SIXray: A Large-scale Security Inspection X-ray Benchmark for Prohibited Item Discovery in Overlapping Images

Caijing Miao[†], Lingxi Xie[‡♮], Fang Wan[†], Chi Su[♯], Hongye Liu[♯], Jianbin Jiao[†], Qixiang Ye[†§*]
[†]University of Chinese Academy of Sciences [‡]The Johns Hopkins University [♯]Kingsoft Cloud
[♮]Noah's Ark Lab, Huawei Inc. [§]Peng Cheng Laboratory
{miaocaijing16, wanfang13}@mails.ucas.ac.cn, 198808xc@gmail.com
{suchi, liuhongye}@kingsoft.com, {jiaojb, qxye}@ucas.ac.cn

Abstract

In this paper, we present a large-scale dataset and establish a baseline for prohibited item discovery in Security Inspection X-ray images. Our dataset, named SIXray, consists of 1,059,231 X-ray images, in which 6 classes of 8,929 prohibited items are manually annotated. It raises a brand new challenge of overlapping image data, meanwhile shares the same properties with existing datasets, including complex yet meaningless contexts and class imbalance. We propose an approach named class-balanced hierarchical refinement (CHR) to deal with these difficulties. CHR assumes that each input image is sampled from a mixture distribution, and that deep networks require an iterative process to infer image contents accurately. To accelerate, we insert reversed connections to different network backbones, delivering high-level visual cues to assist mid-level features. In addition, a class-balanced loss function is designed to maximally alleviate the noise introduced by easy negative samples. We evaluate CHR on SIXray with different ratios of positive/negative samples. Compared to the baselines, CHR enjoys a better ability of discriminating objects especially using mid-level features, which offers the possibility of using a weakly-supervised approach towards accurate object localization. In particular, the advantage of CHR is more significant in the scenarios with fewer positive training samples, which demonstrates its potential application in real-world security inspection.[1]

1. Introduction

Security inspection has been playing a critical role in protecting public space from safety threatening such as terrorism. With the growth of population in large cities and crowd density in public transportation hubs, it becomes more and more important to fast, automatically and accurately recognize prohibited items in X-ray scanned images. Recent years, the rapid development of deep learning [19] in particular convolutional neural networks has brought an evolution to image processing and visual understanding, including discovering and recognizing objects in X-ray images [23] [27] [24]. Different from natural images and other X-ray scans [36], security inspection often deals with a baggage or suitcase where objects are randomly stacked and heavily overlapped with each other. Therefore, in the scanned images, the objects of interest may be mixed with arbitrary and meaningless clutters and thus can be ignored even by human inspectors, Fig. 1.

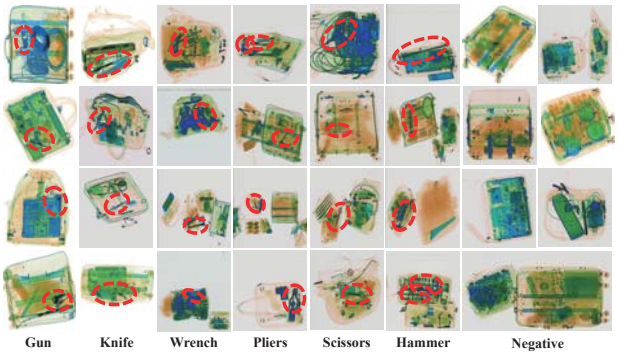

Figure 1. Example images in the presented SIXray dataset with six categories of prohibited items. Challenges include large variety in object scale and viewpoint, object overlapping and complex backgrounds (please zoom in for details).

To provide a public benchmark for research in this field, in this paper, we present a dataset named Security Inspection X-ray (SIXray), which is 100 times larger than the existing largest image collection for prohibited item discovery, *i.e.*, the *baggage* group in the GDXray dataset [25]. SIXray contains more than one million X-ray images in which only less than 1% images have positive labels (*i.e.*, prohibited items are annotated). It mimics a similar testing environment to the real-world scenarios where inspectors often aim

*Qixiang Ye is the corresponding author.
[1]Data and code: https://github.com/MeioJane/SIXray.

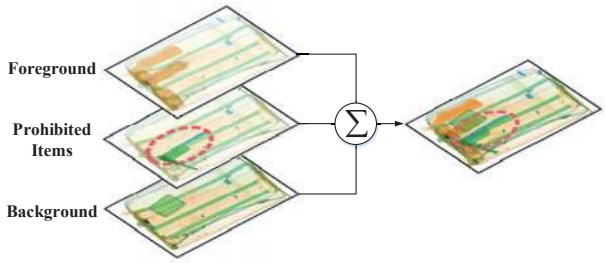

Figure 2. An X-ray image is composed of a set of overlapping images, each of which is transparent. (Best viewed in color).

at recognizing prohibited items appearing in a very low frequency (*e.g.*, 1 in 1,000). Unlike GDXray which only contains grayscale images in simple backgrounds, our dataset is much more challenging. Although a color-X-ray scanner assigns various colors to different materials, objects in the containers often suffer a considerable variety in scale, viewpoint, and style, yet a prohibited item may be mixed and overlapped with arbitrary numbers and types of safe items, as shown in Fig. 1.

We formulate this problem into an optimization task which, provided a dataset $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n^*)\}_{n=1}^N$, aims at minimizing the expected loss function between ground-truth and prediction $|\mathbf{y}_n^* - \mathbf{f}(\mathbf{x}_n; \boldsymbol{\theta})|^2$. Here $\mathbf{x}_n$ denotes image data and $\mathbf{y}_n^*$ is a C-dimensional vector with each index indicating whether a specific class is present in $\mathbf{x}_n$. Based on this framework, we point out a clear difference between natural images and X-ray images. A natural image $\mathbf{x}_n$ can always be divided into regions and each of them contains only one class c_n, thus can be sampled from a distribution $\mathcal{P}(\mathbf{x} \mid c_n)$. However, an X-ray image is often composed of a set of overlapping images which, provided a multi-class label $\mathbf{y}_n^*$ (C dimensions), can be formulated using a mixture distribution $\mathbf{x}_n = \sum_c y_{n,c}^* \cdot \mathbf{x}_{n,c}$ where $\mathbf{x}_{n,c}$ is sampled from a hidden distribution $\mathcal{P}(\mathbf{x} \mid c)$, as shown in Fig. 3.

We present an approach in the context of deep neural networks to deal with this complex scenario. The key idea is to combine two sources of information, namely, using mid-level features $\mathbf{x}_n$ (most often sampled from a mixture distribution) to determine high-level semantics $\mathbf{y}_n$, and reversely filtering irrelevant information out of $\mathbf{x}_n$ by referring to the information contained in $\mathbf{y}_n$. To this end, we formulate the high-level supervision signals into reversed network connections. To alleviate data imbalance, we introduce a loss-balancing term based on this hierarchy. This leads to the complete pipeline named class-balanced hierarchical refinement (CHR). With $\mathbf{y}_n$ being unobserved, an iterative process is required in optimization, which is computationally expensive in practice. To accelerate, we switch off iteration so that more training data are processed in a unit time period. In testing, CHR fuses visual information from different stages towards higher recognition accuracy, yet remains efficient in computation.

We evaluate CHR on the SIXray with different ratios of positive/negative samples[2]. CHR reports significantly higher classification performance over various baselines, *i.e.*, different network backbones, demonstrating the effectiveness of using high-level cues to assist mid-level features. In addition, we verify the necessity of adding the class-balanced loss term as we observe more significant improvement on less balanced training data. Last but not least, we provide annotations of prohibited items at the bounding box level in the testing set, and apply the class activation mapping (CAM) algorithm [39] as a baseline for weakly-supervised object localization.

The major contributions of this work are two-fold. (1) We provide a benchmark for future research in this challenging vision task. (2) We present an approach named CHR, which integrates multi-level visual cues and achieves class balance in the hierarchical structure.

2. Related Work

- **X-ray Images and Benchmarks**

X-ray images are captured by irradiating the objects with X-ray and rendering them with pseudo colors according to their spectral absorption rates. Therefore, in X-ray images, objects made of the same material are assigned with very similar colors, *e.g.*, metals are often shown in blue while impenetrable objects are often shown in red. Besides, the most significant difference between X-ray and natural images lies in object overlapping, because X-ray is often applied in the scenarios that some objects may heavily occlude others, *e.g.*, in a *baggage*, personal items are often stacked randomly. This property brings a new challenge to computer vision algorithms, while the traditional difficulties persist, *e.g.*, scale and viewpoint variance, intra-class variance and inter-class similarity, *etc.*, as widely observed in other object localization benchmarks like PascalVOC [9] and MS-COCO [21].

Researchers designed much work to deal with these difficulties and also approach the promising commercial value after them [1] [10] [26] [30] [34]. But unfortunately, very few X-ray datasets have been published for research purposes. A recently released benchmark, GDXray [25], contains three major categories of prohibited items including *gun*, *shuriken* and *razor blade*. However, images in GDXray were provided with few background clutters as well as overlaps, thus, it becomes considerably easy to recognizing these images and/or detecting the objects within. In addition, the relatively small number of negative samples (images not containing prohibited items) ease the algorithm in both training and testing stages. ChestXray8 [36] is a large-scale chest X-ray corpus for medical imaging analy-

[2]Throughout this paper, images with at least one prohibited item are called "positive", otherwise called "negative".

sis. Different from our scenario, objects in these images are rarely overlapping with each other.

- **X-ray Images and Benchmarks**

The research field of object recognition has been dominated by deep learning approaches. With the availability of large-scale datasets [18] and powerful computational resources, researchers are able to design and optimize very deep neural networks [18] [31] [16] [4] [13] [14] to learn visual patterns in a hierarchical manner. In the scenario that each image may contain more than one objects, there are typically two types of localization methods. The first one worked on the image level which produces a score for each class indicating its presence or absence [39]. The second one instead worked on the object level, and produced a bounding box as well as a class label for each object individually [12] [11] [29] [22] [28]. The former type often encounters the issues of multi-object classification and training data imbalance [36], for which binary cross entropy (BCE) loss [5] as well as class-balancing techniques [36] [15] were explored. The second type, on the other hand, was often based on a pipeline that first extracts a number of proposals in the image [12] [11] [29], and then determines the class of each proposal.

This paper studies image-level recognition, as per-object annotation is missing for training data, while our approach has the ability of object-level localization. This is related to the research in weakly-supervised object localization [3] [6] [33], or a series of work in localizing objects using top-down class activation [8] [7] [40]. There were also efforts about formulating the object localization in multiple instance learning frameworks where convolutional filters behave as detectors which activate regions of interest on the feature maps [3] [37] [33] [35].

In the context of object recognition in X-ray images, researchers realized that these images often contain fewer texture information, yet shape information stands out to be more discriminative. Therefore, in the era of bag-of-visual-word models [34] [2], the topic of designing effective and efficient handcrafted features is explored in depth [30] [26]. As deep learning becomes a standard tool of optimizing complex functions, researchers started to apply it to either extracting compact visual features for X-ray image representation [1] or fine-tuning a pre-trained model on X-ray images so that knowledge learned from natural images can be borrowed. This paper mainly focuses on the second approach.

3. The SIXray Benchmark

3.1. Data Acquisition

We collected a dataset named Security Inspection X-ray (SIXray), which contains a total of 1,059,231 X-ray images, and is more than 100 times larger than the only existing public dataset for the same purpose, *i.e.*, the *baggage* group of the GDXray dataset [25]. These images were collected from several subway stations with the original meta-data indicating the presence or absence of prohibited items. There are six common categories of prohibited items, namely, *gun*, *knife*, *wrench*, *pliers*, *scissors*, and *hammer*. The *hammer* class with merely 60 samples is not used in our experiments.

The SIXray Dataset (1,059,231)						
Positive (8,929)						Negative
Gun	Knife	Wrench	Pliers	Scissors	Hammer	
3,131	1,943	2,199	3,961	983	60	1,050,302

Table 1. The class distribution of the SIXray dataset. There is another *hammer* class with 60 items, but it is not used due to the small number of samples.

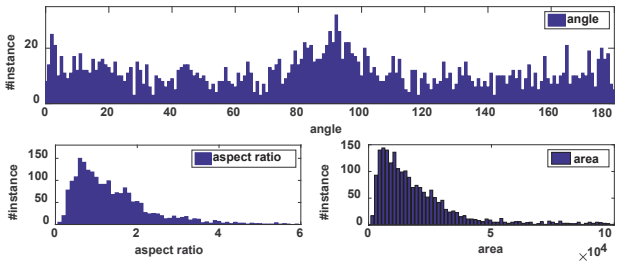

Figure 3. Distributions of object view angle, aspect ratio and area in the SIXray test set.

The distribution of these objects aligns with the real-world scenario, in which there are much fewer positive samples compared to negative samples. A statistics on this dataset is shown in Tab. 1. Each image was scaned by security inspection machine , which assigned different colors to objects made of different materials. All images were stored in JPEG format with an average size of 100K pixels.

To study the impact brought by training data imbalance, we construct three subsets of this dataset, and name them as SIXray10, SIXray100 and SIXray1000, respectively, with the number indicating the ratio of negative samples over positive samples. In SIXray10 and SIXray100, all 8,929 positive images are included, and there are exactly $10\times$ and $100\times$ negative images. SIXray100 has a very close distribution to the real world scenario. To maximally explore the ability of our algorithm to deal with data imbalance, we construct the SIXray1000 dataset by randomly choosing only 1,000 positive images each class but mixing them with all the 1,050,302 negative images. Each subset is further partitioned into a training set and a testing set, with the former containing 80% of the images and the latter containing 20% (the ratio training/testing images is $4:1$).

On the entire dataset, we use the image-level annotations provided by human security inspectors, *i.e.*, whether each type of prohibited items is present. In addition, on the *test* sets, we manually add a bounding-box for each prohibited item to evaluate the performance of object localization.

3.2. Dataset Properties

The SIXray dataset has several properties which bring difficulties to visual recognition. **First**, these images were mostly obtained from X-ray scans on personal luggage, *e.g.*, *bags* or *suitcases*, in which objects are often randomly stacked. When these items passed an X-ray scan, the penetration property makes it possible to see even the occluded items in the image. This leads to the most important property of this dataset, which we call it *overlapping*. **Second**, prohibited items can appear in many different scales, viewpoints, styles and even subtypes, all of which cause considerable intra-class variation and increase the difficulty of recognition, Fig.3. **Third**, the images can be heavily cluttered yet it is almost impossible to assign all objects especially those non-prohibited ones with a clear class label. Thus, there is noise coming from an open set of objects, which makes it difficult to expect what appears in the background regions. **Fourth and last**, as mentioned above, the positive images (with at least one prohibited item) only occupy a small fraction of this dataset. Without a special treatment, it is easy for the training stage to bias towards the negative class, as simply guessing a negative label yields sufficiently high accuracy. This raises a challenge to training stability.

In the following section, we present our approach which takes these properties into consideration, especially the first and fourth properties which are specific to this dataset.

4. Our Approach

4.1. Motivation and Formulation

As observed in the previous section, a significant characteristic of X-ray images lies in that objects are overlapped with each other. Note that overlapping is different from occlusion in which the rear object is invisible. Instead, as X-ray is penetrable, both front and rear objects are visible in the image. This is named the *penetration assumption*, based on which we use a mixture model to formulate these data.

Let there be C classes of possible items appearing in the dataset, with an index set of $\{1, 2, \ldots, C\}$. Among them, C' classes are considered prohibited, *e.g.*, in the SIXray dataset, $C' = 5$. Without loss of generality, we assign them with the class index of $1, 2, \ldots, C'$. Let the dataset $\mathcal{D}$ contain N images. For each input image $\mathbf{x}_n$, our goal is to obtain a C-dimensional vector $\mathbf{y}_n$ for each $\mathbf{x}_n$, each dimension in which, $y_{n,c}$, is either 0 or 1, with 1 indicating the specified prohibited item is present in this image and 0 vice versa. Note that the ground-truth of $\mathbf{y}_n^\star$ only exists for the first C' dimensions, while others remain unobserved.

To obtain a mathematical formulation of $\mathbf{x}_n$, we assume that it is composed of C sub-images $\mathbf{x}_{n,c}$, each of which corresponds to a specified class c and is sampled from a conditional distribution $\mathcal{P}_c \doteq \mathcal{P}(\mathbf{x} \mid c)$. Then, based on the *penetration assumption*, each image can be written as:

$$\mathbf{x}_n \approx \sum_{c=1}^{C} y_{n,c} \cdot \mathbf{x}_{n,c}, \quad \mathbf{x}_{n,c} \sim \mathcal{P}_c. \tag{1}$$

This formulation is of course not accurate as we ignore the overlapping relationship between objects as well as the order that objects are stacked, but it serves as an approximate formulation of how overlapping impacts image data.

Our goal is to learn a discriminative function $\mathbf{y}_n = \mathbf{f}(\mathbf{x}_n; \boldsymbol{\theta})$ to predict the image label. Since the object of interest may appear in various scales. In order to recognize and further detect it, a popular choice [17] [20] is to combine multi-stage visual information. Here we simply consider feature vectors extracted from L different layers, the l-th of which is denoted as $\mathbf{x}_n^{(l)}$. A regular solution is to train a classifier beyond each layer, $\mathbf{y}_n^{(l)} = \mathbf{h}^{(l)}\left(\mathbf{x}_n^{(l)}; \boldsymbol{\xi}^{(l)}\right)$, using the ground-truth signal $\mathbf{y}_n^\star$ as supervision. In the testing stage, we fuse all $\mathbf{y}_n^{(l)}$ as the final output, *i.e.*, $\mathbf{y}_n = \sum_{l=1}^{L} \mathbf{y}_n^{(l)}$.

However, we note a significant weakness of this model, which comes from the penetration assumption, *i.e.*, Eqn (1), applied to mid-level features[3]. This is to say, each $\mathbf{x}_n^{(l)}$ is the composition of sub-images sampled from different classes, including those items of no interest, and thus $\mathbf{h}^{(l)}\left(\mathbf{x}_n^{(l)}; \boldsymbol{\xi}^{(l)}\right)$ may be distracted. A reasonable idea is to refine $\mathbf{x}_n^{(l)}$ to get rid of these irrelevant information. This is achieved by a function $\mathbf{g}^{(l)}\left(\mathbf{x}_n^{(l)}, \mathbf{y}_n; \boldsymbol{\tau}^{(l)}\right)$, which shares the same dimensionality with $\mathbf{x}_n^{(l)}$. Summarizing these contents yields the following optimization problem:

$$\boldsymbol{\theta}^\star, \boldsymbol{\xi}^\star, \boldsymbol{\tau}^\star = \arg\min_{\boldsymbol{\theta}, \boldsymbol{\xi}, \boldsymbol{\tau}} \mathbb{E}_{\mathbf{x}_n \in \mathcal{D}} \sum_{l=1}^{L} \mathcal{L}_n^{(l)}, \quad \text{where} \tag{2}$$

$$\mathcal{L}_n^{(l)} = \mathcal{L}\left\{\mathbf{y}_n^\star, \mathbf{h}^{(l)}\left(\tilde{\mathbf{x}}_n^{(l)}; \boldsymbol{\xi}^{(l)}\right)\right\}, \tag{3}$$

$$\tilde{\mathbf{x}}_n^{(l)} = \mathbf{g}^{(l)}\left(\mathbf{x}_n^{(l)}, \mathbf{y}_n; \boldsymbol{\tau}^{(l)}\right), \quad \text{and} \tag{4}$$

$$\mathbf{y}_n = \frac{1}{L} \cdot \sum_{l=1}^{L} \mathbf{h}^{(l)}\left(\tilde{\mathbf{x}}_n^{(l)}; \boldsymbol{\xi}^{(l)}\right). \tag{5}$$

Here $\mathcal{L}\{\cdot, \cdot\}$ is a loss function which is discussed in details later. The above formulate define a recurrent model, in which $\mathbf{y}_n$ cannot be observed even in the training stage. The standard way of optimization involves iteration, in which we start with an $\mathbf{x}_n$ sampled from $\mathcal{D}$ and any $\mathbf{y}_n$ (in the training process, $\mathbf{y}_n$ always has C' dimensions). the first C' dimensions are provided by ground-truth and other $C - C'$

[3] Eqn (1) fits mid-level features best, because low-level features (*e.g.*, raw image pixels) are often largely impacted by small noise, in both case it is learning the class-conditional distribution $\mathcal{P}_c \doteq \mathcal{P}(\mathbf{x} \mid c)$ suffers a higher difficulty. Similarly, the very last layers (*e.g.*, containing class-specific *logits*) are less likely to be additive as in Eqn (1).

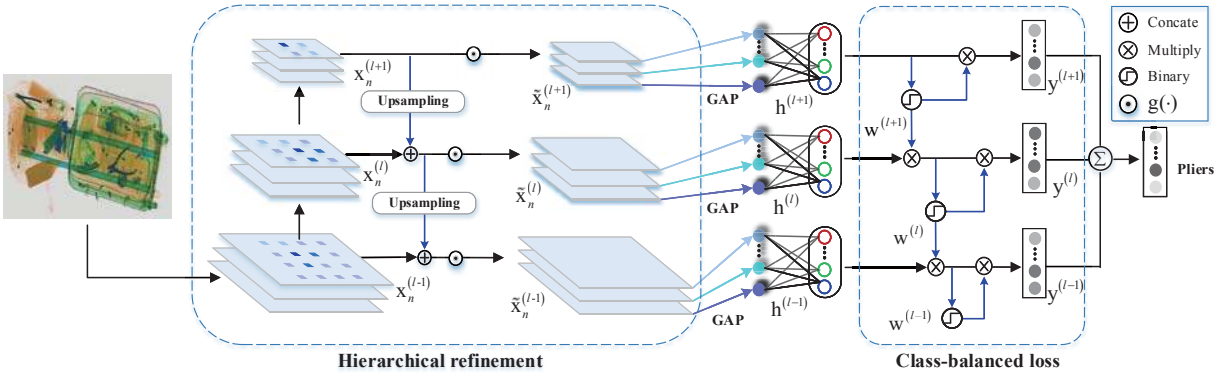

Figure 4. The overall architecture of the proposed class-balanced hierarchical refinement (CHR) approach (best viewed in color). The network backbone $\mathbf{f}(\mathbf{x}_n;\boldsymbol{\theta})$ is shown on the leftmost column, from which L layers are chosen as feature extractors. For simplicity, we show an example with $L=3$. Each $\mathbf{x}_n^{(l)}$, $l>1$, is up-sampled and concatenated with $\mathbf{x}_n^{(l-1)}$ and fed into a refinement function that simulates $\tilde{\mathbf{x}}_n^{(l-1)}=\mathbf{g}\left(\mathbf{x}_n^{(l-1)},\mathbf{x}_n^{(l)};\boldsymbol{\tau}^{(l-1)}\right)$, and $\tilde{\mathbf{x}}_n^{(l-1)}$ is sent into $\mathbf{h}^{(l-1)}\left(\tilde{\mathbf{x}}_n^{(l-1)};\boldsymbol{\xi}^{(l-1)}\right)$ for classification. GAP denotes global average pooling. A class-balancing loss is built upon the same hierarchy, on which mid-level negative samples are filtered out using high-level cues.

dimensions can be randomly initialized) We first compute $\mathbf{x}_n^{(l)}$ for each l accordingly, and use it to compute the first version of $\mathbf{y}_n^{(l)}=\mathbf{h}^{(l)}\left(\mathbf{x}_n^{(l)};\boldsymbol{\xi}^{(l)}\right)$. In each round, we compute $\mathbf{y}_n$ and use it to compute $\mathbf{g}^{(l)}\left(\mathbf{x}_n^{(l)},\mathbf{y}_n;\boldsymbol{\tau}^{(l)}\right)$ so that $\mathbf{x}_n^{(l)}$ is updated as $\tilde{\mathbf{x}}_n^{(l)}$. Within this process, parameters $\boldsymbol{\xi}^{(l)}$ and $\boldsymbol{\tau}^{(l)}$ are updated accordingly with ground-truth $\mathbf{y}_n^\star$ and gradient back-propagation. This iteration continues until convergence or a maximal number of rounds is achieved[4].

4.2. Approximation with Hierarchical Refinement

In practice, however, the above formulation has two major drawbacks. The first one lies in the inaccuracy of generative models. We expect a model $\mathbf{g}^{(l)}(\cdot)$ to eliminate the components in $\mathbf{x}_n^{(l)}$ that correspond to the non-targeted classes in $\mathbf{y}_n$. This is increasingly difficult especially when the $\mathbf{x}_n^{(l)}$ is far from $\mathbf{y}_n$. So, we assume that $\mathbf{x}_n^{(l)}$ only receives supervision signals from $\mathbf{x}_n^{(l+1)}$, which is much closer than $\mathbf{y}$, while $\mathbf{x}_n^{(l+1)}$ continues to receive information from $\mathbf{x}_n^{(l+2)}$ and this process continues until $\mathbf{y}_n$ is reached. In implementation, this implies that reversed connections only emerge between neighboring feature layers. Here an exception happens at the last feature layer, $\mathbf{x}_n^{(L)}$, which is connected to $\mathbf{y}_n$ via a classifier $\mathbf{h}^{(L)}(\cdot)$. Since direct supervisions have already been provided by this classifier, we ignore the connection between $\mathbf{y}_n$ and $\mathbf{x}_n^{(L)}$, leaving a total of $L-1$ connections between $\mathbf{x}_n^{(l)}$ and $\mathbf{x}_n^{(l+1)}$, for $l=1,2,\ldots,L-1$. This is to say, $\mathbf{g}\left(\mathbf{x}_n^{(l)},\mathbf{y}_n;\boldsymbol{\xi}^{(l)}\right)$ is replaced by $\mathbf{g}\left(\mathbf{x}_n^{(l)},\mathbf{x}_n^{(l+1)};\boldsymbol{\xi}^{(l)}\right)$. Nevertheless, $\mathbf{x}_n^{(l)}$ can still obtain supervision signals from $\mathbf{y}_n$ in an indirect manner, *i.e.*, via a few intermediate steps. This is named the hierarchical refinement strategy.

Implementation details are illustrated in Fig. 4. We start with $\tilde{\mathbf{x}}_n^{(L)}\equiv\mathbf{x}_n^{(L)}$, the feature extracted from the top layer. It is concatenated with the feature at the previous stage, $\mathbf{x}_n^{(L-1)}$, before which it is up-sampled if necessary. The concatenated feature is then fed into $\mathbf{g}^{(L-1)}\left(\mathbf{x}_n^{(L-1)},\mathbf{x}_n^{(L)};\boldsymbol{\tau}^{(L-1)}\right)$ to produce $\tilde{\mathbf{x}}_n^{(L-1)}$. This process continues until $\tilde{\mathbf{x}}_n^{(l)}$ is obtained. $\mathbf{g}^{(L-1)}(\cdot)$ means that up-sample $\mathbf{x}_n^{(L)}$ and feed it into a function with $\mathbf{x}_n^{(L-1)}$. Each $\tilde{\mathbf{x}}_n^{(l)}$, $l=1,2,\ldots,L$, is sent into the corresponding classifier $\mathbf{h}^{(l)}\left(\tilde{\mathbf{x}}_n^{(l)};\boldsymbol{\xi}^{(l)}\right)$ to obtain $\mathbf{y}_n^{(l)}$. All $\mathbf{y}_n^{(l)}$ are averaged into the final output and supervised by $\mathbf{y}^\star$.

The second drawback is the slowness of an iterative optimization. To accelerate, we switch off iteration so that each case $\mathbf{x}_n\in\mathcal{D}$ is forward-propagated and back-propagated only once, and the updated parameters $\boldsymbol{\theta}$, $\boldsymbol{\xi}^{(l)}$ and $\boldsymbol{\tau}^{(l)}$ are directly applied to another case sampled from $\mathcal{D}$. This can be understood as stochastic gradient descent on $\mathcal{D}$. In practice, this allows us to sample more data in the same period of time, and thus improve training efficiency.

4.3. Class-Balanced Loss

Here we study the impact of the loss function, *i.e.*, Eqn (3), in the training process. In this specific problem, *i.e.*, prohibited item discovery, there are much fewer positive training samples (at least one prohibited item is labeled) than negative ones. This makes regular loss functions such as the Euclidean loss $\mathcal{L}\{\mathbf{y}_n^\star,\mathbf{y}_n\}=$

[4]Here are some side notes. It has been widely believed that a deep network is able to fit training data sampled from one-class distributions, *e.g.*, each sample contains only one object in class c_n, so that $\mathbf{x}_n$ is sampled from $\mathcal{P}_{c_n}$. In such scenarios, $\mathbf{y}_n$ as a one-hot vector is relatively easy to estimate and thus iteration is not required. This is the reason that deep networks produced satisfying performance in the GDXray dataset [25] in which most images contain only one object.

$|\mathbf{y}_n^\star - \mathbf{y}_n|^2$ and the Binary Cross-Entropy (BCE) loss $\mathcal{L}\{\mathbf{y}_n^\star, \mathbf{y}_n\} = -\left[\mathbf{y}_n^{\star\top} \log \mathbf{y}_n + (1-\mathbf{y}_n^\star)^\top \log(1-\mathbf{y}_n)\right]$ less effective, because the network can heavily bias towards negative examples (because simply guessing all training samples to be negative leads to a very low loss function) and, consequently, the recall becomes considerably low. A reasonable solution is to slightly change the loss function so as to equivalently reduce the number of negative training data [36]. Here we combine this approach in the context of hierarchical refinement which once again takes advantage of high-level supervision to guide mid-level features.

The proposed loss function works in a mini-batch $\mathcal{B} \subset \mathcal{D}$. For each case $\mathbf{x}_n$ with $\mathbf{y}_n$, we have a few stages defined previously, each of which produces a feature $\mathbf{x}_n^{(l)}$ followed by a prediction $\mathbf{y}_n^{(l)}$. We add a binary weight vector, denoted by $\mathbf{w}_n^{(l)}$, measuring whether each class in $\mathbf{y}_n^{(l)}$ contributes to the loss function. Thus, Eqn (3) becomes:

$$\mathcal{L}_n^{(l)} = \mathbf{w}_n^{(l)\top} \cdot \mathbf{E}\left(\mathbf{y}_n^\star, \mathbf{y}_n^{(l)}\right), \tag{6}$$

where $\mathbf{E}\left(\mathbf{y}_n^\star, \mathbf{y}_n^{(l)}\right)$ is the loss vector, $\mathbf{E}\left(\mathbf{y}_n^\star, \mathbf{y}_n^{(l)}\right) = -\left[\mathbf{y}_n^\star \odot \log \mathbf{y}_n^{(l)} + (1-\mathbf{y}_n^\star) \odot \log\left(1-\mathbf{y}_n^{(l)}\right)\right]$, and $\odot$ denotes element-wise multiplication.

It remains to define $\mathbf{w}_n^{(l)}$ for each $\mathbf{y}_n^{(l)}$. In the highest (L-th) level, $\mathbf{w}_n^{(L)}$ directly measures whether each class, or each dimension in $\mathbf{y}_n^{(l)}$, has to be considered. This conditional variable is always true for each class with a positive label, while for that with a negative label, it is true only if the prediction is smaller than a fixed threshold ε. In each of the lower levels, a class is considered if the above judgment returns true, as well as all the higher levels support this – in other words, if a class is switched off at some level, it will never be considered in each of the lower levels. This is based on the assumption that high-level features are more reliable in determining which classes are present and which are absent, while low-level features may produce false positives due to various reasons.

Replacing Eqn (3) with Eqn (6) gives the complete class-balanced hierarchical refinement (CHR) approach. In the training process, each $\mathcal{L}_n^{(l)}$ is computed individually and averaged for gradient back-propagation. In the testing stage, all $\mathbf{y}_n^{(l)}$'s are averaged for prediction.

5. Experiments

5.1. Setting and Baselines

We use all three subsets, namely, SIXray10, SIXray100 and SIXray1000, to evaluate different approaches. In each subset, all models are optimized on 80% training data, and evaluated on the remaining 20% testing data. These data splits are random but consistent for all competitors.

We evaluate both image-level classification mean Average Precision and object-level localization accuracy, for the second goal we manually labeled all prohibited items with bounding-box in the testing images. For image classification, we apply the evaluation metric in the PascalVOC image classification task [9], which works on each class individually – all testing images are ranked by the confidence of containing the specified object, and the mean average precision (mAP) is computed. For object localization, we follow [38] to compute the accuracy of pointing localization. A hit is counted if the pixel of the maximum response falls within one of the ground-truth bounding-boxes of the specified object, otherwise a missed is counted. Thus, each class has a localization accuracy computed by $\frac{\#\text{Hits}}{\#\text{Hits}+\#\text{Misses}}$. For both tasks, we also report the overall performance which is the average over all five classes.

We investigate five popular backbones, including ResNets [13] with 34, 50 and 101 layers, Inception-v3 [32], and DenseNet with 121 layers. We follow the conventions to setup all these networks, and CHR is applied to each of them using $L = 3$ – three pooling layers with different spatial resolutions (*e.g.*, in ResNets, 28×28, 14×14, and 7×7) are used as features. It is of course possible to increase L by adding more features, yet in practice we find $L = 3$ is sufficient to provide complementary information.

5.2. Classification: Quantitative Results

We first investigate the overall (averaged over five classes) image classification results which are summarized in Tab. 2. CHR achieves consistent mean Average Precision gain beyond all network backbones as well as in all different subsets, *i.e.*, SIXray10, SIXray100 and SIXray1000. We observe that CHR works better in deeper networks, which is also observed in experiments, *e.g.*, on top of Inception-v3 and DenseNet, the absolute improvement over SIXRay1000 is 8.22% and 9.08%, respectively.

We next observe five types of objects individually. The benefit brought by CHR is different from class to class. Take the DenseNet as an example. When it is aimed at finding *gun*, classification performance is not boosted in all subsets, while we observe significant gains over all the other classes, especially for *scissors*, the accuracy is improved by an impressive amount of 30%. We can see in Table 1 that the training samples of *scissors* is the least among all five prohibited items, for which reason the baseline suffers significant bias in the training stage. CHR, by introducing hierarchical signals for supervision, largely alleviates this bias.

Finally, we study the issue of data imbalance over different subsets. Recall that the ratio of negative over positive images is 10, 100 and 1000, respectively. From Fig. 5, we can see that the performance gain goes up with data imbalance, which, as analyzed in Sec. 5.4, comes from our special treatment towards class balancing.

Method	Gun			Knife			Wrench			Pliers			Scissors			mean		
ResNet34 [13]	89.71	83.06	72.05	85.46	78.75	56.42	62.48	30.49	16.47	83.50	55.24	14.24	52.99	16.14	7.12	74.83	52.74	33.26
ResNet34+CHR	87.16	81.96	73.35	87.17	77.70	60.46	64.31	36.85	23.72	85.79	64.56	17.98	61.58	14.49	18.19	77.20	55.11	38.74
ResNet50 [13]	90.64	84.75	74.19	87.82	77.92	59.82	63.62	28.49	16.03	84.80	50.53	16.59	57.35	19.39	2.87	76.85	52.22	33.90
ResNet50+CHR	87.55	82.64	73.43	86.38	79.60	61.32	69.12	41.19	18.88	85.72	58.02	12.32	60.91	27.89	19.03	77.94	57.87	37.00
ResNet101 [13]	87.65	82.83	76.04	84.26	76.16	63.53	69.33	35.59	13.65	85.29	54.82	15.57	60.39	20.63	11.28	77.38	54.01	36.01
ResNet101+CHR	85.45	83.25	75.38	87.21	77.53	64.80	71.23	42.02	15.27	88.28	68.01	19.02	64.68	32.33	16.21	79.37	60.63	38.14
Inception-v3 [32]	90.05	81.18	75.52	83.80	77.28	56.33	68.11	32.47	24.01	84.45	66.89	16.75	58.66	22.63	20.72	77.01	56.09	38.67
Inception-v3+CHR	88.90	79.22	76.91	87.23	73.48	61.29	69.47	37.20	29.60	86.37	69.01	19.11	65.50	31.81	47.56	79.49	58.15	46.89
DenseNet [14]	87.36	83.23	75.00	87.71	77.24	65.55	64.15	37.72	23.57	87.63	62.69	18.09	59.95	24.89	14.18	77.36	57.15	39.28
DenseNet+CHR	87.05	82.06	74.87	85.89	78.75	71.23	70.47	43.22	29.79	88.34	66.75	21.57	66.07	28.80	44.27	79.56	59.92	48.36

Table 2. Classification mean average precision (%) on the subsets of SIXray (each cell, left to right: SIXray10, SIXray100, SIXray1000).

Method	Gun			Knife			Wrench			Pliers			Scissors			mean		
ResNet34 [13]	71.60	50.62	53.93	51.28	55.38	38.97	43.32	26.74	22.46	68.88	34.54	13.69	22.16	7.95	6.82	51.45	35.05	27.17
ResNet34+CHR	75.62	60.19	70.41	55.38	63.08	26.15	52.41	35.83	37.97	58.44	53.70	25.10	19.32	0.00	2.27	52.23	42.56	32.38
ResNet50 [13]	63.89	47.53	42.32	57.44	52.82	48.72	49.73	28.34	19.79	68.88	39.85	19.77	17.05	1.70	2.84	51.40	34.05	26.69
ResNet50+CHR	68.83	57.72	60.67	58.46	49.23	37.44	54.01	41.18	22.46	77.04	49.91	20.91	15.91	15.34	13.64	54.85	42.67	31.02
ResNet101 [13]	73.77	73.15	70.41	65.13	64.10	60.00	28.34	25.13	15.51	62.24	31.50	14.07	21.02	11.36	5.68	50.10	41.05	33.13
ResNet101+CHR	80.86	79.32	79.03	73.85	69.23	61.54	52.41	27.81	21.93	9.30	48.39	17.11	40.34	6.25	19.32	51.35	46.20	39.78
Inception-v3 [32]	79.94	64.81	71.16	75.38	65.64	52.31	59.36	40.11	7.49	59.58	32.83	18.63	40.34	26.14	1.70	62.92	45.91	30.26
Inception-v3+CHR	78.70	67.59	73.41	74.36	63.08	41.54	52.41	23.53	23.53	59.96	54.27	7.60	52.27	39.20	11.36	63.54	49.53	31.49
DenseNet [14]	74.38	71.60	58.05	71.28	62.05	56.92	59.89	24.60	26.20	71.54	55.60	20.53	35.23	9.66	11.36	62.46	44.70	34.61
DenseNet+CHR	79.01	78.40	76.78	76.92	62.56	57.95	59.36	41.71	39.04	72.49	63.76	39.92	40.34	5.11	5.68	65.62	50.31	43.87

Table 3. Localization accuracy (%) on the subsets of SIXray (each cell, left to right: SIXray10, SIXray100, SIXray1000).

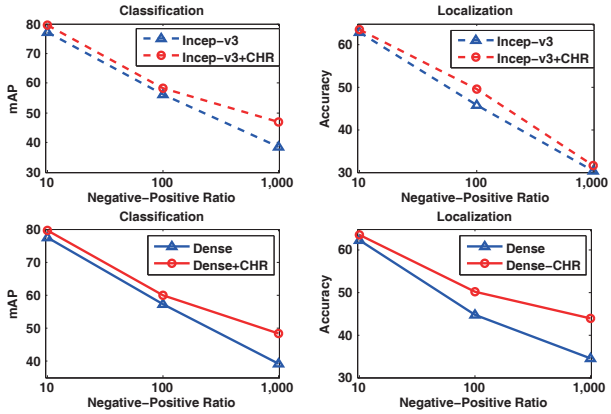

Figure 5. The overall accuracy gain of CHR becomes more significant in the subsets with larger negative-positive ratios.

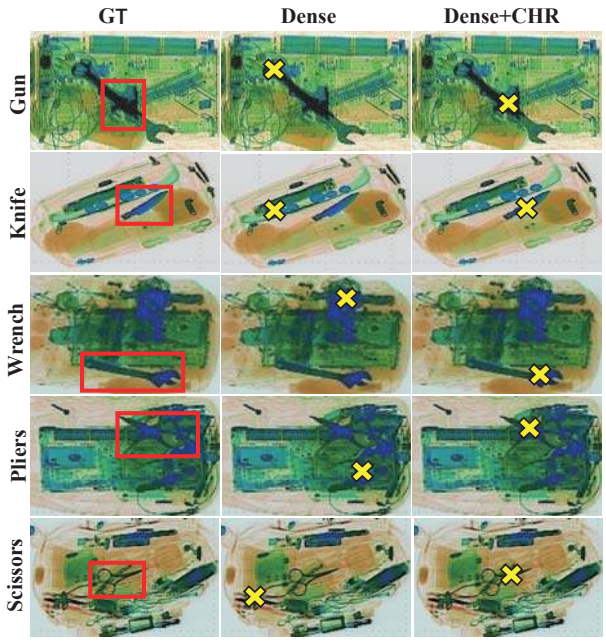

Figure 6. Examples of object localization based on DenseNet, Which shows CHR is effective in complex background and overlapping images. (best viewed in color).

5.3. Localization: Quantitative Results

To verify that CHR is not over-tuned to image classification, we attach the class activation map (CAM) [39], an weakly supervised approach for object localization, on top of the features extracted at different stages. CAM produces one heatmap for each class individually, and on each of these maps. We first rescale the maps to the original image size. If the maximal response across scales falls within one of the ground truth bounding boxes of the specified object, the predicted location is considered a valid localization.

Tab. 3 summarizes localization results. CHR based on DenseNet outperforms DenseNet by 5.61% (50.31% vs 44.70%) for SIXray100 and 9.26% (43.87% vs 34.61%) for SIXray1000. In particular, for the *wrench* class in SIXray1000, Inception-v3+CHR outperforms Inception-v3 by 16.04% (23.53% vs 7.49%). Again, we observe significant accuracy gain on deeper networks (which produces more powerful features) and larger negative-over-positive ratios. more localization results are shown in Fig. 6.

5.4. Ablation Studies

In this part, we investigate the impact brought by hierarchical refinement and class-balanced loss, respectively. All experiments are performed on three subsets of SIXray, which have different ratios of negative-positive samples. Results are summarized in Tab. 4.

First, we study the performance of hierarchical refinement, namely, the reversed connections in the network. It can be seen that the top-down refinement (ResNet34+HR) improves the classification and localization accuracies by about 1% and 6.52% on SIXray100, and 3.15% and 2.13% on SIXray1000. We note that ResNet34+HR outperforms ResNet34+H, the direct way incorporating hierarchical information, because the ResNet34+HR allows low-level features to be refined with high-level semantic cues.

Second, we study the impact of different loss functions. ResNet34+CH, with the designed class-balanced loss, improves classification and localization accuracies by 1.00% and 3.77% on SIXray100, and 3.09% and 3.44% on SIXray1000. By combining hierarchical refinement with the class-balance loss, ResNet34+CHR further improves the classification and localization accuracies by $+2.37\%$ and $+7.51\%$ on SIXray100, and $+5.48\%$ and $+5.11\%$ on SIXray1000, over the baseline. This shows the usefulness of class balance in imbalanced scenarios.

CHR achieves accuracy gain with a relatively small amount of extra computation. For example, ResNet34 requires 7.68ms to process each testing image and ResNet34-CHR requires 8.28ms, both tested on a Tesla V100 GPU. This is to say, CHR requires 7.81% extra time.

5.5. ILSVRC2012 Classification

Last but not least, we evaluate CHR on ILSVRC2012, a large-scale image classification dataset. This is to observe how CHR generalizes to natural image data, provided that it achieves significant accuracy gain on overlapping image data. ILSVRC2012 is a popular subset of the ImageNet databased, which has $1{,}000$ classes and each of them contains a well-defined concept in WordNet. There are 1.3M training images and 50K validation images, both of which are roughly uniformly distributed over all classes.

We follow the standard training and testing pipelines, including the policies of model initialization, data augmentation, learning rate decay, *etc*. Since ILSVRC2012 is not an imbalanced dataset, we switch off the weight terms in the loss function which was designed for this purpose. The top-1 error of CHR based on ResNet18 is 27.01% [13], which slightly lower than the baseline by 0.87% (27.01% vs 27.88%). Besides, the top-1 and top-5 errors of CHR based on ResNet50 [13] are 22.00% and 6.22%. which are lower than the baseline by 0.85% (22.00% vs 22.85%) and 0.49% (6.22% vs 6.71%), respectively. This slight but consistent accuracy gain delivers two-fold messages. The reversed connections in our approach which carries high-level supervision to mid-level features do not conflict with natural images – although it aligns with overlapping image data much better. Given that the additional computational costs are almost negligible, it is worth investigating its extension in the natural image domains.

Method	SIXray10		SIXray100		SIXray1000	
ResNet34	74.83	51.45	52.74	35.05	33.26	27.17
ResNet34+H	74.43	49.91	53.59	38.70	34.78	28.68
ResNet34+CH	76.28	48.01	54.59	42.47	37.87	32.12
ResNet34+HR	75.87	50.19	53.72	41.57	36.41	29.30
ResNet34+CHR	**77.20**	**52.23**	**55.11**	**42.56**	**38.74**	**32.38**

Table 4. Classification and localization accuracies ($\%$) on the SIXray subsets using different options (refinement method, loss function, *etc.*) of CHR. The backbone is ResNet34. For the explanation of different options, see the main texts in Sec. 5.4.

6. Conclusions

In this paper, we investigate prohibited item discovery in X-ray scanned images, which is a promising application in industry yet remains fewer studied in computer vision. To facilitate research in this field, we present SIXray, a large-scale dataset consisting of more than one million X-ray images, all of which were captured in real-world scenarios and therefore covered complicated scenarios. We manually annotated six types and more than $20{,}000$ prohibited items, which is at least 100 times larger than all existing datasets. In methodology, we formulate X-ray images as the overlap of several sub-images, therefore sampled from a mixture distribution. Motivated by filtering irrelevant information, we present an algorithm to refine mid-level features in a hierarchical and iterative manner. In practice, we switch off iteration to optimize the network weights in an approximate but efficient manner. A novel loss function is also built upon the hierarchical architecture to deal with heavy data imbalance between positive and negative classes. Beyond a few popular network backbones, our approach produces consistent gain in both classification and localization accuracy, establishing a strong baseline for the proposed task.

The future research mainly lies in two directions. First, the formulation of overlapping images from the penetration assumption is not accurate in many aspects – we look forward to more effective approaches based on a better physical model. Second, the connection between overlapping images and natural images, *e.g.*, object occlusion, remains unclear – studying this topic may imply some ways of extending these approaches to a wider range of applications.

Acknowledgments. This work was supported by the NSFC grant 61836012, 61771447, and 61671427, and Beijing Municipal Science and Technology Commission grant Z181100008918014.

References

[1] Samet Akçay, Mikolaj E Kundegorski, Michael Devereux, and Toby P Breckon. Transfer learning using convolutional neural networks for object classification within x-ray baggage security imagery. In *ICIP*, pages 1057–1061, 2016. 2, 3

[2] Muhammet Baştan, Mohammad Reza Yousefi, and Thomas M Breuel. Visual words on baggage x-ray images. In *CAIP*, pages 360–368, 2011. 3

[3] Hakan Bilen and Andrea Vedaldi. Weakly supervised deep detection networks. In *CVPR*, pages 2846–2854, 2016. 3

[4] Szegedy Christian, Liu Wei, Jia Yangqing andF Sermanet Pierre, Reed Scott, Anguelov Dragomir, Erhan Dumitru, Vanhoucke Vincent, and Rabinovich Andrew. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 3

[5] Antonia Creswell, Kai Arulkumaran, and Anil A. Bharath. On denoising autoencoders trained to minimise binary cross-entropy. *CoRR*, abs/1708.08487, 2017. 3

[6] Ali Diba, Vivek Sharma, Ali Mohammad Pazandeh, Hamed Pirsiavash, and Luc Van Gool. Weakly supervised cascaded convolutional networks. In *CVPR*, number 8, page 9, 2017. 3

[7] Thibaut Durand, Taylor Mordan, Nicolas Thome, and Matthieu Cord. Wildcat: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation. In *CVPR*, pages 5957–5966, 2017. 3

[8] Thibaut Durand, Nicolas Thome, and Matthieu Cord. Weldon: Weakly supervised learning of deep convolutional neural networks. In *CVPR*, pages 4743–4752, 2016. 3

[9] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010. 2, 6

[10] Thorsten Franzel, Uwe Schmidt, and Stefan Roth. Object detection in multi-view x-ray images. *PR*, pages 144–154, 2012. 2

[11] Ross Girshick. Fast r-cnn. In *ICCV*, pages 1440–1448, 2015. 3

[12] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. 3

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3, 6, 7, 8

[14] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, number 2, page 3, 2017. 3, 7

[15] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002. 3

[16] Simonyan Karen and Zisserman Andrew. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3

[17] Wei Ke, Jie Chen, Jianbin Jiao, Guoying Zhao, and Qixiang Ye. Srn: Side-output residual network for object symmetry detection in the wild. In *CVPR*, pages 1068–1076, 2017. 4

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 3

[19] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015. 1

[20] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie. Feature pyramid networks for object detection. In *CVPR*, number 2, page 4, 2017. 4

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014. 2

[22] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37, 2016. 3

[23] Domingo Mery. X-ray testing by computer vision. In *CVPRW*, pages 360–367, 2013. 1

[24] Domingo Mery and Carlos Arteta. Automatic defect recognition in x-ray testing using computer vision. In *WCCV*, pages 1026–1035, 2017. 1

[25] Domingo Mery, Vladimir Riffo, Uwe Zscherpel, German Mondragón, Iván Lillo, Irene Zuccar, Hans Lobel, and Miguel Carrasco. Gdxray: The database of x-ray images for nondestructive testing. *Journal of Nondestructive Evaluation*, 34(4):42, 2015. 1, 2, 3, 5

[26] Domingo Mery, Erick Svec, and Marco Arias. Object recognition in baggage inspection using adaptive sparse representations of x-ray images. In *PSIVT*, pages 709–720, 2015. 2, 3

[27] Domingo Mery, Erick Svec, Marco Arias, Vladimir Riffo, Jose M Saavedra, and Sandipan Banerjee. Modern computer vision techniques for x-ray testing in baggage inspection. *Systems, Man, and Cybernetics: Systems*, 47(4):682–692, 2017. 1

[28] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. 3

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 3

[30] M Roomi and R Rajashankarii. Detection of concealed weapons in x-ray images using fuzzy k-nn. *International Journal of Computer Science, Engineering and Information Technology*, 2(2), 2012. 2, 3

[31] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017. 3

[32] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016. 6, 7

[33] Peng Tang, Xinggang Wang, Xiang Bai, and Wenyu Liu. Multiple instance detection network with online instance classifier refinement. In *CVPR*, pages 3059–3067, 2017. 3

[34] Diana Turcsany, Andre Mouton, and Toby P Breckon. Improving feature-based object recognition for x-ray baggage security screening using primed visualwords. In *ICIT*, pages 1140–1145, 2013. 2, 3

[35] Fang Wan, Pengxu Wei, Zhenjun Han, Jianbin Jiao, and Qixiang Ye. Min-entropy latent model for weakly supervised object detection. *TPAMI*, 2019. 3

[36] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *CVPR*, pages 3462–3471, 2017. 1, 2, 3, 6

[37] Ren Weiqiang, Huang Kaiqi, Tao Dacheng, and Tan Tieniu. Weakly supervised large scale object localization with multiple instance learning and bag splitting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(2):405–416, 2016. 3

[38] Jianming Zhang, Sarah Adel Bargal, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. *IJCV*, 126(10):1084–1102, 2018. 6

[39] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, pages 2921–2929, 2016. 2, 3, 7

[40] Yi Zhu, Yanzhao Zhou, Qixiang Ye, Qiang Qiu, and Jianbin Jiao. Soft proposal networks for weakly supervised object localization. 2017. 3