, 149, 184, 185]. In [86, 149, 185], the query is defined by textual attributes instead of images and the goal is to return images exhibiting query attributes. In the references, the query attributes need to be semantically meaningful such that the query can be specified by text. In this work, we address instance search given one query image, which is a different task as the correct answers have to exhibit the same instance (not just the same attributes), and we use automatically learned attributes which as a consequence may or may not be semantic. [137, 184] consider non-semantic attributes for category retrieval, while this work addresses generic instance retrieval.

The use of category-level information to improve instance search has been explored in [33, 53, 191]. [53] uses category labels to learn a projection to map the original feature to a lower-dimensional space such that the lower-dimensional feature incorporates certain category-level information. In this work, instead of learning a feature mapping, we augment the original representation with additional features to capture the category-level information. In [33], Fisher vector representation is expanded with the concept classifier output vector of the 2659 concepts from Large Scale Concept Ontology for Multimedia (LSCOM) [116]. In [191], a 1000-dimensional concept representation [1] is utilized to refine the inverted index on the basis of semantic consistency between images. Both [191] and [33] combine category-level information with low-level representation. In this work, we consider the combination of category-level information with category-specific attributes rather than a low-level representation. We argue this is a more principled combination as the category-level information by definition makes category-level distinction and the category-specific attributes are optimized for within-category discrimination.

Person re-identification is a well-studied topic [13, 52, 165], where the work mainly branches into two aspects, feature designing [55, 102, 182] and metric learning [22, 62, 126]. Among the vast amount of work in literature, most related to this work are papers focusing on building a good representation [3, 12, 55, 94, 102, 148, 182, 193, 194]. [55] uses AdaBoost to select features from an ensemble of localized features. [102] encodes the local descriptors using Fisher vector. [12] exploits the symmetry and asymmetry

properties of human body to capture the cues on the human body only, pruning out background clutters. [193] learns human saliency in an unsupervised manner to find reliable and discriminative patches. [194] proposes to learn mid-level patch filters that are viewpoint invariant and discriminative in differentiating identities. [182] employs a salient color names based representation. [94] records the maximal local occurrence of a pattern to achieve invariance to viewpoint changes. [3] simultaneously learns features and a similarity metric using deep learning. [148] proposes to learn semantic fashion-related attribute representation from auxiliary datasets and adapt the representation to target datasets. In this work, we propose to learn a non-semantic attribute representation without using auxiliary data to handle the large appearance variations caused by viewpoint differences, illumination variations, deformation and others. Furthermore, in this work, inspired by [195], we treat person re-identification as a special case of the generic instance search problem, where the instance of interest is now a specific person, and address the problem using the same attribute-based approach as for other types of instance search, *e.g.*, shoes and buildings.

4.2.1 *Contributions*

Our work makes the following contributions. We propose to pursue generic instance search from 1 example where the instance can be an arbitrary 3D-object recorded from a wide range of imaging angles. We argue that this problem is harder than the approximately one-sided instance search of buildings [133], logos [80] and remote scenes [73]. We evaluate state-of-the-art methods on this problem. We observe what works best for buildings loses its generality for shoes and reversely what works worse for buildings may work well for shoes.

Second, we propose to use automatically learned category-specific attributes to handle the wide range of appearance variations in generic instance search. Here we assume we know the category of the query instance which provides critical knowledge when there is only 1 query image. Information of the query category can be given through interactive user interface or automatic image categorization (*e.g.*, shoe, dress, *etc.*). On the problem of searching among instances from the same category as the query, our category-specific attributes outperform existing instance search methods by a large margin when large appearance variations exist.

Third, inspired by [195], we treat person re-identification as a special case of generic instance search, where the instance of interest is a specific person. On the popular VIPeR dataset [54], we reach state-of-the-art performance with the same attribute-based method.

As our fourth contribution, we extend our method to search instances without restricting to the known category. We propose to augment the category-specific attributes with category-level information which is carried by high-level deep learning features learned from large-scale image categorization and the category-level classification scores. We show that combining category-level information with category-specific attributes achieves superior performance to combining category information with low-level features such as Fisher vector.

4.3 THE DIFFICULTY OF GENERIC INSTANCE SEARCH

The first question we raise in this work is how the state-of-the-art methods perform on generic instance search from 1 example where the query instance can be an arbitrary object. *Can we search for other objects like shoes using the same method that has been shown promising for buildings?* To that end, we evaluate several existing instance search algorithms on both buildings and shoes.

We evaluate the following methods. ***ExpVLAD*:** [156] introduces locality at two levels to improve instance search from one example. The method considers locality in the picture by evaluating multiple candidate locations in each of the database images. It also considers locality in the feature space by efficiently employing a large visual vocabulary for VLAD and Fisher vector and by an exponential similarity function to give disproportionally high scores on close local descriptor pairs. The locality in the picture was shown effective when searching for instances covering only a part of the image. And the the locality in the feature space was shown useful on all the datasets considered in the reference. ***Triemb*:** [77] proposes triangulation embedding and democratic aggregation. The triangulation embedding encodes a local descriptor with respect to the visual word centers using only directions, not magnitudes. As shown in the paper, the triangulation embedding outperforms Fisher vector [144]. The democratic aggregation assigns a weight to each local descriptor extracted from an image to ensure all descriptors contribute equally to the self-similarity of the image. This aggregation scheme was shown better than the sum aggregation. ***Fisher*:** We also consider Fisher vector as it has been widely applied in instance search and object categorization where good performance has been reported [76, 144]. ***Deep-FC*:** It has been shown recently that the activations in the fully connected layers of a deep convolutional neural network (CNN) [87] serve as good features for several computer vision tasks [10,48,138]. ***VLAD-Conv*:** Very recently, [120] proposes to apply VLAD encoding [76] on the output of the convolutional layers of CNN for instance search.

Datasets. Oxford buildings dataset [133], often referred to as *Oxford5k*, contains 5062 images downloaded from Flickr. 55 queries of Oxford landmarks are defined, each by a query example. *Oxford5k* is one of the most popular datasets for instance search, which has been used by many works to evaluate their approaches. Figure 22a shows examples of two buildings from the dataset.

As a second dataset, we collect a set of shoe images from Amazon[2]. It consists of 1000 different shoes and in total 6624 images. Each shoe is recorded from multiple imaging angles including views from front, back, top, bottom, side and some others. One image of a shoe is considered as the query and the goal is to retrieve all the other images of the same shoe. Although these images are with clean background as often seen on shopping websites, this is a challenging dataset mainly due to the presence of considerably large viewpoint variations and self-occlusion. We refer to this dataset as *CleanShoes*. Figure 22b shows examples of three shoes from *CleanShoes*. There is a shoe dataset available, proposed by [14]. However, this dataset is not suited for instance search as it does not contain multiple images for one shoe. [147] also considers shoe

2 The properties are with the respective owners. The images are shown here only for scientific purposes.

Figure 22: (a) Examples of two buildings from Oxford5k, and (b) Examples of three shoes from our CleanShoes dataset. There exists a much wider range of viewpoint variability in the shoe images.

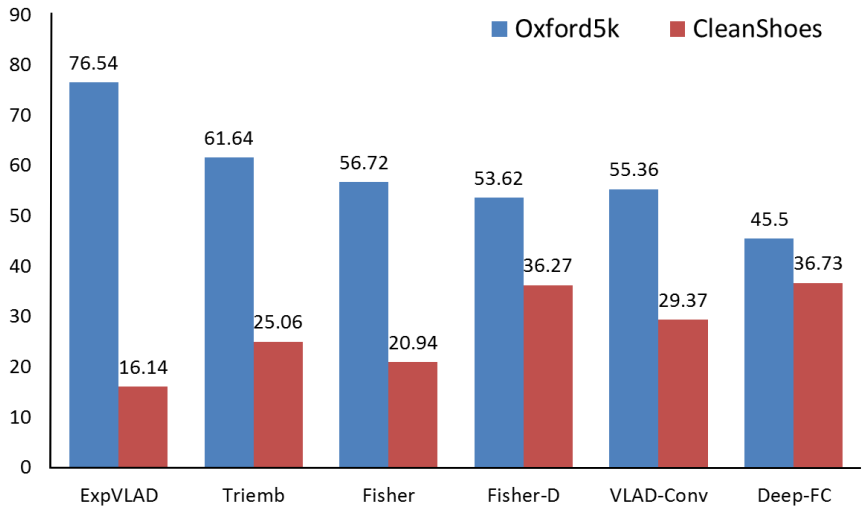


Figure 23: Performance of various state-of-the-art methods for instance search measured in mean average precision%: ExpVLAD [156], Triemb [77], Fisher [76], VLAD-Conv [120] and Deep-FC [87]. For Fisher vector, we consider two versions. Fisher denotes the version with interest points and SIFT descriptors, and Fisher-D uses densely sampled RGB-SIFT descriptors. ExpVLAD achieves better performance than others on Oxford5k, but gives lowest result on CleanShoes. On the other hand, Deep-FC obtains best performance on CleanShoes, but has lower result than others on Oxford5k.

images, but the images are well aligned, whereas the images in *CleanShoes* provide a much wider range of viewpoint variations.

Implementation details. For *ExpVLAD*, *Triemb* and *Fisher*, we use the Hessian-Affine detector [128] to extract interest points. The SIFT descriptors are turned into RootSIFT [8]. The full 128D descriptors are used for *ExpVLAD* and *Triemb*, following [77, 156], while for *Fisher*, the local descriptor is reduced to 64D using PCA, as the PCA reduction has been shown important for Fisher vector [76, 144]. The vocabulary size is 20k, 64 and 256 for *ExpVLAD*, *Triemb* and *Fisher* respectively, following the corresponding references [76, 77, 156]. We additionally run a version of Fisher vector with densely sampled RGB-SIFT descriptors [163] and a vocabulary of 256 components, denoted by *Fisher-D*. For *Deep-FC*, we use an in-house implementation of the AlexNet [87] trained on ImageNet categories, and take the ℓ_2 normalized output of the second fully connected layer as the image representation. For *VLAD-Conv*, we apply VLAD encoding with a vocabulary of 100 centers on the conv5_1 responses of the VGGNet [150], following [120]. For *Triemb*, *Fisher*, *Fisher-D* and *VLAD-Conv*, power normalization [132] and ℓ_2 normalization are applied.

Results and discussions. Figure 23 summarizes the results on *Oxford5k* and *CleanShoes*. *ExpVLAD* adopts a large vocabulary with 20k visual words and the exponential similarity function. As a result, only close local descriptor pairs in the feature space matter in measuring the similarity of two examples. This results in better performance than others on *Oxford5k* where close and relevant local descriptor pairs do exist. However, on the shoe images where close and true matches of local descriptors are rarely present due to the large appearance variations, *ExpVLAD* achieves lowest performance. Both

Triemb and *Fisher* obtain quite good results on buildings but the results on shoes are low. This is again caused by the fact that local descriptor matching is not reliable on the shoe images where large viewing angle differences are present. *Triemb* outperforms *Fisher*, consistent with the observations in [77]. In this work, we do not consider the RN normalization [77] because it requires extra training data to learn the projection matrix and it does not affect the conclusion we make here. *Fisher-D* works better than *Fisher* on *CleanShoes* by using color information and densely sampled points. Color is a useful cue for discriminating different shoes, and dense sampling is better than interest point detector on shoes which do not have rich textural patterns. However, *Fisher-D* does not improve over *Fisher* on *Oxford5k*. *VLAD-Conv* is in the middle on both sets. *Deep-FC* has lowest performance on buildings, but outperforms others on shoes.

Overall, the performance on shoes is much lower than on the buildings. More interestingly, *ExpVLAD* achieves better performance than others on *Oxford5k*, but gives lowest result on *CleanShoes*. On the other hand, *Deep-FC* obtains best performance on *CleanShoes*, but has lower result than others on *Oxford5k*. We conclude that none of the existing methods work well on both buildings, as an example of 2D one-sided instance search, and shoes, as an example of 3D full-view instance search.

4.4 ATTRIBUTES FOR GENERIC INSTANCE SEARCH

Attributes, as a higher level abstraction of visual properties, have been shown advantageous in categorization when the training examples are insufficiently covering the appearance variations in the original feature space [4, 40, 184]. In our problem, there is only 1 example available and there still exists a wide range of appearance variations. *Can we employ attributes to address generic instance search?*

4.4.1 *Method*

In the literature, two types of attributes have been studied, manually defined attributes with names [4, 89] and automatically learned unnameable attributes [146, 184]. Obtaining manually defined attributes requires a considerable amount of human efforts and sometimes domain expertise, making it hard to scale up to a large number of attributes. Moreover, the manually picked attributes are not necessarily machine-detectable, and not guaranteed to be useful for the task under consideration [184]. On the other hand, learned attributes do not need human annotation and have the capacity to be optimized for the task [146, 184]. For some tasks, like zero-shot learning [4] and image retrieval by textual query [149], it is necessary to use human understandable attributes with names. However, in instance search given 1 image query, having attributes with names is not really necessary. In this work, we use automatically learned attributes. Specifically, we focus on searching among instances known to be of the same category in this section using automatically learned category-specific attributes.

Provided with a set of training instances from a certain category, we aim to learn a list of category-specific attributes and use them to perform instance search on new (unseen) instances from the same category. Concretely, given m training images of n objects ($m > n$ as each object has one or multiple examples), the goal is to learn k

attribute detectors. In the search phase, the query image and the dataset images are represented by k-dimensional attribute detection scores, and the search is performed by comparing the distances in the k-dimensional feature space.

Analogous to the class-attribute mapping in attribute-based categorization [4, 40, 89], an instance-attribute mapping $A \in \mathbb{R}^{n \times k}$ is designed automatically. The challenge is how to obtain a useful A. As the goal in instance search is to differentiate different instances, the attributes should be able to make distinctions among the training instances. On the other hand, as the attributes will be used later for instance search on new, unseen instances, the learned attributes need to be able to generalize on unseen instances. To that end, visually similar training instances are encouraged to share attributes. Attributes specific to one training instance are less likely to generalize on unknown instances than those shared by several training instances. And sharing needs to be restricted only among visually similar training instances as the latent common visual patterns among visually dissimilar instances are less likely to be present and detected on new instances even if they can be learned provided with a high dimensional feature space. Besides, to make the best out of the k attributes, it is desirable to have low redundancy among the attributes. Formally, taking the above considerations into account, we design A by

$$\underset{A}{\text{maximize}} \quad f_1(A) + \lambda f_2(A) + \gamma f_3(A), \tag{4.1}$$

where $f_1(A)$, $f_2(A)$ and $f_3(A)$ are defined as follows:

$$\begin{aligned} f_1(A) &= \sum_{i,j}^{n} \|A_{i\cdot} - A_{j\cdot}\|_2^2, \\ f_2(A) &= -\sum_{i,j}^{n} S_{ij} \|A_{i\cdot} - A_{j\cdot}\|_2^2, \\ f_3(A) &= -\|A^T A - I\|_F^2. \end{aligned} \tag{4.2}$$

$A_{i\cdot}$, the i-th row of A, is the attribute representation of the i-th instance. $f_1(A)$ ensures instance separability. S in $f_2(A)$ is the visual proximity matrix, where S_{ij} represents visual similarity between instance i and instance j, measured *a priori* in certain visual feature space. The similarity between two training instances is computed as the average similarity between the images of the two instances. $f_2(A)$ encourages similar attribute representations between visually similar instances, inducing shareable attributes. $f_3(A)$ penalizes redundancy between attributes. λ and γ are two parameters of the objective. Larger λ encourages more attribute sharing among visually similar instances and larger γ penalizes more on the redundancy in the learned attributes. This formulation was originally proposed in [184] for category recognition. Following [184], the optimization problem is solved incrementally by obtaining one column of A, *i.e.*, one attribute at each step. Next we briefly describe the optimization procedure.

The objective (Equation 4.1) can be rewritten as

$$\underset{A}{\text{maximize}} \quad Tr(A^T P A) - \gamma \|A^T A - I\|_F^2, \tag{4.3}$$

where $P = Q - \lambda L$. Q is an $n \times n$ matrix with diagonal elements being $n-1$ and off-diagonal elements being -1. L is the Laplacian of S [167]. Initializing A as an empty matrix, A can be learned incrementally, one column at one step, by

$$\underset{\mathbf{a}}{\text{maximize}} \quad \mathbf{a}^T R \mathbf{a} \quad s.t. \quad \mathbf{a}^T \mathbf{a} = 1, \tag{4.4}$$

where $R = P - 2\gamma A A^T$. The optimal $\mathbf{a}$ is the eigenvector of R with the largest eigenvalue. A is updated by $A = [A, \mathbf{a}]$ at every step. In this work, each attribute, *i.e.*, $\mathbf{a}$, is binarized during the optimization.

Attribute detectors. Once the instance-attribute mapping A has been obtained, the next step is to learn the attribute detectors. In this work, the attribute detectors are formulated as linear SVM classifiers. To train the j-th attribute detector, images of the training instances with $A_{ij} > 0$ are used as positive examples and the rest images are negative examples[3].

Attribute representation. Given a new image, the attribute representation is generated by applying all the learned attribute detectors and concatenating the SVM classification scores. The attribute representation is discriminative in distinguishing different instances as it is optimized to be so when designing A. The attribute representation is invariant to the appearance variations of an instance as the invariance is built in the attribute detectors which take all the images of one instance as either all positive or all negative during learning.

4.4.2 *Datasets*

Evaluation sets. The category-specific attributes as learned are evaluated on shoes, cars and buildings. For shoes, the dataset *CleanShoes* described Section 4.3 is used. For cars, we collect 1110 images of 270 cars from eBay, denoted by *Cars*. Figure 24 shows some images of two cars[4]. For buildings, a dataset is composed by gathering all 567 images of the 55 Oxford landmarks from *Oxford5k*, denoted by *OxfordPure*. We reuse the 55 queries defined in *Oxford5k*.

Training sets. To learn shoe-specific attributes, we collect 2100 images of 300 shoes from Amazon. To train car-specific attributes, we collect 1520 images of 300 cars from eBay. To learn building-specific attributes, we use a subset of the large building dataset introduced in [10]. We randomly pick 30 images per class and select automatically the 300 classes that are most relevant to *OxfordPure* according to the visual similarity. We end up with in total 8756 images as some URLs are broken and some classes have less than 30 examples. For all shoes, cars and buildings, the instances in the evaluation sets are not present in the training sets.

3 We have also tried designing the instance-attribute mapping A with continuous values and learning a regressor for each attribute. However, this is not better in terms of instance search performance.

4 The properties are with the respective owners. The images are shown here only for scientific purposes.

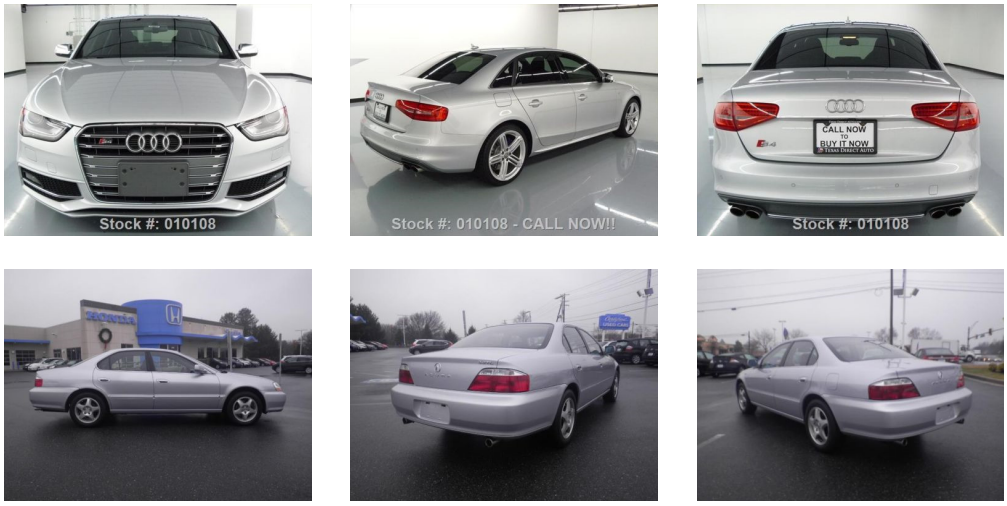


Figure 24: Examples of two cars from the dataset Cars.

4.4.3 *Empirical parameter study*

We empirically investigate the effect of the two parameters of the learning algorithm (λ and γ in Equation 4.1) on the search performance. We learn different sets of category-specific attributes with different λ and γ values and evaluate the instance search performance. The study is conducted on the shoe dataset.

Fisher vector [144] with densely sampled RGB-SIFT [163] is used as the underlying representation to compute the visual proximity matrix S in Equation 4.2 and learn the attribute detectors. S is built as a mutual 60-NN adjacent matrix throughout the work.

First, we study the effect of λ by fixing γ. An extreme case is setting λ to be 0, which means no attribute sharing among training instances. As shown in Figure 25 (left), when λ is 0, the search performance is much worse than when λ is from 1 to 5, especially when the number of attributes is low. When there is no sharing induced, the learned attributes on the training instances cannot generalize well on the new instances in the search set. As long as sharing is enabled, the search performance is robust to the value of λ.

Second, we study the effect of γ by fixing λ. As can be seen from Figure 25 (right), when γ is small (0.01), which means large redundancy in the learned attributes, the search performance is very low, but stabilizes once γ is large enough.

The above study shows the importance of enforcing attribute sharing and low redundancy during learning as well as the robustness of the learning algorithm against the values of λ and γ, in terms of the instance search performance. In the rest of work, we set λ and γ to be 2 and 7 respectively to be consistent with the earlier version of the work [157].

4.4.4 *Comparison with manual attributes*

We compare the learned attributes with manually defined attributes on shoe search. For manually defined attributes, we use the list of attributes proposed by [68]. We manually annotate the same 2100 training images. In the reference, 42 attributes are defined.

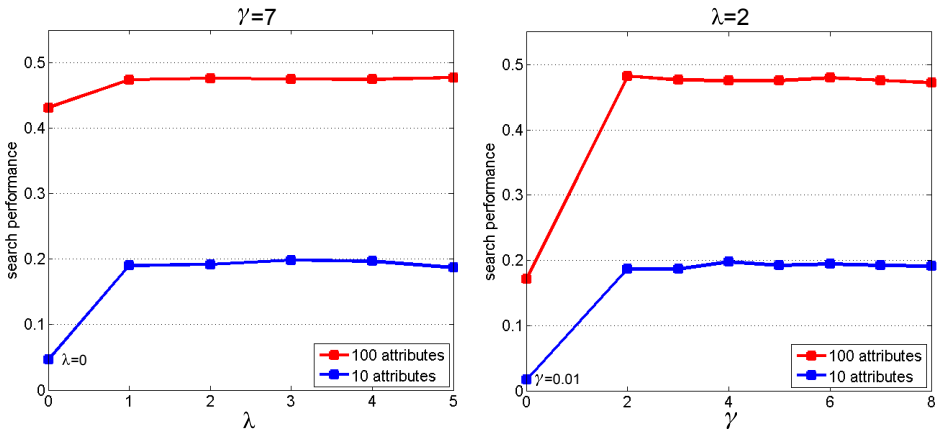


Figure 25: The impact of the parameters of the attribute learning algorithm (λ and γ) on the search performance, measured in mean average precision. The experiments are conducted on CleanShoes. When there is no attribute sharing enforced between instances ($\lambda = 0$) or there is large redundancy in the learned attributes ($\gamma = 0.01$), the search performance is low. It indicates the importance of enforcing attribute sharing and low redundancy. The observation on the impact of λ holds when fixing γ to other values (The same holds for the observation on γ.).

	number	CleanShoes
Manual attributes	*40*	18.99
Learned attributes	*40*	39.44
Learned attributes	*1000*	56.57

Table 12: Comparison of learned attributes and manually defined attributes on shoe search. The performance is measured in mean average precision%.

However, we merge *super-high* and *high* of "upper" and "heel height" because it is hard to annotate *super-high* and *high* as two different attributes. This results in 40 attributes.

Again, Fisher vector is used as the underlying representation to learn attribute detectors. As shown in Table 12, with the same number of attributes, the automatically learned attributes work significantly better than the manual attributes. Moreover, automatically learned attributes are easily scalable, improving performance further. Figure 26 shows four automatically learned attributes. Although the attributes have no explicit names, they do capture common visual properties between shoes.

4.4.5 *Empirical study of underlying feature representation*

In theory, attributes can be learned from any underlying feature representation. In this section, we empirically evaluate the impact of various underlying features for attribute learning on the instance search performance. We consider 5 different feature representations investigated in Section 4.3, *i.e.*, *Triemb*, *Fisher*, *Fisher-D*, *VLAD-Conv* and *Deep-FC*. *ExpVLAD* is not included as it does explicitly form a vector representation

Figure 26: Four automatically designed attributes. Each row is one attribute and the shoes are the ones that have high response for that attribute. Although the automatically learned attributes have no semantic names, apparently they capture sharing patterns among shoes. The first attribute represents high boots. The second describes the high heels. The third is probably about colorfulness. The last one is about openness. The first two are also found in the manually defined attributes while the other two are novel ones discovered automatically.

	dim	CleanShoes	Cars	OxfordPure
ExpVLAD [156]	—	16.14	23.70	**87.01**
Triemb [77]	*8064*	25.06	18.56	75.33
Fisher [76]	*16384*	20.94	18.37	70.81
Fisher-D [76, 163]	*40960*	36.27	20.89	67.41
VLAD-Conv [120]	*51200*	29.37	27.27	69.05
Deep-FC [87]	*4096*	36.73	22.36	59.48
Attributes(Triemb)	*1000*	19.83	28.15	71.58
Attributes(Fisher)	*1000*	17.67	31.21	69.33
Attributes(Fisher-D)	*1000*	56.57	51.11	77.36
Attributes(VLAD-Conv)	*1000*	**63.19**	**63.99**	82.86
Attributes(Deep-FC)	*1000*	57.11	38.07	69.51

Table 13: Performance in mean average precision% of existing methods (top part of the table) and the attributes learned from single underlying features (bottom part). The attributes learned from Fisher-D, VLAD-Conv or Deep-FC outperform existing methods significantly on shoes and cars, and achieve comparable performance on buildings. Attributes learned from the underlying features that capture densely the visual cues (Fisher-D, VLAD-Conv and Deep-FC) are better than those learned from the underlying features based on sparse interest points (Triemb and Fisher).

to facilitate the learning. The proximity matrix S is measured in the same feature space as used for learning attributes.

First, we evaluate the attributes learned from single underlying features and compare with existing approaches. The results are summarized in Table 13. We observe that when the underlying feature representation for attribute learning is based on sparse interest points, including *Triemb* and *Fisher*, the learned attribute representation does not always improve the search performance over the original representation. However, when the underlying feature representation is based on densely extracted visual cues, including *Fisher-D*, *VLAD-Conv* and *Deep-FC*, the attribute representation always outperforms the underlying feature representation by a large margin. This indicates that the mapping from the original feature representation to the attribute representation is selective. It selects the useful information which is discriminative among different instances and invariant to the variations of the same instance, while discarding other disturbing information. We argue that a large amount of useful information has already been filtered by the internal selection step of the interest point detector and therefore attribute representation learned on interest point based features does not help much. The attribute representation learned using *VLAD-Conv* achieves better performance than those learned from other underlying representations. On the shoe and car datasets, the learned attribute representation significantly outperforms existing approaches. Attributes are superior in addressing the large appearance variations caused by the large imaging angle difference present in the shoe and car images, even though the attributes are learned from other instances. The attribute representation also works well on the buildings. In addition, attribute representation has a much lower dimensionality than other representations.

	dim	CleanShoes	Cars	OxfordPure
Fisher-D [76, 163],VLAD-Conv [120]	*92160*	35.64	26.18	71.29
Fisher-D [76, 163],Deep-FC [87]	*45056*	41.55	22.65	69.41
VLAD-Conv [120],Deep-FC [87]	*55296*	36.25	26.25	69.84
Fisher-D [76, 163],VLAD-Conv [120],Deep-FC [87]	*96256*	39.04	25.58	71.42
Attributes(Fisher-D,VLAD-Conv)	*1000*	63.97	69.19	83.22
Attributes(Fisher-D,Deep-FC)	*1000*	67.45	59.96	78.66
Attributes(VLAD-Conv,Deep-FC)	*1000*	67.06	69.02	**83.75**
Attributes(Fisher-D,VLAD-Conv,Deep-FC)	*1000*	**67.87**	**71.74**	83.06

Table 14: Performance in mean average precision% of combining multiple existing representations (top part of the table) and the attributes learned from multiple underlying features (bottom part). The learned attribute representation significantly outperforms the underlying representation. Comparison with Table 13 shows that the attribute representation learned from multiple underlying features outperforms those learned on single features. Interestingly, combining the same underlying features and directly using them for instance search without attributes does not necessarily improve over individual features.

Second, in Table 14, we investigate the effects of using multiple underlying features for attribute learning. Again, the attribute representation outperforms the underlying feature representation significantly. Comparing Table 14 and Table 13, it is clear that the attribute representation learned on the combination of multiple underlying features outperforms those learned on single features. This demonstrates the advantage of using multiple underlying feature representations, which as a whole can better capture the various types of visual properties than a single representation. Interestingly, combining the same underlying features and directly using them for instance search without attributes does not necessarily improve over individual features, which confirms again the advantage of attributes. The attribute representation learned on the combination of *Fisher-D*, *VLAD-Conv* and *Deep-FC* achieves best performance on shoes and cars, and close to best performance on buildings, improving the results reported in the earlier version of the work [157] from 56.57% to 67.87% on *CleanShoes*, from 51.11% to 71.74% on *Cars*, and from 77.36% to 83.06% on *OxfordPure* in mean average precision.

4.5 PERSON RE-IDENTIFICATION AS INSTANCE SEARCH

Person re-identification is the problem of identifying the images in a database which depict the same person as in the probe image. The probe image and the relevant images in the database are usually captured by different cameras with different recording settings, causing large viewpoint and illumination variations. Besides, a person might have different poses in different recordings and might be partially occluded. All these result in large intra-person variations, making person re-identification a challenging problem. In this work, we treat person re-identification as a specific person search problem, and address the problem using the attribute-based method presented in Section 4.4.

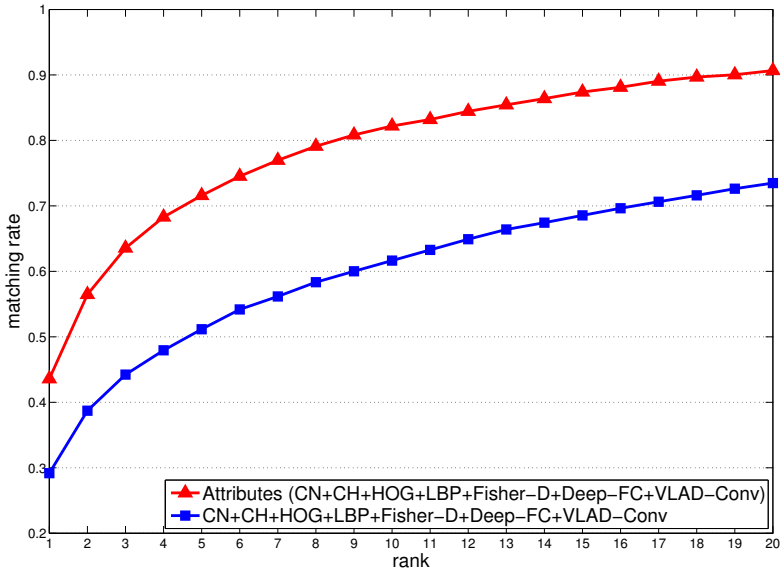


Figure 27: Performance on VIReR dataset [54], measured in matching rate on top ranked images. The learned attribute representation significantly outperforms the original underlying representation.

Dataset and evaluation protocol. We use the VIPeR dataset [54]. It has been widely used for benchmark evaluation. It contains 632 pedestrians, each recorded by two cameras. One view is considered as the probe image and the goal is to identify the other view of the same person. The 632 pairs are randomly divided into two halves, one for training and one for testing. The performance is evaluated using the Cumulative Match Characteristic (CMC) curve [54] which estimates the expectation of finding the correct answer in the top k results. The experiment is repeated 10 times to report an average performance[5].

Implementation details. 1000 attributes detectors are learned using the training split. To learn the attributes, we employ multiple underlying features. We use the bag-of-word histograms on local color histograms (*CH*), local color naming descriptors (*CN*), local HOG (*HOG*) and local LBP descriptor (*LBP*), provided by [195][6]. Besides, we employ *Deep-FC*, *Fisher-D* and *VLAD-Conv*. Vocabularies with 16 components and 8 centers are used for *Fisher-D* and *VLAD-Conv* respectively. The visual proximity matrix S in equation 4.2 is built as a mutual 60-NN adjacent matrix, the same as in previous sections.

Results. As shown in Figure 27, the learned attribute representation significantly outperforms the original underlying representation. The learned attributes can handle well the large appearance variations. Table 15 summarizes the comparison with the state-of-the-art. Although the proposed attribute-based method is not specially designed for person re-identification, it achieves good performance, on par with the state-of-the-art.

5 We use the 10 divisions provided by [195]

6 `http://www.liangzheng.com.cn/Project/project_fusion.html`

	rank=1	rank=5	rank=10	rank=20
Zheng *et al.* [195]	30.2	51.6	62.4	73.8
Ahmed *et al.* [3]	34.8	—	—	—
Chen *et al.* [22]	36.8	70.4	**83.7**	91.7
Shi *et al.* [148]	31.1	68.6	82.8	**94.9**
Liao *et al.* [94]	40.0	—	80.5	91.1
Paisitkriangkrai *et al.* [126]	**45.9**	—	—	—
Ours	43.6	**71.6**	82.2	90.7

Table 15: Comparison with state-of-the-art on VIPeR dataset [54] by correct matching rates(%). Although not being specialized for person, our method keeps up with the state-of-the-art for all ranks.

4.6 CATEGORIES AND ATTRIBUTES FOR GENERIC INSTANCE SEARCH

In this section, we consider searching for an instance from a dataset which contains instances from various categories. As the category-specific attributes are optimized to make distinctions among instances of the same category, they might not be able to distinguish the instance of interest from the instances of other categories. In order to address the possible confusion of the query instance with instances from other categories, we propose to use the category-level information also.

Ideally one could first categorize all the images in the database and then search using category-specific attributes among the images from the same category as the query. However, as errors made in categorization are irreversible, we choose to avoid explicit binary classification but augment the attributes with category-level information.

We consider two ways to capture the category-level information. First, we adopt the 4096-dimensional output of the second fully connected layer of a CNN [87] as an additional feature, as it has been shown the activations of the top layers of a CNN capture high-level category-related information [188]. The CNN is trained using ImageNet categories. Second, we build a general category classifier to alleviate the potential problem of the deep learning feature, namely the deep learning feature may bring examples that have common elements with the query instance even if they are irrelevant, such as skins for shoes. Combining the two types of category-level information with the category-specific attributes, the similarity between a query q and an example d in the search set is computed by

$$S(q,d) = S_{deep}(q,d) + S_{class}(d) + S_{attr}(q,d), \tag{4.5}$$

where $S_{deep}(q,d)$ is the similarity of q and d in the deep learning feature space, $S_{class}(d)$ is the classification response on d and $S_{attr}(q,d)$ is the similarity in the attribute space. The three scores are normalized to be $[0,1]$.

Datasets. We evaluate on shoes. A set of 15 shoes and in total 59 images is collected from two fashion blogs[7]. These images are recorded in streets with cluttered background,

7 http://www.pursuitofshoes.com/ and http://www.seaofshoes.com/. The properties are with the respective owners. The images are shown here only for scientific purposes.

Figure 28: Examples of two shoes from StreetShoes. As there is only 1 query example, by manual annotation, we only consider the object region to ensure the object to search is clear, as shown in the second column. The goal is to retrieve from an image collection the target images which depict the same shoe. Note large differences in scale and viewpoint between query and target images.

different from the 'clean' images in *CleanShoes*. We consider one image of a shoe as the query and aim to find other images of the same shoe. The shoe images are inserted into the test and validation parts of the Pascal VOC 2007 classification dataset [37]. The Pascal dataset provides distractor images. We refer to the dataset containing the shoe images plus distractors as *StreetShoes*. Figure 28 shows two examples. To learn the shoe classifier, we use the 300 'clean' shoes for attributes learning in Section 4.4 as positive examples and consider the training part of the Pascal VOC 2007 classification dataset as negative examples.

Implementation details. As there is 1 query image, by manually annotation we only consider the object region to ensure the target is clear. It is worthwhile to mention that although only the object part in the query image is considered, we cannot completely get rid of skins for some shoes, as shown in Figure 28. We use selective search [162] to generate many candidate locations in each database image and search over these local objects in the images as [156]. We adopt a short representation with 128 dimensions. Specifically, we reduce the dimensionality of the deep learning features and the attribute representations with a PCA reduction. And for Fisher vectors, we adopt the whitening technique proposed in [72], proven better than PCA. We reuse the attribute detectors from Section 4.4.

Results and discussions. The results are shown in Table 16. On *StreetShoes*, the proposed method of combining category-specific attributes with two types of category-level information achieves the best performance, 30.45% in mean average precision. We observe that when considering deep features alone as the category-level information, the system brings many examples of skins. The shoe classifier trained on clean shoe images

	StreetShoes
Deep(128D)	21.68
Fisher(128D)	9.38
Attributes(128D)	3.10
Deep + Fisher	19.76
Deep + Attributes	18.43
Deep + Classifier + Fisher	22.70
Deep + Classifier + Attributes	**30.45**

Table 16: Performance in mean average precision% on StreetShoes. The proposed method of combining the category-specific attributes with two types of category-level information outperforms the combination of category-level information with Fisher vector.

help eliminate these irrelevant examples. We conclude that the proposed method of combining the category-specific attributes with two types of category-level information is effective, outperforming the combination of category-level information with Fisher vector. Figure 29 shows the search results of three query instances returned by the proposed method, two success cases and a failure case.

4.7 CONCLUSION

In this chapter, we pursue generic instance search from 1 example. Firstly, we evaluate existing instance search approaches on the problem of generic instance search, illustrated on buildings and shoes, two contrasting categories of objects. We observe that what works for buildings does not necessarily work for shoes and what works worse for buildings may work well for shoes.

Secondly, we propose to use category-specific attributes to handle the large appearance variations present in generic instance search. We assume the category of the query is known, *e.g.*, from the user input. When searching among instances from the same category as the query, attributes outperform existing approaches by a large margin on shoes and cars at the expense of knowing the category of the instance and learning the attributes. For instance search from only one example, it may be reasonable to use more user input. On the building set, the category-specific attributes obtain a comparable performance.

Thirdly, we consider person re-identification as a special case of generic instance search where the query is a specific person. We show the same attribute-based approach achieves competitive performance, on par with the state-of-the-art in person re-identification.

Fourthly, we consider searching for an instance in datasets containing instances from various categories. We propose to use the category-level information to address the possible confusion of the query instance with instances from other categories. We show that combining category-level information carried by deep learning features and the

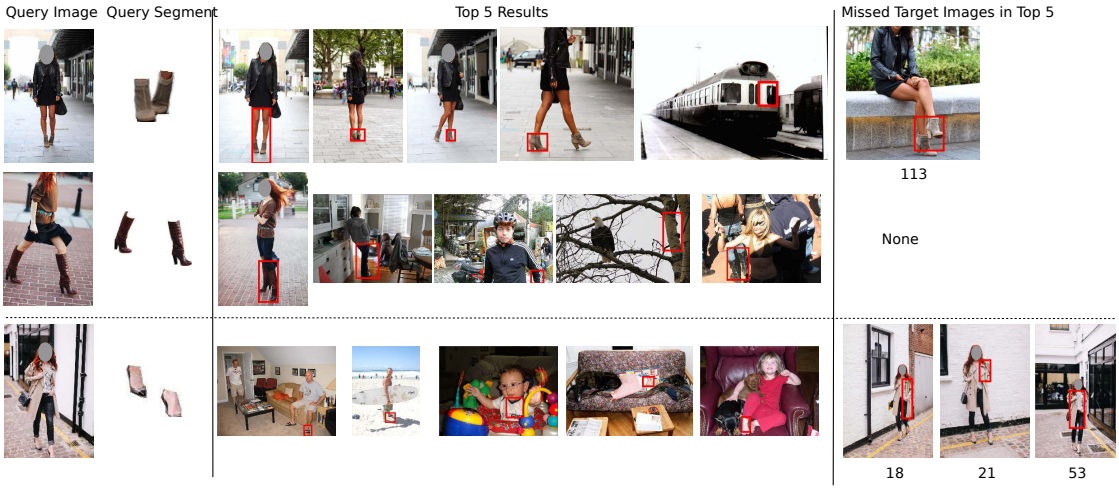


Figure 29: Search results of three query instances, two success cases (the first two) and a failure case (the third one). Only the segment is used as query. For the first instance, it has 5 relevant images in the search set, and 4 of them are returned in the top 5 positions. For the second instance, there is only 1 relevant example in the search set and it is returned at the first position. For the instance at the bottom, it has 3 relevant images and none of them are returned in the top 5. It is a very hard case, as the shoe is partially visible and the majority of the query segment is about the bare feet. Images of bare footed people appear in the top results. The correct images are ranked at 18, 21 and 53, and they are actually retrieved based on wrong information.

categorization scores with the learned category-specific attributes outperforms combining the category information with Fisher vector.

Going back to the experiments using attributes alone, the proposed same method achieves 67.87% in mean average precision (mAP) on *CleanShoes* for shoe search (Table 14), 71.74% in mAP on *Cars* for car search (Table 14), 83.06% in mAP on *OxfordPure* for building search (Table 14) and 43.6% in matching rate at rank 1 on *VIPeR* for person search (Table 15), while the best performance of existing methods are 36.73% (Table 13), 27.27% (Table 13), 87.01% (Table 13) and 45.9% (Table 15) respectively. The method is generic for instance search indeed.

5

SIAMESE INSTANCE SEARCH FOR TRACKING

5.1 INTRODUCTION

[1] At the core of many tracking algorithms is the function by which the image of the target is matched to the incoming frames. The matching function for tracking ideally provides good matching even if the target in the video is occluded, changes its scale, rotates in and out-of-plane or, undergoes uneven illumination, camera motion and other disturbing factors [152, 180]. One way to proceed is to model each of these distortions explicitly in the matching by introducing affine transformations [101], probabilistic matching [28], eigen images [142], illumination invariants [121], occlusion detection [127]. While one explicit matching mechanism may be well-fitted to solve one distortion, it is likely to disturb another.

In this work, rather than explicitly modeling the matching for particular distortions, we propose *to learn* the matching mechanism. More specifically, we suggest that we learn from external videos that contain various disturbing factors the invariances without, however, explicitly modeling these invariances. If the set of external videos is sufficiently large, the goal is to learn a generically applicable matching function *a priori*. We take extra care that there is absolutely no overlap between the videos we use for training and any of the tracking videos for evaluation. Namely, we *do not* aim to do any offline learning of the tracking targets, since in that case we would essentially learn an object detector. Instead, in the learning we focus on the generic set of object appearance variations in videos. In this way, we optimize the matching function between an arbitrary target and patches from subsequent frames. Once the matching function has been learnt on the external data we do not adapt it anymore and, we apply it as is to new tracking videos of previously unseen target objects.

We focus on learning the matching function suited for application in trackers. Hence, our aim is not to build a fully fledged tracker which might need explicit occlusion detection [129], model updating [57, 63, 189], tracker combination [189], forget mechanisms [57, 121] and other. We rather focus on the matching function alone, similar to the simplicity of the normalized cross-correlation (NCC) tracker [17, 34]. In this work, we simply match the initial target in the first frame with the candidates in a new frame and return the most similar one by the learnt matching function, without updating the target, tracker combination, occlusion detection and alike. Figure 30 illustrates the tracking algorithm.

1 Published in *IEEE Conference on Computer Vision and Pattern Recognition, 2016* [155].

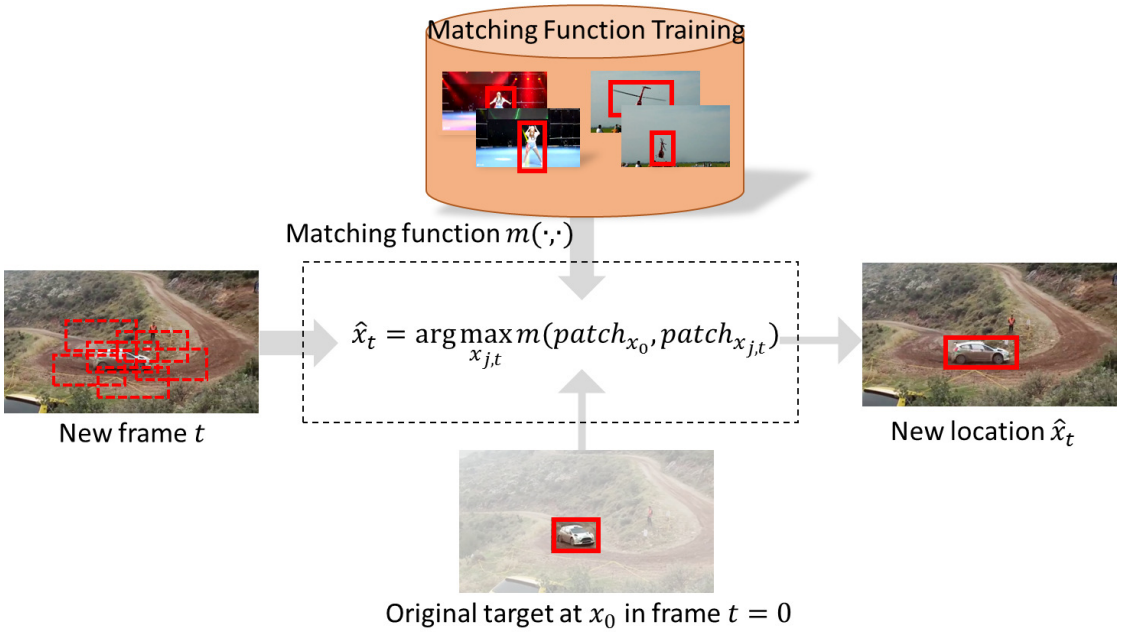


Figure 30: The tracker simply finds the patch that matches best to the original patch of the target in the first frame, using a learned matching function. The matching function is learned once on a rich video dataset. Once it has been learned, it is applied as is, without any adapting, to new videos of previously unseen target objects. We do not *apply offline target learning and the target is not included in the training video dataset.*

This approach to tracking bears some similarity to instance search [133, 156, 157, 159], where the target specified in the query image is searched for in a pile of images. Introducing matching learning [157] allows for accurate instance search of generic objects even when the relevant images in the search set show drastically different views of the target object from the query image. Here we intend to learn a generic matching function to cope with all sorts of appearance variations from tracking examples. After learning, the matching function is capable of comparing patches recorded under very different conditions for new objects, or, even for new object types that the function has not seen before.

We summarize the contributions of the work as follows. First, we propose to learn a generic matching function for tracking, from external video data, to robustly handle the common appearance variations an object can undergo in video sequences. The learnt function can be applied as is, without any adapting, to new tracking videos of previously unseen target objects. Second, on the basis of the learnt generic matching function, we present a tracker, which reaches state-of-the-art tracking performance. The presented tracker is radically different from state-of-the-art trackers. We apply no model updating, no occlusion detection, no combination of trackers, no geometric matching and alike. In each frame, the tracker simply finds the candidate patch that matches best to the initial patch of the target in the first frame by the learned matching function. Third, to learn the matching function, we use a two-stream Siamese network [18], which we design specifically for tracking. Further, in the absence of any drifting that one would expect by on-the-fly model updating, the proposed tracker allows for successful target object

re-identification after the target was absent for a long period of time, *e.g.*, a complete shot.

5.2 RELATED WORK

Matching functions in tracking One of the most basic concept of tracking is the direct matching between the intensity values of the target patch and the patches taken from the incoming image. The oldest tracking algorithm does just that by normalized cross-correlation [17, 34]. Its simplicity is also its strength, still being in use as part of the TLD-tracker [82]. Subsequent trackers have reconsidered the matching function by focusing on the various distortions to the target image faced in tracking. The Lucas and Kanade tracker [101] adds an affine transformation to the matching function. MST [28] relies on probabilistic matching. FRT [2] uses the earth mover's distance matching. And IVT [142] matches by the metric of eigen images obtained during tracking. L1T [106] is successful with L1-metric matching on graphs of fragments. SPT [174] uses super-pixels for matching, HBT [49] uses HOG-features in a probabilistic approach, and FBT [121] uses color invariants for robustness against illumination variations. Different from all methods above, which pursue explicit modeling of the matching function, this work aims to *learn* the matching function from example videos annotated with the correct boxes.

Recent tracking methods In recognition of the hardness of the task, composite trackers have been introduced. TLD [82] integrates the NCC matching for recovery with a differential tracker and a complex updating model. Struck [57] is based on structural SVM with the displacement as the continuous output, with a cautious update mechanism. More recently, MEEM [189] successfully learns and updates a discriminative tracker, keeping a set of historical snapshots as experts who derive the per frame prediction based on an entropy regularized optimization. Alien [129] is a successful long-term tracker relying on oversampling of local features and RANSAC-based geometric matching. In the very recent MUSTer [65] one component stores short-term memories of the target for short-term tracking using integrated correlation filters, where the long-term memory is based on RANSAC matching again. Finally, the AND-OR tracker [179] proposes a discriminative learning of hierarchical, compositional and-or graphs that account for the appearance and structural variations of the object. In this work, we focus on simple tracking inference scheme, namely finding the patch that matches best to the initial target in the first frame. The complexity, instead, is incorporated externally, where the matching function is trained to be robust against appearance variations. Hence, rather than learning on-the-fly, we learn what can be encountered *in general* without requiring target-specific learning. Once learned, the matching function can be built in the successful, aforementioned composite trackers to enhance their performance.

Deep learning in tracking [172] uses a stacked denoising autoencoder to learn tracking features. The features are performing poorly, however. [90] learns a target classifier online, which is fundamentally hampered by a lack of data. [64] focuses on learning target-specific saliency map using pre-trained ImageNet network. [170] pre-trains a convolutional neural network for measuring generic objectness on ImageNet 2014 detection set and adapts the network online to predict the target-specific objectness. Compared to previous works, this work focuses on a different part of a tracker. We

employ deep neural networks to learn a generic matching function from rich external data to compare patches for use in tracking.

Instance Search Instance search from one example, also known as particular object retrieval, is related to object tracking, especially when localized [79, 156]. The most popular paradigm is based on matching local image descriptors between the query and the candidate image [73, 133, 136, 151, 156, 158] and is especially accurate for buildings [133]. Recently, [157] proposed to learn a robust representation for instance search of less textured, more generic objects, showing good accuracy despite the significant appearance changes between the query and the database images. We derive some inspiration from [157]. We propose to learn a robust matching function for matching arbitrary, generic objects that may undergo all sorts of appearance variations. We focus, however, on tracking. Instead of focusing on a specific category *e.g.*, shoes, and learning from images with a white background [157], we learn in this work a universal matching model suited for tracking that applies to all categories and all realistic imaging conditions.

Siamese architecture [18] proposes the two-stream Siamese architecture for signature verification. Later, the two-stream network architecture has been applied to face verification [25, 154], ground-to-aerial image matching [95], local patch descriptor learning [56, 186] and stereo matching [187]. In this work, we design a Siamese network-architecture to learn robust and generic representation for object tracking, aiming to be invariant to all sorts of appearance variations in practical tracking scenarios.

Fast localization Tracking also bears resemblance to the object localization problem. Usually, it requires efficient processing of multiple regions in one frame. [93] proposes efficient region computation by reordering the encoding, pooling and classification steps for the 'shallow' representations such as Fisher vector [132]. Recent work by Girshick [47] proposes an efficient way of processing multiple regions in one single pass through the deep neural network for fast object detection. Inspired by [47], we incorporate the region-of-interest pooling layer into our network for fast processing of multiple regions in one frame for tracking.

5.3 SIAMESE INSTANCE SEARCH TRACKER

In the following we describe the proposed method for tracking, which is coined Siamese INstance search Tracker, SINT for abbreviation. We first present the matching function, which is the core of the tracker. Then we describe the simple online tracking inference.

5.3.1 *Matching Function*

To learn a matching function robust to all sorts of distortions as described earlier, we need a model that operates on pairs of data, (x_j, x_k). A network architecture that has been successfully shown to work well on pairs of data is the two-stream Siamese architecture [18, 25]. A Siamese architecture builds on top of convolutional networks. Next, we analyze the different components of the proposed two-stream network which we coin *Siamese Invariance Network*.

Network architecture We use a Siamese architecture composed of two branches. The Siamese network processes the two inputs separately, through individual networks

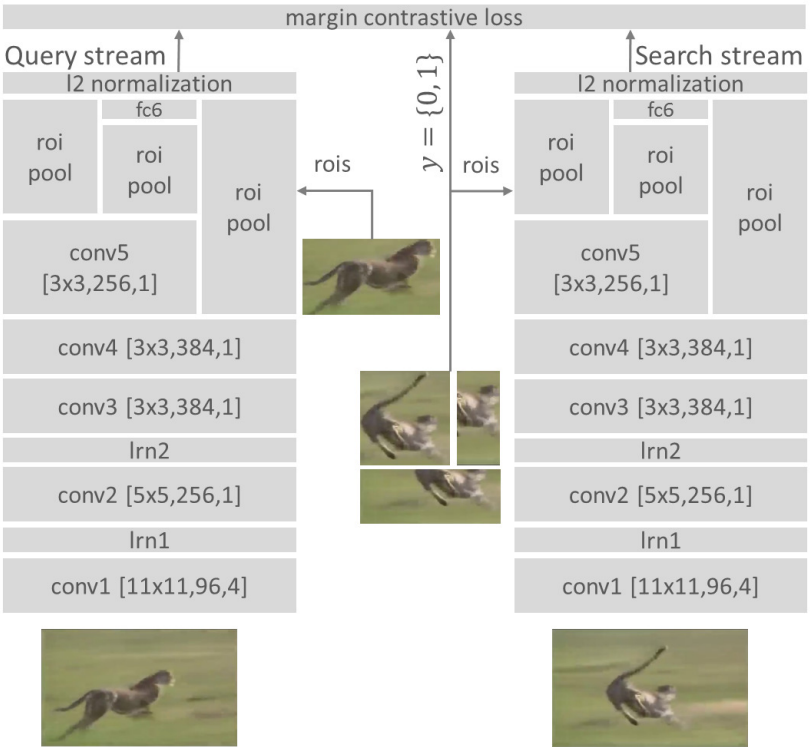
that usually take the form of a convolutional neural network. For individual branches, we design and compare two different network architectures, a small one similar to AlexNet [87] and a very deep one inspired by VGGNet [150] (Figure 31). In the following we highlight the distinctive designs of the networks as compared to the successful AlexNet and VGGNet.

Being largely a localization task the tracking problem is naturally susceptive to rough discretizations. Aiming for precise localization, we design our network with very few maxing pooling layers, fewer than the networks in [87, 150]. Indeed, as max pooling maintains only the strongest of the activations from a local neighborhood to use as input for the subsequent layers, the spatial resolution of the activations is aggressively reduced, at the very least by 50% only in the simple case of 2×2 local neighborhoods. An advantage of max pooling is it introduces invariance to local deformations. However, this is more important for object categorization, where the objects vary a lot in appearance. In tracking even if the target object changes its appearance over time, it still remains the same object in all frames. Moreover, it is important to be able to follow the small appearance changes, such as local deformations, of the object over time. Regarding the two architectures we propose, for the AlexNet-like small net we do not include any max pooling layer (see Figure 31a), while for the VGG-like large net, we only have two max pooling at the very early stage (see Figure 31b), as the lower level layers learn filters of very small receptive fields and their max pooling layers are important to maintain robustness to local noise.

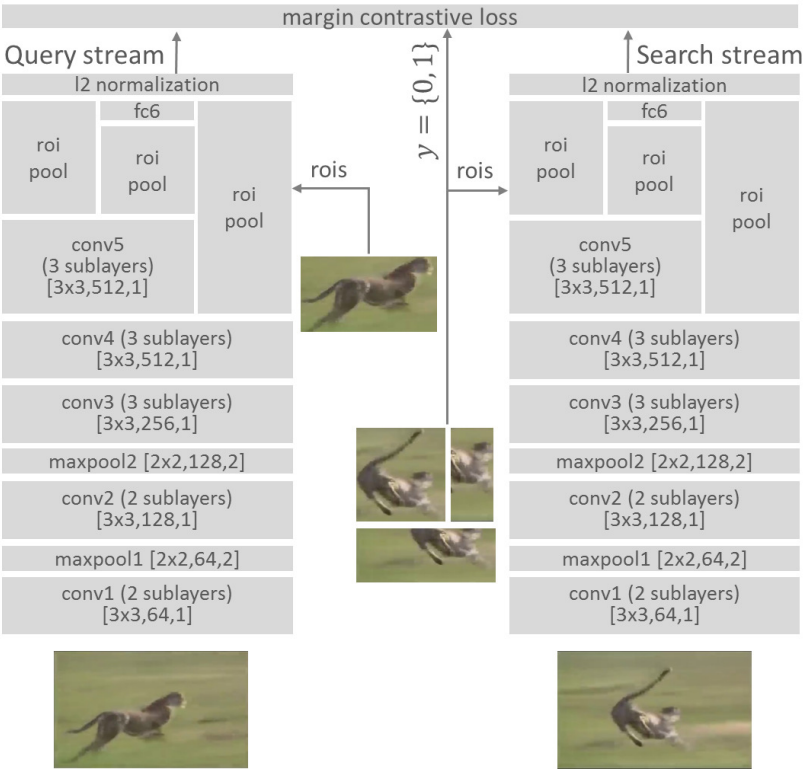
In tracking one typically needs to evaluate hundreds of candidate regions for the next frame. Although one can simply pass the candidate regions into the network for processing independently, this would lead to a severe computation overhead, especially since there is a significant overlap between the candidate regions. Therefore, we employ a region pooling layer [47] for the fast processing of multiple overlapping regions. Each branch of the Siamese architecture takes as input one image and a set of bounding box regions. The network first processes the entire image for a few layers, then the region pooling layer converts the feature map from a particular region into a fixed-length representation. Having a fixed length representation, one can now proceed to the subsequent layers.

The layers in a deep network capture progressively more abstract representations [188]. Typically, the filters of the lower layers get activated the most on lower level visual patterns, such as edges and angles, whereas higher layers get activated the most on more complex patterns, such as faces and wheels. Also, the deeper one layer is, the more invariant it is to appearance changes but also less discriminative, especially for instance-level distinction. In tracking we do not know the type of target object we want to track, whether it is highly textured with rich low level patterns or not. We do not know either the complexity of the background, whether there are confusing objects in which case higher discrimination would probably be more helpful. For this reason we propose to use the outputs from multiple layers as the intermediate representation that is then fed to the loss function. Similar observations have also been made in [58, 98] for different tasks, semantic segmentation and fine-grained localization specifically. All activations are pooled using the region pooling layers.

Given that modern convolutional neural networks use rectified linear units that do not bound the output values, the nonlinear activations can vary a lot in the range of



(a)



(b)

Figure 31: The proposed two-stream Siamese networks to learn the generic matching function for tracking. 'conv', 'lrn', 'maxpool', 'roipool' and 'fc' stand for convolution, local response normalization, max pooling, region-of-interest pooling [47] and fully connected layers respectively. Numbers in square brackets are kernel size, number of outputs and stride. The fully connected layer has 4096 units. All conv layers are followed by rectified linear units (ReLU) [115].

values they produce. As such and without considerations, the network output and the loss function will be heavily influenced by the scale of the generated features and not their representation quality. To avoid this we propose to add an ℓ_2 normalization layer before the loss layer. The normalization layer is applied on each of the layer activations that are fed to the loss layer and has the property of maintaining the direction of the feature, while forcing features from different scales to lie on the same unit sphere.

Compared to standard convolutional neural networks, AlexNet and VGGNet [87, 150], our architecture has several differences, highlighted above. However, we also explicitly design our networks to be compatible to AlexNet and VGGNet. In this way, we are able to initialize the weights of our networks using the ImageNet-pretrained AlexNet and VGGNet to avoid training from scratch, something that would likely lead to overfitting. Last, note that we keep the parameters of the two convolutional network branches tied together, as there would be an increased danger of overfitting otherwise.

Network input Our training data consist of videos of objects, whose bounding box location is provided to us. To emulate the instance search paradigm and to avoid confusion, we coin the first stream of our network as query stream, whereas the second stream of our network as search stream. For the query stream we randomly pick one frame from the video and use the annotated patch of the target. Since we want to be robust to as many types of variations that we might face when tracking novel objects as possible, for the search stream we randomly pick another video frame that does not need to be adjacent to the frame of the query stream. From the frame of the search stream we sample boxes and the ones that overlap more than ρ_+ with the ground truth are deemed positives, while the ones that overlap less than ρ_- with the ground truth are deemed negatives. From these we form positive and negative pairs of data that we use for the training.

Loss In the end, the two branches in the Siamese Invariance Network are connected with a single loss layer. For tracking we want the network to generate feature representations, that are close by enough for positive pairs, whereas they are far away at least by a minimum for negative pairs. Bearing these requirements in mind and inspired by [25], we employ the margin contrastive loss

$$\mathcal{L}(x_j, x_k, y_{jk}) = \frac{1}{2} y_{jk} D^2 + \frac{1}{2}(1 - y_{jk}) \max(0, \epsilon - D^2), \tag{5.1}$$

where $D = \|f(x_j) - f(x_k)\|_2$ is the Euclidean distance of two ℓ_2-normalized latent representations, $y_{jk} \in \{0, 1\}$ indicates whether x_j and x_k are the same object or not, and ϵ is the minimum distance margin that pairs depicting different objects should satisfy.

Data As tracking is an inherently online task, where no training data related to the target object are available, it is important to emphasize that *the network is learnt on external videos that do not appear in the tracking evaluation sets*. The data should be varying enough, covering a good amount of semantics and not focus on particular objects, otherwise the tuned network parameters will overfit to particular object categories. Furthermore, as we do not explicitly learn types of invariances, namely we do not learn "illumination invariance" separately from "scale invariance", therefore in the external data

we do not need any specific variation labels. The only requirement is the box annotations within the video following a particular object.

5.3.2 *Tracking Inference*

Once we have completed the learning of the matching function, we are ready to deploy it as is to tracking, without any further adapting. We propose a simple tracking strategy. As the only reliable data we have for the target object is its location at the first frame, at each frame we compare the sampled candidate boxes with the target object at the first frame. We pass all the candidate boxes from the search stream of our network and pick the candidate box that matches best to the original target,

$$\hat{x}_t = \arg\max_{x_{j,t}} m(x_{t=0}, x_{j,t}), \tag{5.2}$$

where $x_{j,t}$ are all the candidate boxes at frame t, m is the learned matching function, $m(x,y) = f(x)^T f(y)$.

Candidate sampling We employ the radius sampling strategy [57]. More specifically, around the predicted location of the previous frame we sample locations evenly on circles of different radii. Different from [57], to handle scale variations we generate at each sampled location multiple candidate boxes at different scales.

Box refinement Provided that the box prediction is accurate enough, [41,48] showed that a refinement step of the boxes can improve localization accuracy significantly. To this end we adopt their strategy and refine at each frame the predicted bounding box further.

As in [48] we train four Ridge regressors for the (x,y) coordinates of the box center and the width and height (w,h) of the box based on the first frame. The regressors are not updated during tracking in order to avoid the risk of contaminating the regressors with noisy data. For each frame, the regressors take the representation of the picked candidate box as input and produce a refined box.

5.4 EXPERIMENENTS

5.4.1 *Implementation Details*

Candidate Sampling We use the radius sampling strategy [57] to generate candidate boxes. We use 10 radial and 10 angular divisions. The search radius is set to be the longer axis of the initial box in the first frame. At each sample location, we generate three scaled versions of the initial box with the scales being $\{\frac{\sqrt{2}}{2}, 1, \sqrt{2}\}$.

Network training We use the ALOV dataset [152] for training and validation. We choose ALOV for training as it covers many types of variations one could expect in tracking. *We exclude the 12 videos in ALOV that are also in tracking benchmark (OTB) [180], as we evaluate the proposed tracker on OTB.* After removing the 12 videos, the training set and the tracking evaluation set have no common objects. From every two frames in a video, we generate multiple pairs. One element in a pair is the groundtruth

bounding box in one frame and the other element is a box sampled in the other frame. The pair is considered to be positive if the sampled box has a intersection-over-union overlap larger than 0.7 with the corresponding groundtruth box and considered to be negative if the overlap is smaller than 0.5. The training pairs and validation pairs are generated from different videos, and therefore from different objects. For training, in total we have sampled from ALOV dataset $60,000$ pairs of frames and each pair of frames has 128 pairs of boxes. For validation, we have gathered $2,000$ pairs of frames and the same as for training each pair of frames contains 128 pairs of boxes.

Instead of training the two-stream Siamese network from scratch, we load the pre-trained network parameters and fine tune the Siamese network. Specifically, we use the networks pre-trained for ImageNet classification, available in the Caffe library [78]. The initial fine tuning learning rate is 0.001 and the weight decay parameter is 0.001. The learning rate is decreased by a factor of 10 after every 2 epochs. We stop tuning when the validation loss does not decrease any more.

5.4.2 *Dataset and evaluation metrics*

Dataset To evaluate the tracking performance, we use the online tracking benchmark (OTB) [180]. OTB is a collection of 50 videos. 51 tracking sequences are defined with bounding box annotations. The dataset covers various challenging aspects in object tracking, such as fast motion, deformation, background clutter and occlusion.
Evaluation metrics We follow the evaluation protocol of [180], where two metrics are used: success plot and precision plot. Both metrics measure the percentage of successfully tracked frames. For the success plot, a frame is declared to be successfully tracked if the estimated bounding box and the groundtruth box have an intersection-over-union overlap larger than a certain threshold. For precision plot, tracking on a frame is considered successful if the distance between the centers of the predicted box and the groundtruth box is under some threshold. A plot is given by varying the threshold values. Tracking algorithms are ranked based on the area under curve (AUC) score for the success plot and precision at threshold 20 (Prec@20) for the precision plot. We use the available toolkit provided by the benchmark to generate plots and numbers. In the following, we also use success rate where needed, *i.e.*, the percentage of successfully tracked frames.

5.4.3 *Design evaluation*

We first validate our design choices of the network. In this sets of experiments, box refinement is not considered.

Network tuned generically on external video data vs. network pre-tuned on ImageNet vs. network fine tuned target-specifically on first frame In this experiment, we show the effectiveness of the Siamese network tuned on external data. To that end, we compare the Siamese fine tuned AlexNet-style network using ALOV (denoted as "Siamese-finetuned-alexnet-fc6") with the ImageNet pre-tuned AlexNet ("pretrained-alexnet-fc6") and the Siamese fine tuned network using the training pairs gathered in

	AUC	Prec@20
(a) pretrained-alexnet-fc6	42.8	66.3
(b) firstframe-Siamese-finetuned-alexnet-fc6	44.0	67.9
(c) Siamese-finetuned-alexnet-fc6	47.4	72.0
(d) pretrained-alexnet-fc6-nomaxpooling	50.0	70.8
(e) Siamese-finetuned-alexnet-fc6-nomaxpooling	53.9	74.8
(f) Siamese-finetuned-alexnet-conv45fc6-nomaxpooling	55.0	76.2
(g) Siamese-finetuned-vgg16-conv45fc6-nomaxpooling	59.2	83.6

Table 17: Evaluation of different architectural and design choices of the Siamese invariance network for tracking on the OTB dataset [180]. We use the recommended evaluation methods, namely the area under the curve (AUC) for the success plot and the precision at 20 (Prec@20) for the precision plot.

the first frame ("firstframe-Siamese-finetuned-alexnet-fc6"). In this comparison, all three use a single layer *fc6* for feature representation. As shown in the rows *(a)-(c)* of Table 17, the Siamese fine tuned network using ALOV *(c)* significantly improves over the pre-tuned net *(a)*, while fine tuning on the first frame *(b)* gives a marginal improvement. We conclude that Siamese networks fine tuned using large amount of external data are to be preferred.

To max pool or not to max pool? We now examine our design choice of having no maxing pooling layers in the network ("pretrained-alexnet-fc6-nomaxpooling" vs. "pretrained-alexnet-fc6" and "Siamese-finetuned-alexnet-fc6-nomaxpooling" vs. "Siamese-finetuned-alexnet-fc6"). As shown in Table 17, *(d)* vs. *(a)* and *(e)* vs. *(c)*, including max pooling layers deteriorates accuracy, as expected due to the reduction of the resolution of the feature maps which causes poor localization. When inspecting the results when no max pooling layers are included, the success rate improvement is higher at higher intersection-over-union overlap ratios, see Table 18. We conclude that max pooling layers are not necessary for our Siamese invariance network with small AlexNet-style architecture.

Multi-layer features vs. single-layer features Next, we evaluate whether it is more advantageous to use features from a single layer or from multiple layers. We compare "Siamese-finetuned-alexnet-conv45fc6-nomaxpooling", which uses the outputs of layers *conv4*, *conv5* and *fc6* as features, with "Siamese-finetuned-alexnet-fc6-nomaxpooling", which uses the output of *fc6* as feature. Table 17 shows that using multi-layer features is helpful ((e) vs. (f)). We conclude that using features from multiple layers is advantageous.

Large net vs. small net Lastly, we compare a VGGNet-style architecture with an AlexNet-style architecture ("Siamese-finetuned-vgg16-conv45fc6-nomaxpooling" vs. "Siamese-finetuned-alexnet-fc6-nomaxpooling"). Both use as features the outputs of three layers. As shown in the last two rows *(f)* and *(g)* of Table 17, using a deeper network improves the performance significantly.

	sr@0.3	sr@0.5	sr@0.7
pretrained-alexnet-fc6	68.3	46.2	19.6
pretrained-alexnet-fc6-nomaxpooling	75.3	58.1	32.6
Siamese-finetuned-alexnet-fc6	74.6	56.2	25.4
Siamese-finetuned-alexnet-fc6-nomaxpooling	79.3	67.6	38.8

Table 18: Success rates (sr) of the tracker at three intersection-over-union overlap ratios for different network architectures. From the table it is clear that a network architecture without max pooling delivers a more precise localization and hence a better matching function.

5.4.4 *State-of-the-art comparison*

Overall comparison In addition to the 29 trackers included in the benchmark [180], *e.g.*, TLD [82], Struck [57] and SCM [196], we also include the most recent trackers for comparison. The included recent trackers are TGPR [44], MEEM [189], SO-DLT [170], KCFDP [67] and MUSTer [65].

As described earlier, the proposed SINT focuses on the tracking matching function, while having a simple online inference. As a preliminary demonstration that SINT can be further improved by employing more advanced online components, we also evaluate a variant of SINT, coined SINT+, which uses an adaptive candidate sampling strategy suggested by [171] and optical flow [19]. In SINT+, the sampling range is adaptive to the image resolution, set to be $30/512 * w$ in this experiment, where w is the image width. Optical flow is used in SINT+ to filter out motion inconsistent candidates. Specifically, given the pixels covered by the predicted box in the previous frame and the estimated optical flow, we know where those pixels are in the current frame and we remove the candidate boxes that contain less than 25% of those pixels, as these candidates are deemed inconsistent to the motion.

Figure 32 shows the overall performance. For clarity, only the top performing trackers are shown. Despite relying on a simple NCC-like tracking inference, SINT reaches state-of-the-art performance, being tantalizingly close to MUSTer [65] and more accurate than others by a considerable margin. SINT+, with an adaptive sampling and a simple use of optical flow, further improves SINT, outperforming clearly all state-of-the-art other trackers.

Temporal and spatial robustness To verify the robustness of the proposed tracker, we conduct the temporal robustness evaluation (TRE) and spatial robustness evaluation (SRE) defined by the benchmark. The results are summarized in Table 19. Compared to MEEM and MUSTer, SINT is temporally and spatially the same as robust, if not better.

Per distortion type comparison Further, the 50 videos in the benchmark are annotated with 11 distortion types(*e.g.*, illumination variation, occlusion *etc.*). To gain more insights, we evaluate the performance of SINT for individual attributes and compare with MUSTer [65]. SINT performs better in 6 and 7 out of the 11 groups for the AUC and the Prec@20 metrics respectively (see Figure 33). It is observed that MUSTer is better mainly in "occlusion" and "deformation", whereas SINT is better in "motion blur", "fast

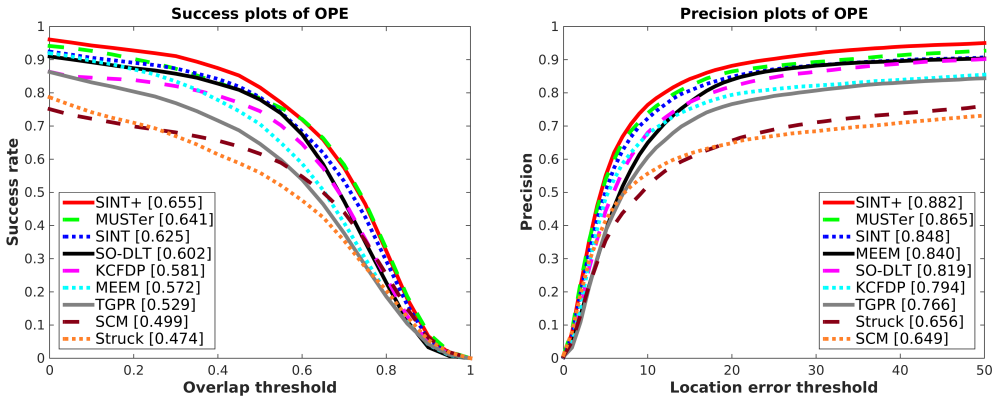


Figure 32: State-of-the-art comparison on OTB [180]. In spite of the fact that the online part of the proposed SINT is just selecting the patch that matches best to the target in the first frame, SINT is on par with state-of-the-art tracker. SINT+, using a better candidate sampling than SINT and optical flow as an additional component, achieves the best performance.

	OPE	TRE	SRE
MEEM	57.2 / 84.0	58.5 / 83.2	51.8 / 76.9
MUSTer	62.1 / 83.6	60.9 / 81.1	56.2 / 78.9
SINT	62.5 / 84.8	64.3 / 84.9	57.9 / 80.6

Table 19: Robustness evaluation on OTB, measured in AUC/Prec@20. OPE is one-pass evaluation. TRE and SRE are temporal and spatial robustness evaluation. The results of MEEM are taken from [189] and the results of MUSTer are obtained using the publicly available code.

motion", "in-plane rotation", "out of view" and "low resolution".

Failure modes of SINT When similar objects appear in view, the tracker may jump from the target to another as it only looks for the maximum similarity with the original patch of the target in the first frame (Figure 34: left). And, when there is large occlusion, the matching function might suffer (Figure 34: right).

5.4.5 *Additional sequences and re-identification*

We now further illustrate the strength of the proposed SINT on 6 newly collected sequences from YouTube. We downloaded the sequences so that they are extra challenging in terms of tracking distortions as defined by [180]. Figure 35 shows example frames from these sequences. The sequences have considerable degrees of scale change ("Fishing", "Rally", "BirdAttack" and "GD"), fast motion ("BirdAttack", "Soccer" and "Dancing"), out-of-plane rotation ("Rally" and "Dancing"), non-rigid deformation ("Fishing", "BirdAttack" and "Dancing"), low contrast ("Fishing"), illumination variation ("GD" and "Dancing") and poorly textured objects ("Fishing" and "BirdAttack").

We evaluate the proposed tracker, SINT, with MEEM [189] and MUSTer [65] on these sequences. The performance is summarized in Table 20, where we adopt the AUC

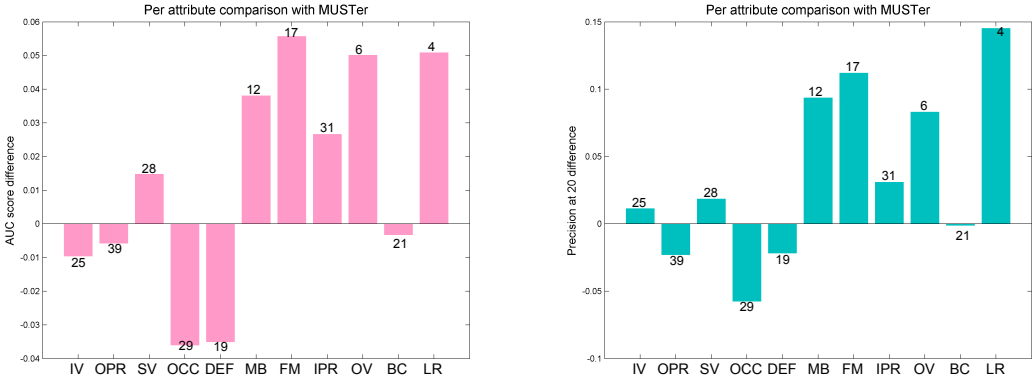


Figure 33: Per attribute comparison on AUC score and Prec@20 of the proposed SINT with MUSTer [65]. The bars are the performance difference between SINT and MUSTer. Positive means SINT is better. The integer number at each bar is the number of tracking sequences belonging to that group.

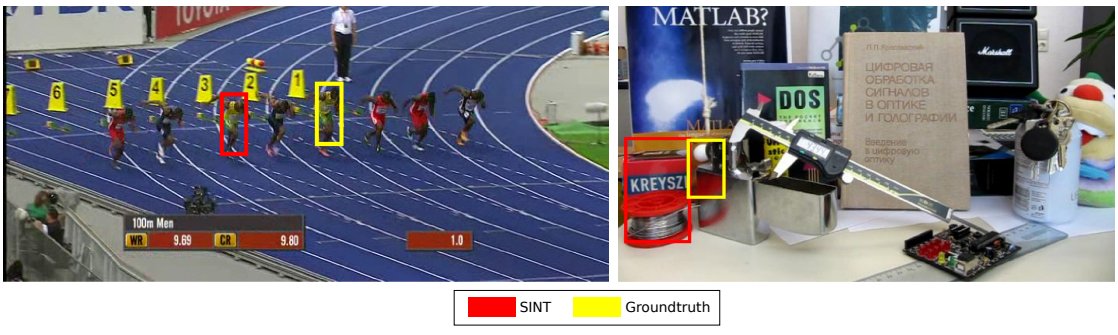


Figure 34: Failure cases of SINT: similar confusing object (left) and large occlusion (right). Examples are from OTB sequences 'Bolt' and 'Lemming' respectively. In the left example, the tracker fires on another Jamaican runner in the same uniform as the target. In the right example, the target is heavily occluded by the lighter.

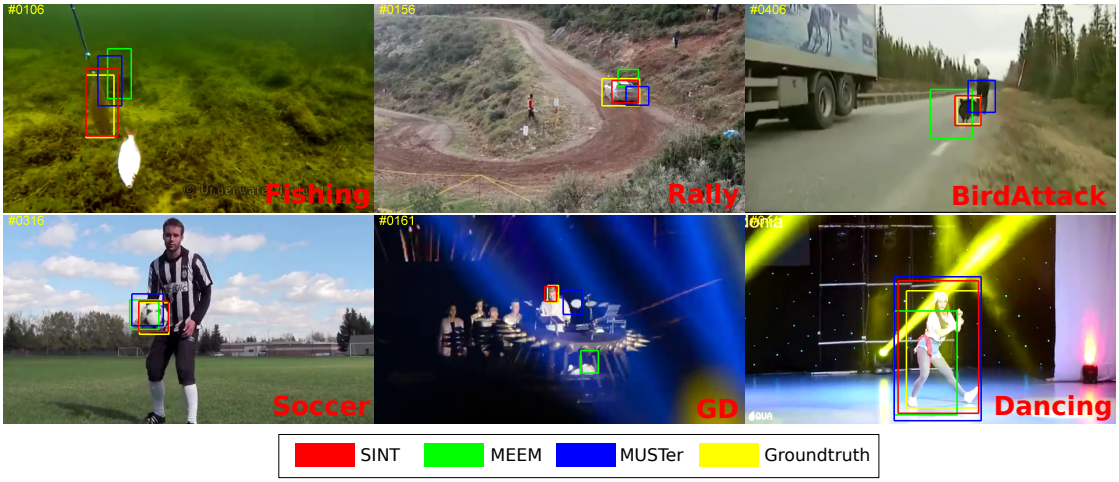


Figure 35: Example frames from the 6 test sequences.

	MEEM [189]	MUSTer [65]	SINT
Fishing	4.3	11.2	53.7
Rally	20.4	27.5	53.4
BirdAttack	40.7	50.2	66.7
Soccer	36.9	48.0	72.5
GD	13.8	34.9	35.8
Dancing	60.3	54.7	66.8
mean	29.4	37.8	58.1

Table 20: Comparison on AUC score of the proposed SINT with MEEM [189] and MUSTer [65].

score metric from the benchmark [180]. Results show that SINT is again a competitive tracker, outperforming MUSTer [65] and MEEM [189].

We, furthermore, observe that provided a window sampling over the whole image using [198], SINT is accurate in target re-identification, after the target was missing for a significant amount of time from the video. We illustrate this in Figure 36, where we track *Yoda*. As shown in Figure 36, the tracker has good capability of discovering the target when it re-enters the scene after being absent for a complete shot.

5.5 CONCLUSION

This work presents Siamese INstance search Tracker, SINT. It tracks the target, simply by matching the initial target in the first frame with candidates in a new frame and returns the most similar one by a learned matching function. The strength of the tracker comes from the powerful matching function, which is the focus of the work. The matching function is learned on ALOV [152], based on the proposed two-stream very deep neural network. We take extra care that there is absolutely no overlap between the training videos and any of the videos for evaluation. Namely, we *do not* aim to do any pre-learning of the tracking targets. Once learned, the matching function is used as is, without any adapting,

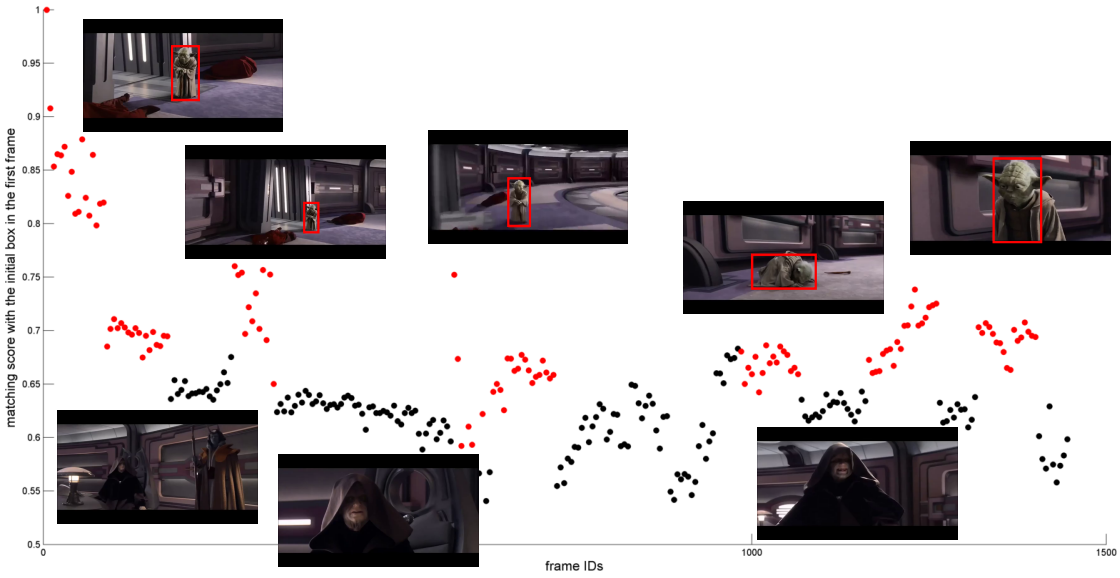


Figure 36: The capability of the tracker to re-discover the target, illustrated on a 1500-frame, 12-shot Star Wars *video. One object (*Yoda*) is appearing in 6 of the shots, while being absent in the intermediate ones. Red dots indicate* Yoda *is present while black dots indicate* Yoda *is absent. Y-axis is the matching score with the target at the first frame. The results show good capability of the tracker to discover the target when it re-enters the scene.*

to track arbitrary, previously unseen targets. It turns out the matching function is very effective in coping with common appearance variations an object can have in videos. The simple tracker built upon the matching function, reaches state-of-the-art performance on OTB [180], without updating the target, tracker combination, occlusion detection and alike. Further, SINT allows for target re-identification after the target was absent for a complete shot, demonstrated on a 1500-frame, 12-shot *Star Wars* video.

6

CONCLUSIONS

6.1 SUMMARY OF THE THESIS: VISUAL INSTANCE SEARCH FROM ONE EXAMPLE

In Chapter 1, we start with a general discussion on the notion of *instance*. Then we give a precise definition of *visual instance* and pose the main question for this thesis: given 1 image of a visual instance, how to find all the examples of the instance automatically from a collection? Centered around the main question, we phrase four research questions, including how to explore the spatial extent of an instance, how to utilize certain domain knowledge for specialized instance search, how to approach generic instance search and how to address object tracking as an instance search problem.

In Chapter 2 [156], we propose an instance search method which exploits locality for better search accuracy. Different from prior work, which relies on global image representation for the search, we proceed by including locality at all steps of the method. We consider many boxes per database image as candidate targets to search locally in the picture using an efficient point-indexed representation. The same representation allows the application of very large vocabularies in the powerful Fisher vector and VLAD to search locally in the feature space. And, we propose an exponential similarity function to further emphasize locality in the feature space. Locality is advantageous in instance search as it will rest on the matching unique details. We demonstrate a substantial increase in instance search performance from one example on three standard datasets with buildings, logos, and scenes from a distance.

Chapter 3 [83] puts an emphasis on logos. Logos are a special type of instances. Text is often a part of a logo. We exploit the recognized text in images for logo search. To detect words in images, a generic and fully unsupervised word box proposal method is introduced. Where the state-of-the-art text detection methods aim at high f-score, the proposed method is designed to obtain high recall. The detected word regions are used as input of a state-of-the-art word recognition method to perform a word-level textual cue encoding. Adding the textual cues leads to a significant improvement in logo search accuracy. In addition to logo search, we show the proposed method also works effectively for fine-grained business places classification. Moreover, we empirically validate that high recall in word detection is more relevant than high f-score for the two tasks.

Chapter 4 [157] aims for generic instance search from 1 example where the instance can be an arbitrary 3D object like shoes, not just near-planar and one-sided instances like buildings and logos. First, we evaluate state-of-the-art instance search methods on this problem. We observe that what works for buildings loses its generality on shoes. Second, we propose to use automatically learned category-specific attributes to address the various

degrees of appearance variations present in different types of instances. Searching among instances from the same category as the query, the proposed attribute-based method outperforms existing approaches by a large margin on shoes and cars and performs on par with the state-of-the-art on buildings. Third, we treat person re-identification as a special case of generic instance search. On the popular VIPeR dataset, we reach state-of-the-art performance with the same method. Fourth, we extend our method to search instances without restriction to the specifically known category. We show that the combination of category-level information and the category-specific attributes is superior to the alternative method combining category-level information with low-level features such as Fisher vector.

In Chapter 5 [155], we make a connection between generic instance search from 1 example and visual object tracking in videos. We develop a tracker based on the rationale of visual instance search. The tracker is radically different from state-of-the-art trackers: it has no model updating, no occlusion detection, no combination of trackers, no geometric matching, and still delivers state-of-the-art tracking performance. The presented tracker simply treats the initial patch of the target provided in the first frame as the 1 query example and searches for the most similar candidate patch in every incoming frame. The strength of the tracker comes from a similarity function, which is extensively trained in a generic way, *i.e.*, without any data of the target, using a newly designed Siamese deep neural network. Once learned, the similarity function is used as is, without any adapting, to track previously unseen targets. The learned similarity function is so powerful that a simple tracker built upon it, which only uses the original observation of the target from the first frame, suffices to reach state-of-the-art performance. Further, we show the proposed tracker even allows for target re-identification after the target was absent for a complete video shot.

6.2 GENERAL CONCLUSIONS

We conclude the thesis by revisiting the questions posed in the introduction.

We begin with the question: *can we exploit locality for better instance search accuracy?* We find in Chapter 2 that localized search in the image for an instance by evaluating multiple local boxes is advantageous, as the spatial extent of an instance in images is often limited to a portion of an image and the signal to noise ratio within the bounding box is much higher than in the entire image. We show that emphasizing locality in the feature space by efficiently employing a large visual vocabulary and an exponential similarity metric to impose a strict matching criterion is effective in reducing the confusion from other, similar instances. The proposed method, without being optimized for a specific class of objects, leads to a substantial increase in search accuracy on buildings, logos, and scenes from a distance. However, imposing a strict criterion on matching local details is only suited for textured and one-sided instances with a limited viewpoint variation, not working well when large viewpoint variation exists such as when searching for shoes, as shown later in Chapter 4.

As to the second question: *can we exploit domain knowledge for better search accuracy on logos?* Our findings in Chapter 3 support the conclusion that the text information in the logos is useful domain knowledge for logo search. We show the

recognized text in images is an effective cue for logo retrieval. The remaining challenge is that it is hard to recognize the text in exotic orientations and fonts.

Observing that commonly studied instances like buildings, logos, and remote scenes essentially have a common visual characteristic - they are all nearly 2D and one-sided - we pose the question: *can we design a generic method capable of searching for an arbitrary visual instance?* We show the proposed method in Chapter 4 that learns category-specific attributes is a generic solution. The same method can be applied to different kinds of instances, including 2D one-sided objects like buildings, and 3D multi-sided objects like shoes. When searching among instances from the same category, the same method is demonstrated effective on *shoe*, *car* and *person*, as examples of 3D multi-sided objects, and *building*, as an example of approximately 2D one-sided objects. When searching instances in a dataset containing all classes of instances, we show that combining the category-specific attributes and category-level information is an effective tactic, similar to the two-step identification procedure often employed by humans to explain the uniqueness of an instance. Compared to Chapter 2, Chapter 4 takes one step further in generic instance search. The current limitation is that a group of visual aspects need to be learned from a set of instances for every category, although the learned aspects are generalizable to new, previously unseen, instances of the same category. Considering that categories are often related (*e.g.*, hierarchically [31]), co-occurring [107] and sharing visual aspects [89], there might exist a structured set of visual aspects with a limited size, which, once learned, serves as a universally applicable basis to derive specificity of any instance of any kind.

Finally, realizing the strong connection between visual instance search and tracking, we pose the last question: *can we address tracking as instance search problem (over the video at hand)? That is, can we handle tracking without taking the temporal coherence into account?* We show that the proposed tracker in Chapter 5, which only has an instance search core armed with a powerful, end-to-end learned similarity metric, suffices to deliver state-of-the-art tracking performance. The proposed method establishes a new framework for tracking. It only requires one-time offline learning, and once learned, it is ready to track new, previously unseen, objects without any online adapting. And, surely, the tracker can be further adapted to better handle certain specific situations, if needed.

In addition to the remaining challenges mentioned above, a few other directions are worth visiting for future work. One important missing piece in the current visual instance search systems is a generic verification scheme, also known as self-awareness mechanism. Although geometric verification has shown success on rigid and textured instances, it is hard to generalize to non-rigid or poorly textured instances. A generically applicable verification scheme will not only help make a large step towards realizing arbitrary visual instance search, but also will help solve one of the most fundamental problems in tracking known as drifting. Another direction is cross-domain visual instance search. Nowadays, images are recorded from various platforms, such as cell phones, street surveillance cameras, autonomous cars and drones, resulting a large domain gap between the recorded images. Methods that can perform cross-domain search will be in demand. Last, as another future work, can we exploit visual instance search and recognition for better visual categorization?

BIBLIOGRAPHY

[1] Large scale visual recognition challenge. `http://www.imagenet.org/challenges/LSVRC/2010`, 2010.

[2] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[3] E. Ahmed, M. Jones, and T. K. Marks. An improved deep learning architecture for person re-identification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[4] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for attribute-based classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[5] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2189–2202, 2012.

[6] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Word spotting and recognition with embedded attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(12):2552–2566, 2014.

[7] R. Arandjelović and A. Zisserman. Multiple queries for large scale specific object retrieval. In *Proceedings of British Machine Vision Conference*, 2012.

[8] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[9] R. Arandjelović and A. Zisserman. All about VLAD. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[10] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *Proceedings of European Conference on Computer Vision*, 2014.

[11] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[12] L. Bazzani, M. Cristani, and V. Murino. Symmetry-driven accumulation of local features for human characterization and re-identification. *Computer Vision and Image Understanding*, 117(2):130–144, 2013.

[13] A. Bedagkar-Gala and S. K. Shah. A survey of approaches and trends in person re-identification. *Image and Vision Computing*, 32(4):270–286, 2014.

[14] T. L. Berg, A. C. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *Proceedings of European Conference on Computer Vision*, 2010.

[15] K. L. Bouman, G. Abdollahian, M. Boutin, and E. J. Delp. A low complexity sign detection and text localization method for mobile applications. *IEEE Transactions on Multimedia*, 13(5):922–934, 2011.

[16] L. Breiman. Random forests. *Machine Learning*, 2001.

[17] K. Briechle and U. D. Hanebeck. Template matching using fast normalized cross correlation. In *Proceedings of SPIE: Optical Pattern Recognition XII*, 2001.

[18] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah. Signature verification using a "siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688, 1993.

[19] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2011.

[20] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[21] K. Chatfield, R. Arandjelović, O. Parkhi, and A. Zisserman. On-the-fly learning for visual search of large-scale image and video datasets. *International Journal of Multimedia Information Retrieval*, 4(2):75–93, 2015.

[22] D. Chen, Z. Yuan, G. Hua, N. Zheng, and J. Wang. Similarity learning on an explicit polynomial kernel feature map for person re-identification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[23] H. Chen, S. S. Tsai, G. Schroth, D. M. Chen, R. Grzeszczuk, and B. Girod. Robust text detection in natural images with edge-enhanced maximally stable extremal regions. In *Proceedings of IEEE International Conference on Image Processing*, 2011.

[24] Q. Chen, Z. Song, R. Feris, A. Datta, L. Cao, Z. Huang, and S. Yan. Efficient maximum appearance search for large-scale object detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[25] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[26] O. Chum, A. Mikulík, M. Perdoch, and J. Matas. Total recall II: query expansion revisited. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[27] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[28] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[29] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[30] J. Delhumeau, P.-H. Gosselin, H. Jégou, and P. Pérez. Revisiting the VLAD image representation. In *Proceedings of ACM International Conference on Multimedia*, 2013.

[31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[32] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[33] M. Douze, A. Ramisa, and C. Schmid. Combining attributes and fisher vectors for efficient image retrieval. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[34] R. O. Duda, P. E. Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

[35] I. Endres and D. Hoiem. Category independent object proposals. In *Proceedings of European Conference on Computer Vision*, 2010.

[36] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[37] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. `http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html`.

[38] I. Everts, J. C. van Gemert, and T. Gevers. Per-patch descriptor selection using surface and scene properties. In *Proceedings of European Conference on Computer Vision*, 2012.

[39] N. Ezaki, M. Bulacu, and L. Schomaker. Text detection from natural scene images: towards a system for visually impaired persons. In *Proceedings of International Conference on Pattern Recognition*, 2004.

[40] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[41] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[42] B. Fernando, S. Karaoglu, and A. Trémeau. Extreme value theory based text binarization in documents and natural scenes. In *Proceedings of International Conference on Machine Vision*, 2010.

[43] V. Ferrari and A. Zisserman. Learning visual attributes. In *Proceedings of Neural Information Processing Systems*, 2008.

[44] J. Gao, H. Ling, W. Hu, and J. Xing. Transfer learning based visual tracking with gaussian processes regression. In *Proceedings of European Conference on Computer Vision*, 2014.

[45] E. Gavves, B. Fernando, C. G. M. Snoek, A. W. M. Smeulders, and T. Tuytelaars. Fine-grained categorization by alignments. In *Proceedings of International Conference on Computer Vision*, 2013.

[46] E. Gavves, C. G. M. Snoek, and A. W. M. Smeulders. Visual synonyms for landmark image retrieval. *Computer Vision and Image Understanding*, 116(2):238–249, 2012.

[47] R. Girshick. Fast r-cnn. In *Proceedings of International Conference on Computer Vision*, 2015.

[48] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[49] M. Godec, P. M. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. In *Proceedings of International Conference on Computer Vision*, 2011.

[50] V. Goel, A. Mishra, K. Alahari, and C. V. Jawahar. Whole is greater than sum of parts: Recognizing scene text words. In *Proceedings of International Conference on Document Analysis and Recognition*, 2013.

[51] L. Gomez-Bigorda and D. Karatzas. Textproposals: a text-specific selective search algorithm for word spotting in the wild. *arXiv preprint arXiv:1604.02619*, 2016.

[52] S. Gong, M. Cristani, S. Yan, and C. C. Loy. *Person re-identification*, volume 1. Springer, 2014.

[53] A. Gordoa, J. A. Rodríguez-Serrano, F. Perronnin, and E. Valveny. Leveraging category-level labels for instance-level image retrieval. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[54] D. Gray, S. Brennan, and H. Tao. Evaluating appearance models for recognition, reacquisition, and tracking. In *IEEE International Workshop on Performance Evaluation for Tracking and Surveillance*, 2007.

[55] D. Gray and H. Tao. Viewpoint invariant pedestrian recognition with an ensemble of localized features. In *Proceedings of European Conference on Computer Vision*. 2008.

[56] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[57] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. In *Proceedings of International Conference on Computer Vision*, 2011.

[58] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[59] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *Proceedings of International Conference on Computer Vision*, 2009.

[60] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[61] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[62] M. Hirzer, P. M. Roth, M. Köstinger, and H. Bischof. Relaxed pairwise learned metric for person re-identification. In *Proceedings of European Conference on Computer Vision*, 2012.

[63] S. Holzer, S. Ilic, D. J. Tan, M. Pollefeys, and N. Navab. Efficient learning of linear predictors for template tracking. *International Journal of Computer Vision*, 2015.

[64] S. Hong, T. You, S. Kwak, and B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. In *Proceedings of International Conference on Machine Learning*, 2015.

[65] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[66] Y. Hua, K. Alahari, and C. Schmid. Occlusion and motion reasoning for long-term tracking. In *Proceedings of European Conference on Computer Vision*. Springer, 2014.

[67] D. Huang, L. Luo, M. Wen, Z. Chen, and C. Zhang. Enable scale and aspect ratio adaptability in visual tracking with detection proposals. In *Proceedings of British Machine Vision Conference*, 2015.

[68] J. Huang, S. Liu, J. Xing, T. Mei, and S. Yan. Circle & search: Attribute-aware shoe retrieval. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11(1):3, 2014.

[69] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Reading text in the wild with convolutional neural networks. *arXiv preprint arXiv:1412.1842*, 2014.

[70] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014.

[71] M. Jaderberg, A. Vedaldi, and A. Zisserman. Deep features for text spotting. In *Proceedings of European Conference on Computer Vision*, 2014.

[72] H. Jégou and O. Chum. Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In *Proceedings of European Conference on Computer Vision*, 2012.

[73] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proceedings of European Conference on Computer Vision*, 2008.

[74] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

[75] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[76] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1704–1716, 2012.

[77] H. Jégou and A. Zisserman. Triangulation embedding and democratic aggregation for image search. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[78] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[79] Y. Jiang, J. Meng, and J. Yuan. Randomized visual phrases for object search. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[80] A. Joly and O. Buisson. Logo retrieval with a contrario visual query expansion. In *Proceedings of ACM International Conference on Multimedia*, 2009.

[81] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *Proceedings of International Conference on Computer Vision*, 2009.

[82] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1409–1422, 2012.

[83] S. Karaoglu, R. Tao, T. Gevers, and A. W. M. Smeulders. Words matter: Scene text for image classification and retrieval. *IEEE Transactions on Multimedia (under review)*, 2016.

[84] S. Karaoglu, J. C. van Gemert, and T. Gevers. Object reading: Text recognition for object recognition. In *ECCV Workshops and Demonstrations*, 2012.

[85] S. Karaoglu, J. C. van Gemert, and T. Gevers. Con-text: Text detection using background connectivity for fine-grained object classification. In *Proceedings of ACM International Conference on Multimedia*, 2013.

[86] A. Kovashka, D. Parikh, and K. Grauman. Whittlesearch: Image search with relative attribute feedback. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[87] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of Neural Information Processing Systems*, 2012.

[88] C. H. Lampert. Detecting objects in large image collections and videos by efficient subimage retrieval. In *Proceedings of International Conference on Computer Vision*, 2009.

[89] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[90] H. Li, Y. Li, F. Porikli, et al. Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking. In *Proceedings of British Machine Vision Conference*, 2014.

[91] X. Li, C. G. M. Snoek, and M. Worring. Learning social tag relevance by neighbor voting. *IEEE Transactions on Multimedia*, 11(7):1310–1322, 2009.

[92] Y. Li, W. Jia, C. Shen, and A. van den Hengel. Characterness: an indicator of text in the wild. *IEEE Transactions on Image Processing*, 23(4):1666–1677, 2014.

[93] Z. Li, E. Gavves, K. E. van de Sande, C. G. M. Snoek, and A. W. M. Smeulders. Codemaps-segment, classify and search objects locally. In *Proceedings of International Conference on Computer Vision*, 2013.

[94] S. Liao, Y. Hu, X. Zhu, and S. Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[95] T.-Y. Lin, Y. Cui, S. Belongie, J. Hays, and C. Tech. Learning deep representations for ground-to-aerial geolocalization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[96] Z. Lin and J. Brandt. A local bag-of-features model for large-scale object retrieval. In *Proceedings of European Conference on Computer Vision*, 2010.

[97] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur. Dog breed classification using part localization. In *Proceedings of European Conference on Computer Vision*, 2012.

[98] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[99] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[100] S. Lu, T. Chen, S. Tian, J.-H. Lim, and C.-L. Tan. Scene text extraction based on edges and support vector regression. *International Journal on Document Analysis and Recognition*, 18(2), 2015.

[101] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of International Joint Conference on Artificial Intelligence*, 1981.

[102] B. Ma, Y. Su, and F. Jurie. Local descriptors encoded by fisher vectors for person re-identification. In *European Conference on Computer Vision workshops*, 2012.

[103] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

[104] J. Mao, H. Li, W. Zhou, S. Yan, and Q. Tian. Scale based region growing for scene text detection. In *Proceedings of ACM International Conference on Multimedia*, 2013.

[105] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of British Machine Vision Conference*, 2002.

[106] X. Mei and H. Ling. Robust visual tracking using l1 minimization. In *Proceedings of International Conference on Computer Vision*, 2009.

[107] T. Mensink, E. Gavves, and C. G. M. Snoek. Costa: Co-occurrence statistics for zero-shot classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[108] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[109] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[110] A. Mikulík, M. Perdoch, O. Chum, and J. Matas. Learning a fine vocabulary. In *Proceedings of European Conference on Computer Vision*, 2010.

[111] A. Mishra, K. Alahari, and C. Jawahar. Image retrieval using textual cues. In *Proceedings of International Conference on Computer Vision*, 2013.

[112] A. Mishra, K. Alahari, and C. V. Jawahar. Scene text recognition using higher order language priors. In *Proceedings of British Machine Vision Conference*, 2012.

[113] A. Mishra, K. Alahari, and C. V. Jawahar. Top-down and bottom-up cues for scene text recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[114] Y. Movshovitz-Attias, Q. Yu, M. C. Stumpe, V. Shet, S. Arnoud, and L. Yatziv. Ontological supervision for fine grained classification of street view storefronts. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[115] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of International Conference on Machine Learning*, 2010.

[116] M. Naphade, J. R. Smith, J. Tesic, S.-F. Chang, W. Hsu, L. Kennedy, A. Hauptmann, and J. Curtis. Large-scale concept ontology for multimedia. *MultiMedia, IEEE*, 13(3):86–91, 2006.

[117] L. Neumann and J. Matas. A method for text localization and recognition in real-world images. In *Proceedings of Asian Conference on Computer Vision*. 2010.

[118] L. Neumann and J. Matas. Text localization in real-world images using efficiently pruned exhaustive search. In *Proceedings of International Conference on Document Analysis and Recognition*, 2011.

[119] L. Neumann and J. Matas. Real-time scene text localization and recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[120] J. Y.-H. Ng, F. Yang, and L. S. Davis. Exploiting local features from deep networks for image retrieval. *arXiv preprint arXiv:1504.05133*, 2015.

[121] H. T. Nguyen and A. W. M. Smeulders. Robust tracking using foreground-background texture discrimination. *International Journal of Computer Vision*, 2006.

[122] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

[123] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[124] T. Novikova, O. Barinova, P. Kohli, and V. Lempitsky. Large-lexicon attribute-consistent text recognition in natural images. In *Proceedings of European Conference on Computer Vision*, 2012.

[125] P. Over, G. Awad, J. Fiscus, G. Sanders, and B. Shaw. Trecvid 2012 - an introduction of the goals, tasks, data, evaluation mechanisms and metrics. In *The TREC Video Retrieval Evaluation (TRECVID)*, 2012.

[126] S. Paisitkriangkrai, C. Shen, and A. v. d. Hengel. Learning to rank in person re-identification with metric ensembles. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[127] J. Pan and B. Hu. Robust occlusion handling in object tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[128] M. Perdoch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[129] F. Pernici and A. D. Bimbo. Object tracking by oversampling local features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[130] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[131] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[132] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *Proceedings of European Conference on Computer Vision*, 2010.

[133] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[134] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[135] D. Qin, S. Gammeter, L. Bossard, T. Quack, and L. Van Gool. Hello neighbor: accurate object retrieval with k-reciprocal nearest neighbors. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[136] D. Qin, C. Wengert, and L. van Gool. Query adaptive similarity for large scale object retrieval. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[137] M. Rastegari, A. Farhadi, and D. Forsyth. Attribute discovery via predictable discriminative binary codes. In *Proceedings of European Conference on Computer Vision*, 2012.

[138] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[139] J. Revaud, M. Douze, and C. Schmid. Correlation-based burstiness for logo retrieval. In *Proceedings of ACM International Conference on Multimedia*, 2012.

[140] S. Romberg and R. Lienhart. Bundle min-hashing for logo recognition. In *Proceedings of International Conference on Multimedia Retrieval*, 2013.

[141] S. Romberg, L. G. Pueyo, R. Lienhart, and R. Van Zwol. Scalable logo recognition in real-world images. In *Proceedings of International Conference on Multimedia Retrieval*, 2011.

[142] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 2008.

[143] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[144] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek. Image classification with the fisher vector: theory and practice. *International Journal of Computer Vision*, 105(3):222–245, 2013.

[145] J. L. Schonberger, F. Radenovic, O. Chum, and J.-M. Frahm. From single image query to detailed 3d reconstruction. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[146] V. Sharmanska, N. Quadrianto, and C. H. Lampert. Augmented attribute representations. In *Proceedings of European Conference on Computer Vision*, 2012.

[147] X. Shen, Z. Lin, J. Brandt, and Y. Wu. Mobile product image search by automatic query object extraction. In *Proceedings of European Conference on Computer Vision*, 2012.

[148] Z. Shi, T. M. Hospedales, and T. Xiang. Transferring a semantic representation for person re-identification and search. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[149] B. Siddiquie, R. S. Feris, and L. S. Davis. Image ranking and retrieval based on multi-attribute queries. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[150] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of International Conference on Learning Representations*, 2015.

[151] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proceedings of International Conference on Computer Vision*, 2003.

[152] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: an experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2014.

[153] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[154] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[155] R. Tao, E. Gavves, and A. W. M. Smeulders. Siamese instance search for tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[156] R. Tao, E. Gavves, C. G. M. Snoek, and A. W. M. Smeulders. Locality in generic instance search from one example. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[157] R. Tao, A. W. M. Smeulders, and S.-F. Chang. Attributes and categories for generic instance search from one example. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[158] G. Tolias, Y. Avrithis, and H. Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *Proceedings of International Conference on Computer Vision*, 2013.

[159] G. Tolias, Y. Avrithis, and H. Jégou. Image search with selective match kernels: aggregation across single and multiple images. *International Journal of Computer Vision*, 2015.

[160] G. Tolias, Y. Kalantidis, and Y. Avrithis. Symcity: feature selection by symmetry for large scale image retrieval. In *Proceedings of ACM International Conference on Multimedia*, 2012.

[161] J. R. R. Uijlings, A. W. M. Smeulders, and R. J. H. Scha. The visual extent of an object: suppose we know the object locations. *International Journal of Computer Vision*, 96(1):46–63, 2012.

[162] J. R. R. Uijlings, K. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[163] K. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.

[164] J. Van de Weijer, T. Gevers, and A. D. Bagdanov. Boosting color saliency in image feature detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):150–156, 2006.

[165] R. Vezzani, D. Baltieri, and R. Cucchiara. People reidentification in surveillance and forensics: A survey. *ACM Computing Surveys (CSUR)*, 46(2):29, 2013.

[166] L. Vincent. Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. *IEEE Transactions on Image Processing*, 2(2):176–201, 1993.

[167] U. Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[168] H.-C. Wang and M. Pomplun. The attraction of visual attention to texts in real-world scenes. *Journal of vision*, 12(6):26–26, 2012.

[169] K. Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *Proceedings of International Conference on Computer Vision*, 2011.

[170] N. Wang, S. Li, A. Gupta, and D.-Y. Yeung. Transferring rich feature hierarchies for robust visual tracking. *arXiv preprint arXiv:1501.04587*, 2015.

[171] N. Wang, J. Shi, D.-Y. Yeung, and J. Jia. Understanding and diagnosing visual tracking systems. In *Proceedings of International Conference on Computer Vision*, 2015.

[172] N. Wang and D.-Y. Yeung. Learning a deep compact image representation for visual tracking. In *Proceedings of Neural Information Processing Systems*, 2013.

[173] N. Wang and D.-Y. Yeung. Ensemble-based tracking: Aggregating crowdsourced structured time series data. In *Proceedings of International Conference on Machine Learning*, 2014.

[174] S. Wang, H. Lu, F. Yang, and M.-H. Yang. Superpixel tracking. In *Proceedings of International Conference on Computer Vision*, 2011.

[175] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng. End-to-end text recognition with convolutional neural networks. In *Proceedings of International Conference on Pattern Recognition*, 2012.

[176] X. Wang, M. Yang, T. Cour, S. Zhu, K. Yu, and T. X. Han. Contextual weighting for vocabulary tree based image retrieval. In *Proceedings of International Conference on Computer Vision*, 2011.

[177] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD birds 200. 2010.

[178] L. Wu, P. Shivakumara, T. Lu, and C. Tan. A new technique for multi-oriented scene text lines detection and tracking in video. *IEEE Transactions on Multimedia*, 2015.

[179] T. Wu, Y. Lu, and S. Zhu. Online object tracking, learning and parsing with and-or graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[180] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[181] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[182] Y. Yang, J. Yang, J. Yan, S. Liao, D. Yi, and S. Z. Li. Salient color names for person re-identification. In *Proceedings of European Conference on Computer Vision*. 2014.

[183] C. Yi and Y. Tian. Text string detection from natural scenes by structure-based partition and grouping. *IEEE Transactions on Image Processing*, 20(9):2594–2605, 2011.

[184] F. X. Yu, L. Cao, R. S. Feris, J. R. Smith, and S.-F. Chang. Designing category-level attributes for discriminative visual recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[185] F. X. Yu, R. Ji, M.-H. Tsai, G. Ye, and S.-F. Chang. Weak attributes for large-scale image retrieval. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[186] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[187] J. Žbontar and Y. LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[188] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Proceedings of European Conference on Computer Vision*, 2014.

[189] J. Zhang, S. Ma, and S. Sclaroff. Meem: Robust tracking via multiple experts using entropy minimization. In *Proceedings of European Conference on Computer Vision*, 2014.

[190] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *Proceedings of European Conference on Computer Vision*, 2014.

[191] S. Zhang, M. Yang, X. Wang, Y. Lin, and Q. Tian. Semantic-aware co-indexing for image retrieval. In *Proceedings of International Conference on Computer Vision*, 2013.

[192] Y. Zhang, X.-s. Wei, J. Wu, J. Cai, J. Lu, V.-A. Nguyen, and M. N. Do. Weakly supervised fine-grained image categorization. *arXiv preprint arXiv:1504.04943*, 2015.

[193] R. Zhao, W. Ouyang, and X. Wang. Unsupervised salience learning for person re-identification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[194] R. Zhao, W. Ouyang, and X. Wang. Learning mid-level filters for person re-identification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[195] L. Zheng, S. Wang, L. Tian, F. He, Z. Liu, and Q. Tian. Query-adaptive late fusion for image search and person re-identification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[196] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[197] C.-Z. Zhu, H. Jégou, and S. Satoh. Query-adaptive asymmetrical dissimilarities for visual object retrieval. In *Proceedings of International Conference on Computer Vision*, 2013.

[198] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *Proceedings of European Conference on Computer Vision*, 2014.

SAMENVATTING

HET ZOEKEN VAN EEN PLAATJE AAN DE HAND VAN ÉÉN VOORBEELD

Dit proefschrift behandelt het zoeken van plaatjes van een bepaald voorwerp of een bepaalde scene in een grote verzameling plaatjes aan de hand van een voorbeeldplaatje. Kunnen we alle plaatjes terugvinden die het voorbeeld uit het voorbeeldplaatje bevat?

Het voorwerp heeft vaak bepaalde afmetingen en bepaalde variaties, en dus ook de plaatjes daarvan. Daarom stelt het proefschrift voor om lokaliteit uit te buiten in het zoeken naar andere voorbeeldplaatjes van dat voorwerp. Lokalisatie in het plaatje helpt; het voorwerp heeft immers een bepaalde afmeting. Lokalisatie in de kenmerkruimte helpt; het voorwerp heeft immers een beperkte variatie in de verschijningsvormen. En lokalisatie in de gelijkheidsmaat helpt; het voorwerp lijkt immers met name op zichzelf. De combinatie van die drie factoren vermindert de verwarring met plaatjes van vergelijkbare voorwerpen, zodat het zoeken naar plaatjes met hetzelfde voorbeeld aanzienlijk beter gaat.

Voor 3D-voorwerpen werkt deze aanpak niet goed omdat de variatie hoe het voorwerp eruit ziet nu heel groot kan zijn. Daarom voor het zoeken van plaatjes van een 3D-voorwerp leren we eerst visuele attributen die specifiek zijn voor het soort object. Dus van schoenen leert de computer eerst automatisch wat belangrijk is in de beschrijving van een schoen: een hoge hak of een laars eruit ziet. De attributen zijn invariant ten opzichte van toevallige opname omstandigheden zoals het camerastandpunt en de lichtval. Daarna kan de computer beter verschillende objecten uit dezelfde categorie uit elkaar houden. De methode is effectief voor schoenen, auto's, en personen, als voorbeelden van 3D objecten met meerdere zijden. De methode van categorie-gebonden attributen helpt het zoeken naar 3D-voorbeelden aanzienlijk vooruit.

Als we ons specialiseren op het zoeken naar logo's dan is het van belang aandacht te besteden aan de tekst in het plaatje. We concentreren ons op een methode die eerst losse letters leest en vervolgens de waarschijnlijkheid van hele woorden schat. Dat verbetert de zoekresultaten van logo's aanzienlijk.

Een van de oudste problemen in digitale beeldbewerking is het volgen van een voorwerp over tijd. Dit *tracken* kan worden gezien als een speciale vorm van zoeken naar het voorbeeld, dit maal in de video. Het proefschrift stelt voor een *tracker* te maken die in elke plaatje van de video zoekt naar het meest gelijkende voorbeeld en verder niks. De kern is een krachtige vergelijkingsfunctie, met een Siamees netwerk geleerd uit een tweetal plaatjes genomen uit heel veel voorbeeldvideo's. Van elk tweetal was bekend of ze hetzelfde voorbeeld of twee verschillende voorbeelden weergeven. Dat leidt tot een *tracker* die heel simpel is en zo goed is dat het tot de beste *trackers* van dit moment behoort.

ACKNOWLEDGEMENTS

Pursuing the Ph.D. degree has been a truly challenging journey. Fortunately, I was not alone. I am so grateful to all the people who have inspired and helped me throughout these years. Without their help and care, this thesis would not have been possible.

My deepest gratitude goes to my supervisor and promotor, Arnold Smeulders, for his guidance, trust, motivation, and immense knowledge. His guidance helped in all the time of my Ph.D. study. In these years, I have learned from Arnold how to approach a problem as a scientist, how to make plans, how to present the work, how to write a rebuttal, and so many things I could not possibly list. When I was too focused on work, he was the person who reminded me that research was just one of the many dimensions of my life. Besides his great guidance, what I appreciate most is his trust. He has always believed in me, even at the moments when I had doubts in myself. Arnold, it is a great pleasure and honor for me to have you as my advisor.

I would like to thank Cees Snoek. I received enormous advice and support from Cees during my study. Cees, thank you for always being happy to listen to my problems and give helpful advice. I also want to express my gratitude to Theo Gevers for honoring me with being my co-promotor, for his advice and help in finalizing my thesis. I am grateful to Shih-Fu Chang, whom I had the opportunity to work with in Columbia University for two months, for all the discussions and guidance on my research.

Special thanks goes to Stratis Gavves. I enjoyed talking with Stratis very much as he can always come up with nice ideas, sometimes even without noticing that himself. This inspired me a lot for my study. It is a great pleasure to work with him. His advice and endless optimism have helped me tremendously.

I would like to thank my committee, Prof. dr. Rita Cucchiara, Prof. dr. Max Welling, Prof. dr. ir. Frans Groen, Dr. Cees Snoek, and Dr. Efstratios Gavves for honoring me with reading my thesis and participating in my defense.

Next I would like to thank all my other colleagues at UvA: Amir, Agni, Berkay, Changyong, Dennis, Dung, Fares, Felice, Gosia, Hamdi, Hendrik, Honza, Ivo, Jan, Jasper, Jorn, Kandan, Kirill, Koen, Lana, Marcel, Masoud, Matthias, Mihir, Mijung, Morris, Nour, Pascal, Peter, Sezer, Shuai, Silvia, Spencer, Stevan, Thomas, Tom, Virginie, Xirong, Yang, Zhongyu, Zhenyang. It is a great pleasure of being part of the big family, and having you in my life. I am so grateful to everyone in this family for giving me such a nice time in the past few years. A big thanks goes to Virginie for always being very helpful, and Dennis who has been supporting me throughout my research. Silvia, it is a pleasure to defend with you on the same day and thank you for taking the lead in making the final arrangements. Thanks to Lana, Zhenyang, Masoud, Zhaochun, Sezer and Dina for being great friends to me. And a special thank you to Sezer and Dina for your care and support, especially during my bad days.

最后，我要感谢我的父母，谢谢你们无私的爱和一直以来的支持。感谢我的叔叔，感谢你多年的指引和鼓励。感谢我的妻子，季烨，感谢你一路的陪伴。