BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 66, No. 6, 2018
DOI: 10.24425/bpas.2018.125935

Fast multispectral deep fusion networks

V. OSIN[1,2], A. CICHOCKI[1], and E. BURNAEV[1]*

[1]Skolkovo Institute of Science and Technology, 3 Nobel Street, 143026 Moscow, Russia
[2]Philips Lighting, High Tech Campus 48, 5656 AE Eindhoven, Netherlands

Abstract. Most current state-of-the-art computer vision algorithms use images captured by cameras, which operate in the visible spectral range as input data. Thus, image recognition systems that build on top of those algorithms can not provide acceptable recognition quality in poor lighting conditions, e.g. during nighttime. Another significant limitation of such systems is high demand for computational resources, which makes them impossible to use on low-powered embedded systems without GPU support. This work attempts to create an algorithm for pattern recognition that will consolidate data from visible and infrared spectral ranges and allow near real-time performance on embedded systems with infrared and visible sensors. First, we analyze existing methods of combining data from different spectral ranges for object detection task. Based on the analysis, an architecture of a deep convolutional neural network is proposed for the fusion of multi-spectral data. This architecture is based on the single shot multi-box detection algorithm. Comparison analysis of the proposed architecture with previously proposed solutions for the multi-spectral object detection task shows comparable or better detection accuracy with previous algorithms and significant improvement of the running time on embedded systems. This study was conducted in collaboration with Philips Lighting Research Lab and solutions based on the proposed architecture will be used in image recognition systems for the next generation of intelligent lighting systems. Thus, the main scientific outcomes of this work include an algorithm for multi-spectral pattern recognition based on convolutional neural networks, as well as a modification of detection algorithms for working on embedded systems.

Key words: multi-spectral imaging, data fusion, deep learning, convolutional networks, object detection, image segmentation.

1. Introduction

During the last couple of years, research and development in computer vision systems have been rapidly growing. Accordingly, the need has emerged to use additional data sources to solve common problems in image recognition systems, such as object detection and semantic segmentation. Visible cameras that captured visible light in greyscale or RGB images have been the standard device for image capturing. However, image recognition systems, which build upon datasets that contain such images, highly depend on the sun or artificial lighting. Also, such systems can not recognize objects in total darkness.

One of the possible solutions is to use infrared cameras as the primary imaging device. It is possible to use active or passive sensors for infrared cameras. Active sensors, which illuminate the scene with near-infrared radiation, are less dependent on the lighting conditions. However, in many applications, a passive image sensor is preferred. In the mid- and long-wavelength spectrum radiation is emitted by objects themselves. The intensity in such sensors highly depends on the object temperature. Thereby passive sensors do not depend on any external energy source. Hot objects, e.g. humans, are much easier to distinguish in the thermal image, while colors of any other object with low temperature are invisible. A special detector technology is required to capture thermal-infrared radiation. Originally, it was developed for night vision purposes for the military, and the devices were very expensive. Example of visible-infrared pair image is presented in Fig. 1.

The technology was later commercialized and has developed quickly over the last few decades, resulting in both better and cheaper cameras. This has opened a broader market, and the technology is now being introduced to a wide range of different applications, such as building inspection, gas detection, industrial appliances, medical science, agriculture, fire detection, and surveillance [1].

Concerning significant price drop of the infrared sensor, there is a possibility to utilize such cameras in image recognition tasks that require competitive and high nighttime detection accuracy, e.g. self-driving cars or video surveillance. However, despite variety of multi-spectral cameras, most datasets that contain infrared images have a lack of samples for proper training of state-of-the-art algorithms. Besides, such algorithms require significant computational resources, and it makes them

Fig. 1. Visible and thermal image of the same scene [1]

*e-mail: E.Burnaev@skoltech.ru

Manuscript submitted 2018-05-02, revised 2018-06-30, initially accepted for publication 2018-07-03, published in December 2018.

ineffective on the low-powered embedded systems. At the same time, there is high demand for smart embedded systems which can use mentioned algorithms on board, e.g. in smart connected systems and other Internet of Things applications.

Given the above background, the primary goal of this work is to design, implement and experimentally evaluate an algorithm that allows utilizing heterogeneous data from different sources for image recognition tasks. Designed algorithm should provide near real-time performance on embedded systems and accuracy comparable to more resource-intensive state-of-the-art detection frameworks. Thus the following research objectives were defined:

1. Conduct analysis and comparison of modern object detection algorithms. The primary criteria are resource-consumption and detection quality.
2. Perform necessary modifications of the algorithm to run it on low powered embedded devices.
3. Implement architecture extension of the developed algorithm to support multi-spectral data processing.
4. Conduct experiments with the proposed architecture to verify the importance of additional data source in object detection, semantic segmentation with heterogeneous data.

All solutions, proposed in this work, are based on convolutional neural network architectures and fusion functions that perform consolidations of extracted features from multi-spectral input data. The results of this work can be used to build image recognition systems for variety of applications, e.g. in video surveillance systems, self-driving cars, automatic annotation for multi-spectral satellite images and in any other domain which impose combination of heterogeneous data. Near real-time performance of provided multi-spectral algorithm for object detection allows to use it in smart connected systems, e.g. in smart office for light control or energy management. Also, development of a large dataset of infrared images in the future creates possibilities to re-train described models exclusively on infrared images, this will avoid disrupting privacy when installing these systems in office buildings.

The structure of present work is the following. In Section 2 we provide an overview of state of the art in standard and multi-spectral object detection problem, along with a short review of neural network architectures for semantic segmentation. In Section 3 we define basic fusion rules, present multi-spectral architectures for various computer vision tasks and discuss fast object detection on embedded devices. In addition, experimental results are provided with implementation details. In Section 4 we make conclusions.

2. Overview. Selection of a primary architecture

This section provides a top-down review presenting the current problem. We start with a review and analysis of current state-of-the-art object detection algorithms in order to find and justify the most suitable detector for multi-spectral extension regarding accuracy and resource efficiency. Also, previously proposed solutions, based on convolutional neural networks for multi-spectral object detection, are analyzed. A short overview of a neural network architecture for semantic segmentation, which was used in experiments, is also provided.

2.1. Object detection. The primary goal of object detection is to find objects in an image and to classify them. The standard output of object detection algorithm is described by the corresponding class label and bounding box coordinates. Image classification is done before object detection and provides confidences for classes. Localization task is performed by regression for bounding box coordinates. Object detection systems are usually trained with image and ground truth bounding box and the regression is conducted by calculation of L2 distance between the predicted bounding box and the ground truth. Intersection over union (IOU) between predicted bounding boxes and ground truth helps to verify most stable detections. There are two major classes of object detection frameworks – multi-staged and single shot. The brief review of this classes is provided below.

The first representative of the multi-staged detection framework is region-based convolutional neural network (R-CNN). R-CNN is a visual object detection system that combines region proposals with features extracted from a convolutional neural network (CNN). R-CNN creates bounding boxes using an algorithm called selective search [2] that slides through the image with windows of different sizes and combines adjacent pixels by color, intensity or texture to find objects. Once the proposals have been created, R-CNN warps the region to a standard square size and passes it through to a modified version of a convolution neural network feature extractor. On the final layer of the CNN, R-CNN adds a support vector machine that detects object presence and class of detected object. The full structure of R-CNN is presented on Fig. 2.

The main disadvantages of this approach are multi-stage training pipeline that makes it difficult to re-train for new datasets; object detection with VGG16 takes 47 sec/image on GPU which is very slow and far away from near real-time performance. R-CNN is slow because it performs a CNN forward pass for each object proposal, without sharing computations.

To address all issues mentioned above with R-CNN authors proposed a new modified fast R-CNN architecture. The input to the fast R-CNN is input image and the region proposals that have been generated using selective search [2]. In order to avoid CNN forward pass for each region proposal of each image authors introduced technique known as region of inter-

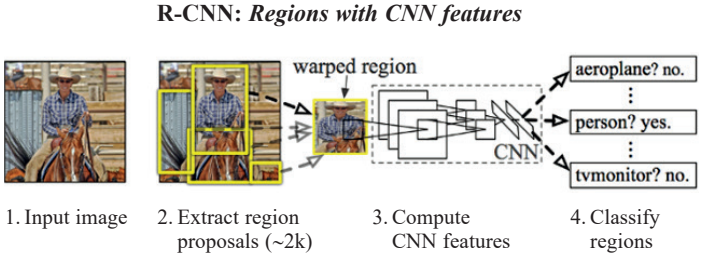

Fig. 2. Region-based Convolution Neural Network structure [3]

esting pooling (RoI pooling layer). RoI pooling layer shares the forward pass of a CNN for an image across its subregions. Convolution features for each region are obtained by mapping of the corresponding section from output feature layer of the CNN. Then max pooling operation is performed with each feature to fix sizes of feature maps for further computations. All these steps allow making only one forward pass of the original image as opposed to 2000 in R-CNN. Also, fast R-CNN utilizes CNN feature extractor, classifier, and bounding box regressor into the single network. The whole network is trained with a dual loss function, the SVM classifier is replaced with a softmax layer on top of the CNN for classification, and linear regression layer is added in parallel for bounding box prediction. The whole structure of fast R-CNN is presented in Fig. 3.

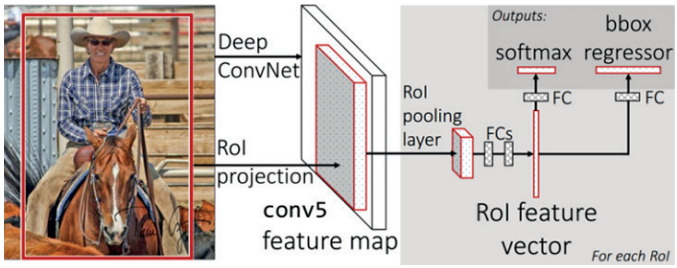

Fig. 3. Fast R-CNN joint training framework structure [4]

The external region proposals problem and still slow inference speed of fast R-CNN have been improved in further modification called faster R-CNN. Faster R-CNN provides real-time detection performance with 10 ms per image including time cost for region proposals. It replaced the previous region proposals method with region proposal network (RPN). RPN accepts convolution feature map of the image as input and provides a set of rectangular object proposals with confidence score as output. In general, faster R-CNN has four losses: RPN classification (object or not object), RPN bounding box proposal, fast R-CNN Classification (standard object classification), fast R-CNN bounding-box regression as a refinement of RPN bounding box proposals.

The detailed structure of faster R-CNN is presented in Fig. 4. All mentioned modifications lead to significant improvement in inference speed and detection accuracy. Comparison of R-CNN based multi-staged object detection frameworks is presented in Table 1.

Table 1
Comparison of multi-staged object detection frameworks

Detection Framework	Test time (sec. per image)	Speedup	mAP (Pascal VOC 2007)
R-CNN	50	1×	66
fast R-CNN	2	25×	66.9
faster R-CNN	0.2	250×	69.9

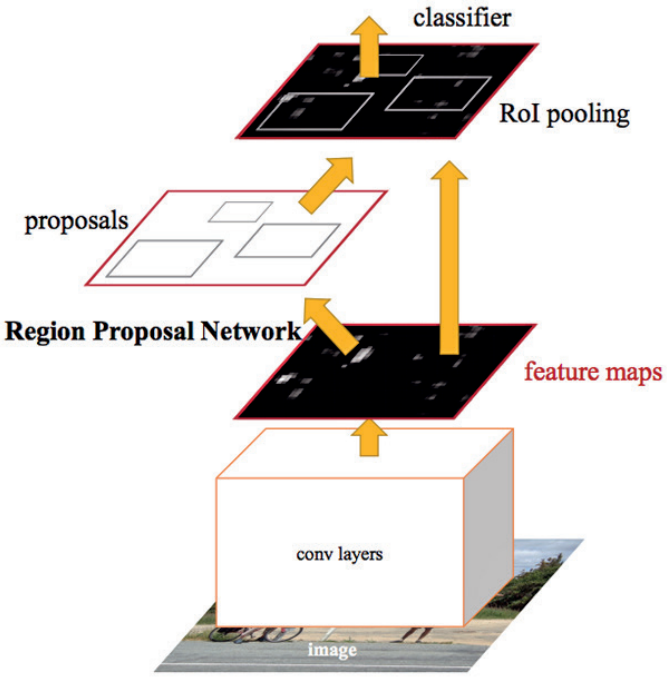

Fig. 4. Faster R-CNN structure overview [6]

According to comparison results, R-CNN object detection algorithms did not achieve near real-time inference speed for launching on low-powered embedded devices with CPU.

Methods mentioned above have region proposals part followed by the high-quality classifier for this proposals. These methods are very accurate but require significant computational resources.

However, there is the second type of object detection algorithms that is called single shot and is designed to combine two stage detection of R-CNN detectors into the single convolutional neural network. The main idea of these algorithms is the usage of pre-defined bounding boxes, and convolutional feature maps from last layers in the network for a class score and bounding box offsets prediction.

The first method that uses this idea is called YOLO [5]. It predicts classes and bounding boxes using single feature map. This approach helps to increase inference speed but creates several limitations during detection, such as inaccuracies due to small objects, different aspects and ratios of objects (Fig. 5).

These problems were successfully addressed in single shot multibox detector (SSD) paper [7]. SSD requires assigning the default boxes to ground truth boxes with some matching strategy. Best default box for corresponding ground truth box is the one that best fits regarding location, aspect ratio, and scale. For this matching strategy default boxes with the best intersection over union and higher than 0.5 are selected. The overall training objective L of SSD is based on the weighted combination of bounding box regression and classification loss functions. Let $x_{ij}^{p} \in \{1, 0\}$ be an indicator for matching the i-th default box to the j-th ground truth box of category p, then

$$L(x, c, l, g) = \left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right)/N,$$

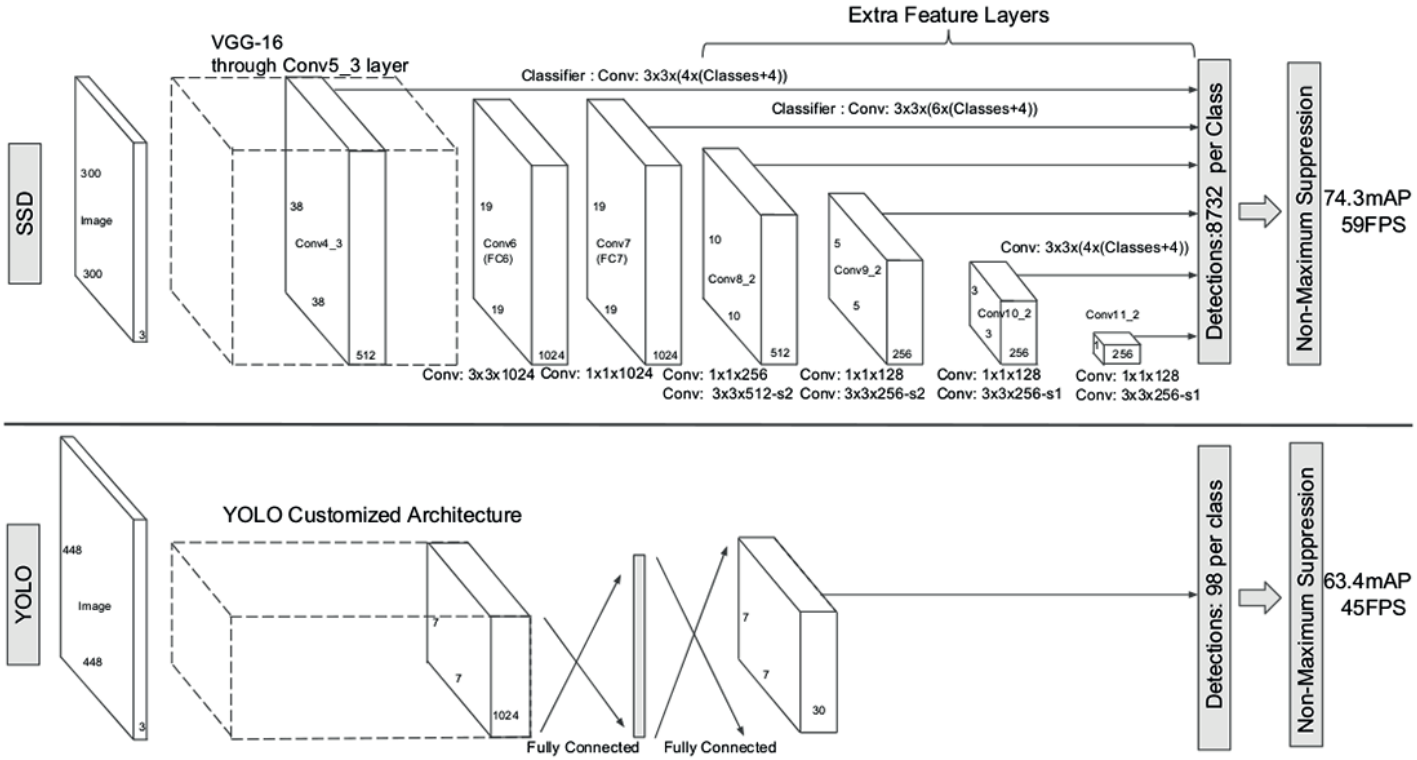

Fig. 5. A comparison between single shot detection models architectures. SSD adds several feature layers to the end of the base network, which predicts the offsets to default boxes of different scales and aspect ratios and their associated confidences [7]

where N is the number of best-matched default boxes and the weight term α is adjusted by cross-validation. If $N = 0$ loss is equal to 0. The localization loss is a Smooth L1 loss [4] between the predicted box (l) and the ground truth box (g) parameters: similar to faster R-CNN [6], SSD method regresses to offsets for the center (cx, cy) of the default bounding box (d) and for its width (w) and height (h) [7], i.e.

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} \cdot \text{smooth}_{\text{L1}}\left(l_i^m - \hat{g}_j^m\right),$$

$$\hat{g}_j^{cx} = \left(g_j^{cx} - d_i^{cx}\right)/d_i^w, \quad \hat{g}_j^{cy} = \left(g_j^{cy} - d_i^{cy}\right)/d_i^h,$$

$$\hat{g}_j^w = \log\left(\frac{g_j^w}{d_i^w}\right), \quad \hat{g}_j^h = \log\left(\frac{g_j^h}{d_i^h}\right).$$

The confidence loss is the softmax loss over multiple class scores (c)

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^p \log\left(\hat{c}_i^p\right) - \sum_{i \in Neg} \log\left(\hat{c}_i^0\right),$$

$$\hat{c}_i^p = \frac{\exp\left(c_i^p\right)}{\sum_p \exp\left(c_i^p\right)}.$$

SSD uses feature maps from different layers to handle scale variance. Scales and aspect ratios for default boxes in each feature map are computed as:

$$s_k = s_{min} + \frac{s_{max} - s_{min}}{m - 1}(k - 1), \quad k \in [1, m],$$

where m is the number of feature maps, $s_{min} = 0.1$, $s_{max} = 0.95$. Different aspect ratios are considered at each scale $a_r \in \{1, 2, 3, 0.5, 0.3\}$. To avoid significant class imbalance between positive and negative default boxes samples with higher confidence score are used to keep positive-negative ratio equal to 3:1. SSD algorithm uses several activation maps in different scales for prediction which helps to achieve higher mean average precision (mAP) by better detecting small objects.

According to results in Table 2, SSD is the best available method for fast multispectral detection task extension because it provides a trade-off between accuracy and inference speed. Also, most commercially available infrared cameras capture images in low resolution which is also perfectly fit with SSD, since it supports lowest possible resolution.

Table 2
Comparison of single shot detection frameworks
(FPS – frames per second)

Method	mAP (VOC 2007)	FPS	Input resolution
Fast YOLO	52.7	155	448×448
YOLO (VGG16)	66.4	21	448×448
SSD300	74.3	46	300×300
SSD512	76.8	19	512×512

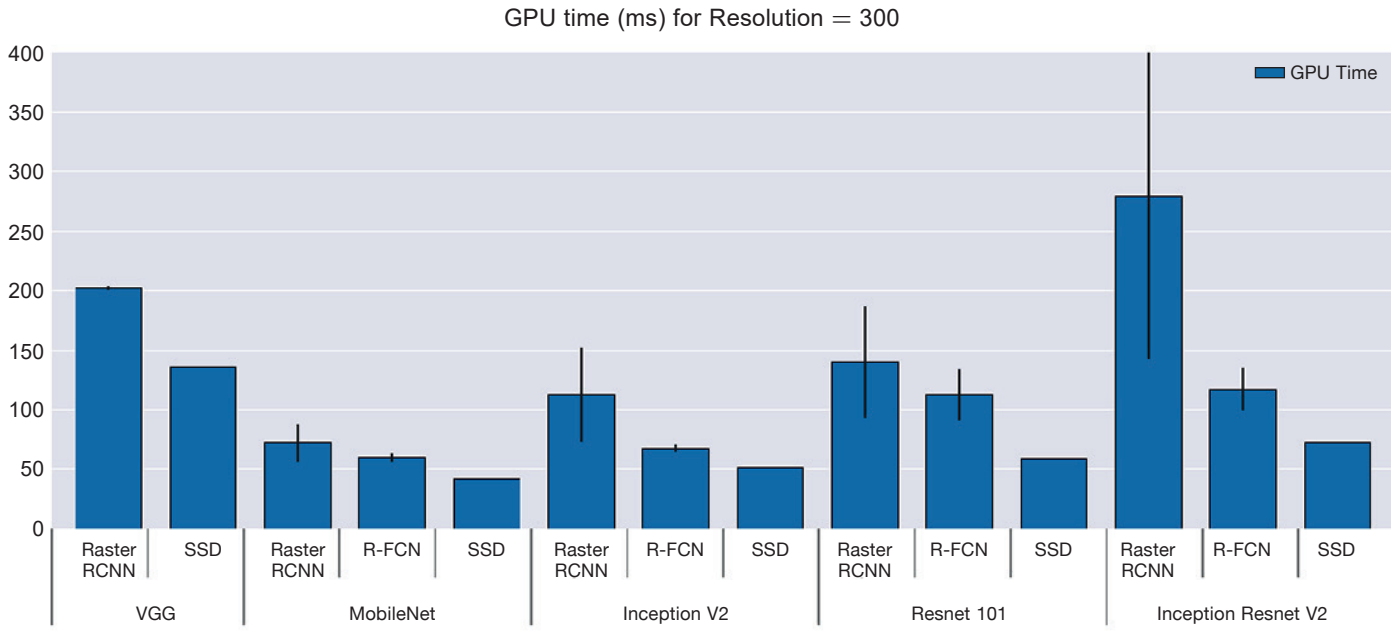

Fig. 6. GPU time (milliseconds) for different models, for image resolution equal to 300 [8]

Also, in [8] authors presented high-quality review and comparison of modern convolutional object detectors regarding speed and accuracy. They analyzed SSD and faster R-CNN for sensitivity to different feature extractors and showed that SSD is less sensitive to the quality of feature extractor than faster R-CNN. This property will be used in subsection 3.2 for fast single shot detector development. Also, in [8] they performed analysis of GPU time for different object detectors which show that SSD is faster than other competitors, such as faster R-CNN.

Let us summarize all mentioned key properties of Single Shot Multibox Detector:

- **Inference speed.** High detection speed achieved by adaptation of single convolutional neural network that directly returned scores for classes and corresponding bounding boxes.
- **Detection accuracy.** Due to different sizes of last feature layers, SSD performs detection for objects with different shapes and sizes.
- **Low resolution support.** Since cheap infrared cameras provide images in resolution 300×300, SSD already supports such resolution which simplifies training procedure and creates opportunities for industrial applications.
- **Modular structure.** Since SSD is less sensitive to changes of base network and extra classification layers can be varied, it is suitable to construct detectors for specific needs, such as detectors for embedded systems.

Considering presented properties, SSD was considered as a primary architecture to be used on low-powered embedded systems and suitable for a multi-spectral extension for object detection task, discussed in subsection 3.2.

Let us also note that different data augmentation strategies, such as sampling patches with some minimum overlap, random sampling, etc. can be used to make a model more robust to various input shapes for both classes of object detection frameworks, discussed above.

2.2. Multispectral object detection. By general definition, Data Fusion is a formal framework for fusion of data originating from different sources. Specifically, image fusion can be classified into three categories depending on the stage at which fusion takes place: pixel level, feature level, decision level. Since the core part of all object detection algorithms mentioned above is CNN, it is reasonable to discuss feature level fusion, because CNN is a perfect feature extractor. Feature level fusion methods deal with data at higher processing level than pixel level methods. The general scheme for feature-level fusion systems is presented in Fig. 7.

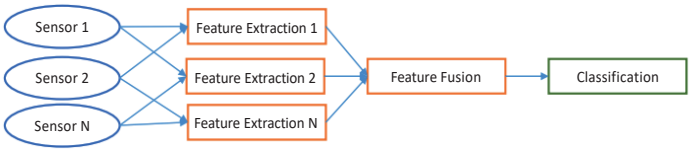

Fig. 7. General feature level fusion scheme

The most popular dataset and benchmark for evaluation of multi-spectral object detection algorithms is KAIST [9]. This dataset contains temporally and spatially aligned visible and thermal images with resolution 640×512. In general, there are 95*k* pairs of visible-thermal infrared pairs. The baseline results for this dataset are provided by algorithm which is based on aggregated channel features (ACF) [10] with an extension that incorporates a contrast-enhanced version of original thermal-infrared images as well as a histogram of oriented gradients features of this image as primary channels.

There is a quite few literature on multi-spectral object detection with convolutional neural networks. In [11] authors proposed multi-spectral detection framework that is based on the

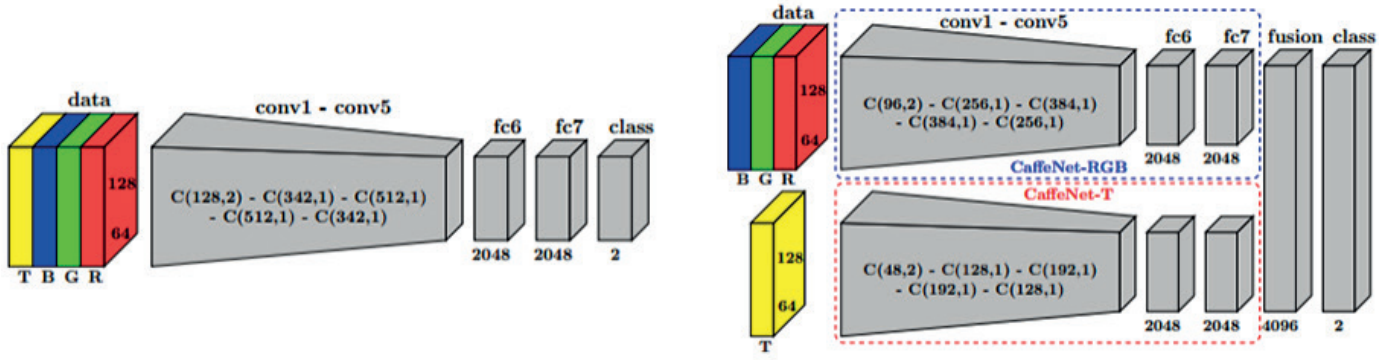

Fig. 8. Multispectral image fusion using AlexNet and ACF [11]. Left: Early fusion. Right: Late fusion

R-CNN framework. The main difference is in region proposal algorithm that is based on ACF [10] from KAIST benchmark [9]. After region proposals convolution neural network is used to fuse information of different modalities and perform binary classification. The primary fusion strategies are early and late fusion. Early fusion architecture is based on AlexNet [12] and different modalities combined at pixel-level. On the other hand, late fusion architectures process data of the two modalities separately in subnetworks and perform fusion in a fully connected layer. According to results, late fusion architecture provides better detection results because it fuses information at a stage where spatial features are less relevant. Both fusion schemes are presented in Fig. 8.

In [13] authors proposed end-to-end multi-spectral detection framework based on faster R-CNN [6]: fusion schemes are based on the fusion of feature maps in different stages in CNN to evaluate the influence of spatial and contextual features on resulting detection score. The most promising architecture was designed to perform halfway fusion.

According to results, two separately trained faster R-CNN that fused multi-spectral features in the middle of the network provided the best solution for multi-spectral object detection. However, this approach requires more computational resources for subnetworks and is highly complex due to specific training pipeline. All current results of multi-spectral detection from all discussed approaches are presented in Table 3. Above-mentioned results are highly inspiring to apply multi-spectral support for single shot detection algorithms in order to improve inference speed of multi-spectral detection as well as to evaluate different fusion functions for this task.

Table 3
Current multispectral pedestrian detectors results on KAIST dataset

Method	Log-average miss rate
ACF + T + HOG [9]	50.48%
R-CNN (AlexNet) [11]	43.80%
faster R-CNN (VGG16) [13]	36.99%

2.3. Semantic segmentation. To better test fusion architectures two additional tasks containing heterogeneous data were selected: multi-spectral semantic segmentation of satellite images and air quality prediction. For the task of air quality prediction custom convolutional neural network architecture was used. This task and experiments are discussed in Section 3.

Semantic segmentation for images can be defined as the process of grouping parts of the image so that each pixel in a group corresponds to the object class of the group. In the present work, the object classes correspond to buildings and background. Besides, for a multi-class semantic segmentation, the classes can be grouped into roads, trees, water, etc. This subsection provides a short overview of the fully convolutional neural network that was used in experiments. This fully convolutional model for the task of semantic segmentation was inspired by the family of U-Net [14] architectures, where low-level feature maps are

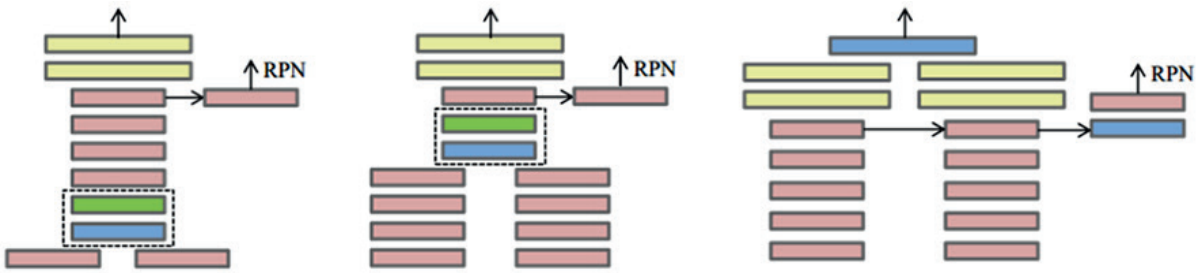

Fig. 9. Proposed approaches to fuse color and thermal images for multispectral pedestrian detection. Left: low level fusion (Early). Center: middle level fusion (Halfway). Right: high level fusion (Late). Red and yellow boxes represent convolutional and fully-connected layers. Blue boxes represent concatenate layer. Green boxes denote dimensionality reduction layer [13]

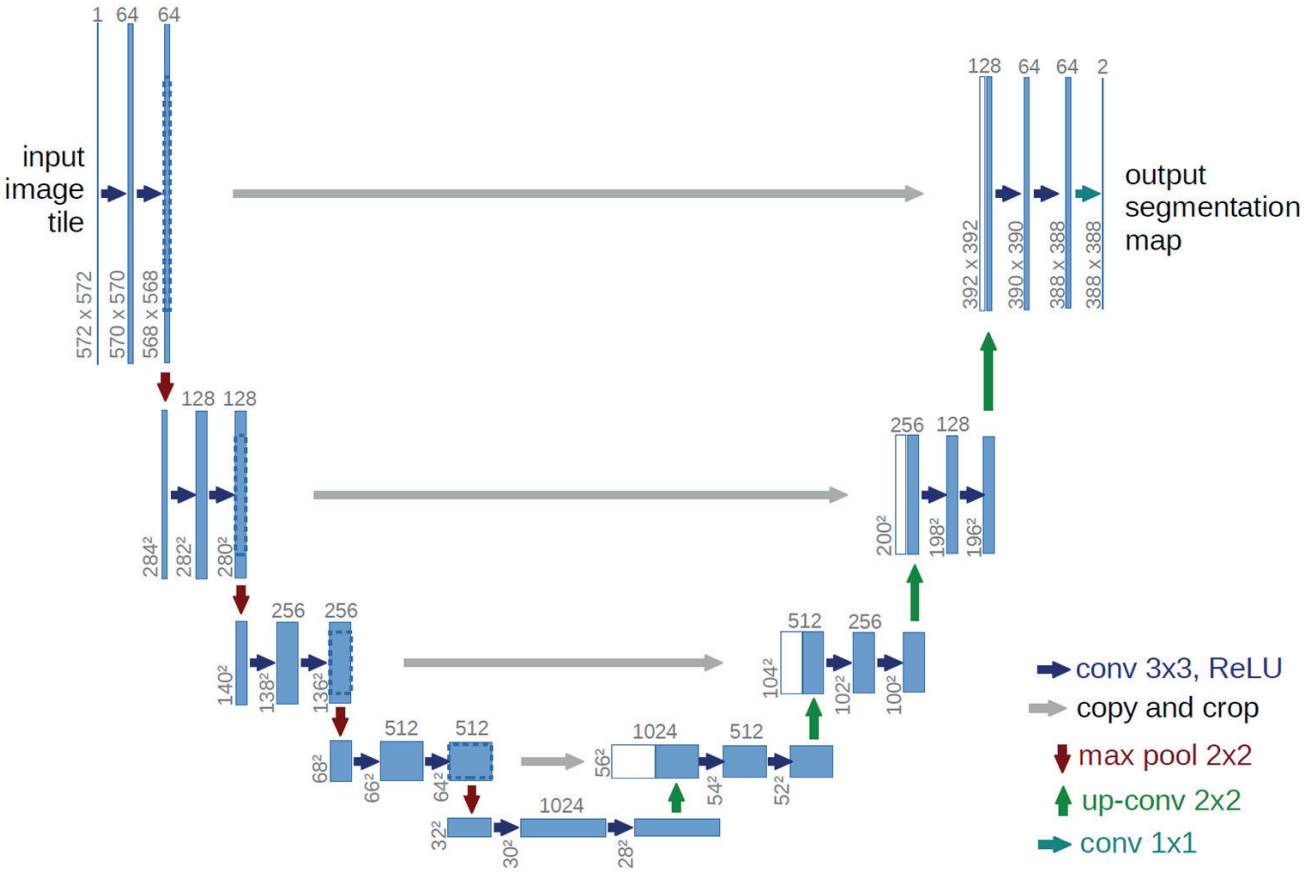

Fig. 10. U-net architecture (example for 32×32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote different operations [14]

combined with higher-level ones, which enables precise localization. The main modification of this network is a large number of feature channels in up-sampling part, which allow propagating context information to higher resolution layers. This type of network architecture was specially designed to solve image segmentation problems effectively.

The basic variant of U-Net architecture is presented in Fig. 10. It consists of a contracting path (left side) and an expansive path (right side). The contracting path follows the typical architecture of a convolutional neural network. It consists of the repeated application of two 3×3 convolutions, each followed by a rectified linear unit activation function and a 2×2 max pooling operation with stride 2 for down-sampling. The number of feature channels is doubled at each down-sampling step. Every step in the expansive path consists of an up-sampling of the feature map followed by a 2×2 convolution that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3×3 convolutions, each followed by an activation function. At the final layer, a 1×1 convolution is used to map each 64-component feature vector to the desired number of classes. In total the network has 23 convolutional layers [14].

Application of such network for multispectral semantic segmentation of satellite images is discussed in Section 3.

3. Methodology and experiments

In this section, fusion functions definition is discussed, as well as the construction of a fusion block for multi-spectral architectures. In addition, faster version of current SSD object detection framework is presented. Inference speed of SSD was improved with base network architecture modifications. Also, two multi-spectral architectures suitable for multi-spectral object detection and semantic segmentation tasks were defined and evaluated on open datasets. Additional experiments were conducted on the task of air quality prediction which also required fusion of heterogeneous data.

3.1. Fusion functions definition. In general, input data for fusion function can represent different types of sources. In this case, only image-like data sources are considered, such as visible-infrared images, multi-spectral satellite images or concentration, temperature, pressure maps. For simplicity, lets consider the case of visible-infrared fusion.

A fusion function $f : x^{RGB}, x^{IR} \rightarrow y$ fuses two convolutional feature maps $x^{RGB} \in \mathbb{R}^{H \times W \times D}$ and $x^{IR} \in \mathbb{R}^{H' \times W' \times D'}$, to create output fused feature map $y \in \mathbb{R}^{H'' \times W'' \times D''}$, where W, H, D are width, height and depth of the considered feature map. In our case feature maps with the same shapes will be fused, i.e. for

simplicity $H = H'$, $W = W'$, $D = D'$. Let us consider several typical fusion functions which can be used in our experiments.

3.1.1. Weighted sum fusion. Weighted sum fusion function $y^{wsum} \in f\left(x^{RGB}, x^{IR}, \alpha\right)$ computes the sum of two feature maps at the same spatial positions i, j and feature channel d with some weight coefficient α:

$$y^{wsum}_{i,j,d} = x^{RGB}_{i,j,d} + \alpha \times x^{IR}_{i,j,d},$$

where $1 \leq i \leq H$, $1 \leq j \leq W$, $1 \leq d \leq D$ and $x^{RGB}, x^{IR}, y \in \in \mathbb{R}^{H \times W \times D}$. Parameter α is adjusted by cross validation. This fusion function can define arbitrary correlation between feature maps, and optimization of filters in subsequent network layers makes this fusion useful.

3.1.2. Max fusion. Maximum fusion function $y^{max} = f\left(x^{RGB}, x^{IR}\right)$ takes the maximum of two feature maps:

$$y^{max}_{i,j,d} = \max\left(x^{RGB}_{i,j,d}, x^{IR}_{i,j,d}\right).$$

3.1.3. Concatenation fusion. Concatenation fusion rule stacks two convolutional feature maps at the same spatial locations i, j across depth axis of this feature map d:

$$y^{concat}_{i,j,2d} = x^{RGB}_{i,j,d}, \; y^{concat}_{i,j,2d-1} = x^{IR}_{i,j,d}, \tag{1}$$

where $y \in \mathbb{R}^{H \times W \times 2D}$.

Concatenation also does not produce correlation between convolutional feature maps and transfer this task to subsequent layers in neural network.

3.1.4. Convolution fusion. Convolution fusion function $y^{conv} = = f^{conv}\left(x^{RGB}, x^{IR}\right)$ stacks two feature map channels using concatenation fusion rule (1). After that it applies convolution operation to the concatenated features with a new set of filters $f \in \mathbb{R}^{1 \times 1 \times 2D \times D}$ and biases $b \in \mathbb{R}^{D}$:

$$y^{conv} = y^{concat} \star f + b,$$

where the output depth is D, and the filters have dimensions $1 \times 1 \times 2D$. This fusion rule performs dimensionality reduction by a factor of two and provides weighted combinations of the feature maps at the same spatial location. Using f^{conv} as a trainable filter we can perform proper feature extraction from the two feature maps and improve further recognition results.

Fusion block, used for multi-spectral architectures construction, can use one of the defined fusion functions in order to perform feature aggregation of different modalities. The main characteristic of convolution fusion block is support of input feature maps with different number of channels, while other fusion block (based on sum and max fusion rules) require equal number of channels for input feature maps. The schematic overview of fusion blocks is presented in Fig. 11.

3.2. Fast single shot detection. In [8] authors already compared different feature extractors for the object detection task, but they mostly considered very deep and computationally expensive CNNs. The only exception is lightweight MobileNet [15] which was recently presented. Thus, we selected several lightweight architectures as base networks with VGG16 [16] that was originally used in [7]. In more detail, we considered the following three feature extractors as base networks:

- **AlexNet [12].** This network used a relatively simple layout, compared to current state-of-the-art architectures. It was made up of 5 convolution layers, pooling layers, regularization layers, and fully connected layers. The designed network was used for classification with 1000 possible categories. Only convolutional and polling layers were used in SSD model with this base network.
- **SqueezeNet [17].** This network used small convolution filters of size 1×1 instead of commonly used 3×3 filters. Each such replacement provided decrease in number of parameters by a factor of 9. This size decrease was performed in the last part of the network to provide large field for convolutional layers. All these strategies were summarized in the "fire" module presented in Fig. 12. The whole network is constructed from such modules. Also, authors removed fully connected layer, located in the end of the network, and used average pooling for the final training layer. All these modifications allowed to reduce number of parameters by a factor of 50 compared with the original AlexNet and to reduce size of the model to 5 MB.
- **ResNet-18 [18].** This network architecture allows to construct very deep convolutional neural networks providing information between previous and further layers via usage of a residual block. In the original convolutional architecture

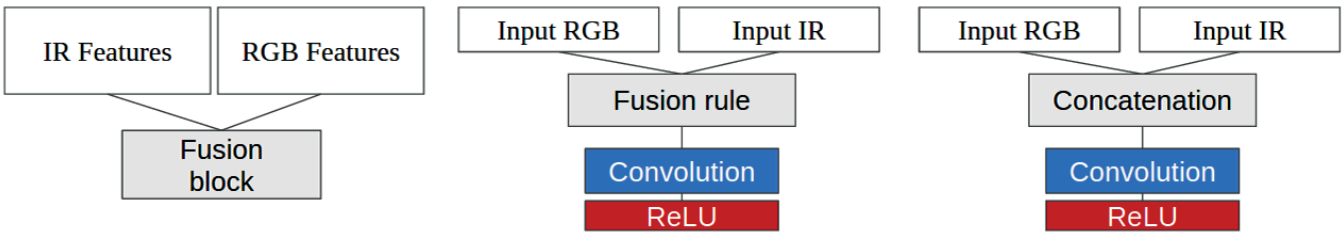

Fig. 11. Left: The general scheme of multi-spectral fusion. Middle: Fusion block structure includes fusion rule, convolution and activation layers. Right: Convolution fusion block with concatenation fusion as a main fusion rule

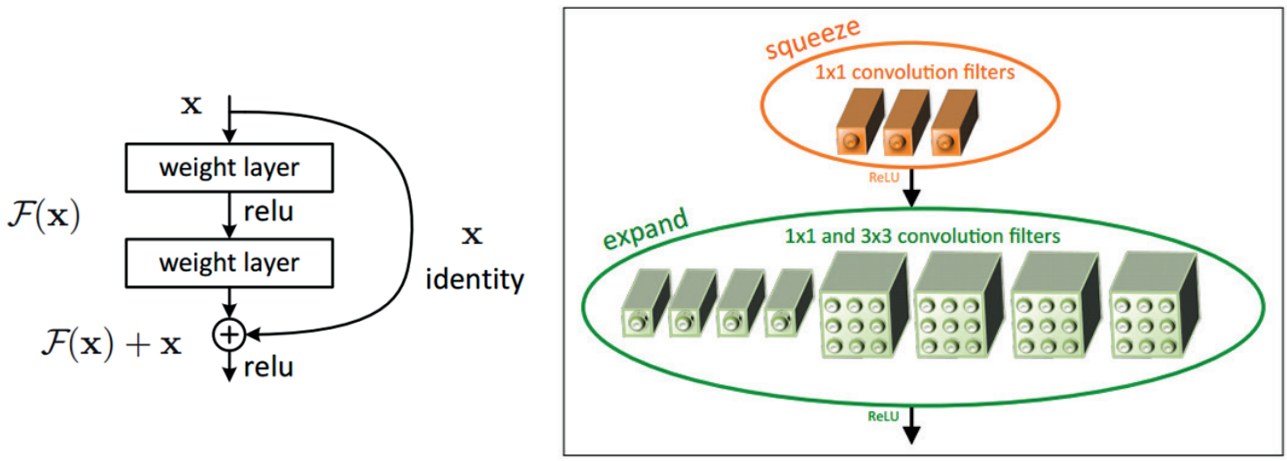

Fig. 12. Main components of tested base networks for SSD. Left: Residual block [18]. Right: "Fire" module [17]

an input x goes through the set of convolution and activation layers, resulting in some function $F(x)$. However, in case of the residual block instead of just computing such transformation, we compute such term $F(x)$ that is added to the input x to model nonlinear correction of identity transformation. Basically, the residual module, presented in Fig. 12, computes a difference or a slight change to the original input x to get a slightly altered representation. The authors believe that it is easier to optimize the residual mapping than to optimize the original mapping.

The primary strategy for experiments was to follow closely the methodology, described in [7]: always select the topmost convolutional feature map and a higher resolution feature map at a lower layer of the network, then adding a sequence of extra convolutional layers with spatial resolution decaying by a factor of 2 with each additional layer used for prediction. Also, batch normalization was used in all extra SSD layers to speed up convergence during training. Since convolutional feature map from the base network has different feature scale compared to other feature maps further in the network, L2 normalization technique [21] was used to scale the feature norm at each spatial location of the feature map to 20 and learn this magnitude during the training procedure.

Pascal VOC dataset was used as the primary dataset for training SSD detectors. This dataset contains images and corresponding labels for 20 different classes. For training combination of train and validation from VOC 2007 and VOC 2012 sets [24] was used resulting in 16551 images. For testing, only VOC 2007 test set was used with 4952 images.

During training it is necessary to optimize joint loss function of SSD that combines cross-entropy and smooth L1 loss for classification and regression tasks respectively. For the minimization of this training objective Adam [19] optimizer was used with learning rate 0.001 that was changed by a learning rate re-factor value 0.9 each 50 epochs. This setting allowed for smooth training without high oscillations of loss function value during optimization. All new detection networks were fine-tuned with the weights of the corresponding base network. All suggested base network architectures were previously trained for image classification task on the Imagenet large scale visual recognition challenge dataset [22]. The parameters of all newly added convolutional layers were initialized with the commonly used method discussed in [20]. SSD with ResNet-18 as a base network shows the best convergence than other tested architectures regarding joint training objective. Learning curves for the validation set are presented in Fig. 13.

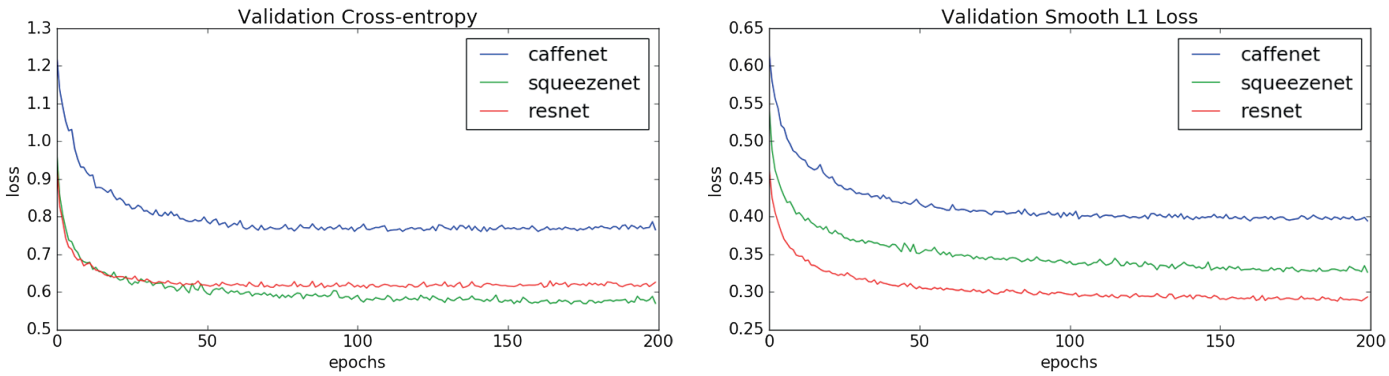

Fig. 13. Joint loss function value during SSD training with different base networks. Left: Cross-entropy loss function during training. Right: Smooth L1 loss function during training

Mean average precision was used as primary detection accuracy metric across all class categories. The training results show high sensitivity to different base network architectures. SSD with AlexNet base network showed bad detection accuracy because of reduced feature extraction ability of AlexNet, which is caused due to a low number of convolution and activation layers, and filters inside convolution layers. Squeezenet showed intermediate quality confirming efficiency of the "fire" module compared to the standard convolution-activation-pooling design. ResNet-18 showed the best detection results across all tested base network architectures, and being almost comparable with the VGG16 base network that was used in the original SSD design. For most classes that required for video surveillance or self-driving systems, ResNet showed nearly the same results as VGG16. Detection results for all tested base network architectures are in Table 4.

Table 4
Detection results for several classes of Pascal VOC 2007 dataset

Base Network	AlexNet	SqueezeNet	**Resnet-18**	VGG16 reduced
Person	43.92	57.02	**72.23**	74.39
Car	51.11	56.67	**79.07**	81.77
Bus	51.94	66.09	**74.98**	77.91
Bicycle	52.31	62.27	**77.89**	79.69
Motorbike	55.56	64.73	**79.43**	77.06
Train	60.44	68.42	**79.24**	84.01
Aeroplane	49.37	56.71	**70.98**	72.15
mAP	40.56	51.68	**69.45**	71.57

Table 5
Benchmarking results for SSD with different base networks on various devices

Base Network	**TITAN X GPU (FPS)**	**CPU I7-5820K CPU (FPS)**	**RPI (sec)**	**Size (MB)**
AlexNet	102	3.2	2.6	26.1
SqueezeNet	138	5.26	1.7	17.8
Resnet-18	**79**	**1.62**	**5.3**	**52.6**
VGG16 reduced	45	0.31	55.6	104.3

In addition, all detection architectures were tested on different devices in order to properly evaluate inference speed. The detector was tested on the following devices: NVIDIA TITAN X GPU, I7–5820K CPU, and Raspberry PI 3 Model B (RPI), as a baseline for the low powered embedded system. This setting allows comparing algorithm performance on high powerful GPU, middle-level CPU, and low-powered CPU. Also, models were compared in terms of size, because it is an important factor for embedded systems.

Squeezenet provided the best results in terms of inference speed, and model size compared to all tested base networks, which highly motivates to use such detector in embedded systems. However, ResNet-18 is the trade-off base network, which achieved comparable accuracy and near real-time performance on Raspberry PI, while reducing the model size by a factor of 2 comparing to the original SSD model. Thus, SSD with this base network architecture will be used for multi-spectral object detection task. All results are in Table 5.

3.3. Multi-spectral image recognition. After defining fusion functions let us propose architectures for multi-spectral image recognition. Proposed architectures are specifically designed for multi-spectral object detection and segmentation tasks, while it is also possible to utilize this design for processing heterogeneous data.

3.3.1. Multi-spectral SSD for pedestrian detection. The primary idea of a multi-spectral extension for object detection task is to extend SSD architecture with the additional feature extraction base network. This network was added in a siamese way to the main architecture. Additional network repeated structure of the standard base network and preserved the same number of filters in convolutional layers and the same spatial dimensions of feature maps. Weights of ResNet-18 were fixed during the fine-tuning procedure, as recommended in [23], and only weights of fusion layers and additional subnetwork were trained. The weights of the additional network were initialized with the scheme described in [20]. Schematic view of the proposed architecture is in Fig. 14.

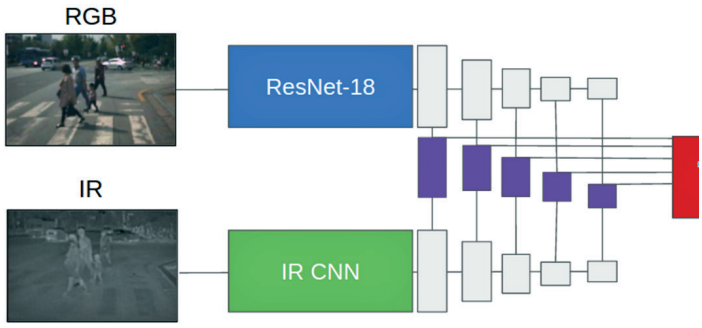

Fig. 14. Multi-spectral SSD architecture. Subnetwork for multi-spectral feature extraction is in green. Fusion blocks are in violet. Classification and non-maximum suppression blocks are in red

All multi-spectral experiments were conducted on a subset of the KAIST [9] dataset. This subset contains not occluded pedestrians with height more than 55 pixels. Test set contains 700 images and training set contains 7000 images. SSD with ResNet-18 base network architecture was fine-tuned using Adam optimizer [19] with initial learning rate 0.0001, 0.0001 weight decay and batch size equal to 16. Evaluation scheme used log-average miss rate described in [9] as the main evaluation metric. All inference speed results, presented earlier for

SSD with the ResNet-18 base network, were slightly degraded. However this architecture still performed faster than multi-spectral architectures based on siamese R-CNN architecture. Architecture with convolution fusion results showed better detection accuracy than other fusion blocks because this type of fusion preserves feature scale and trains multi-modal feature inference during backpropagation. Multi-spectral detection results are presented in Table 6. According to the results, convolution fusion block should be used in other testing examples as a primary fusion block.

Table 6
Comparison of multi-spectral pedestrian detectors in terms of accuracy and inference speed

Method	Log-average miss rate	GPU (FPS)
SSD (ResNet-18) RGB	61.3%	79
SSD (ResNet-18) Sum Fusion	58.3%	56
SSD (ResNet-18) Max Fusion	55.3%	57
SSD (ResNet-18) Convolution Fusion	**47.4%**	**53**
faster R-CNN (VGG16) [13]	36.99%	<7

High miss rate of the final architecture can be explained by the insufficient number of training samples in KAIST dataset. Also, faster R-CNN [13] architecture was fine-tuned on another dataset with daytime images of pedestrians before training on the multi-spectral dataset, which improved test results. The primary goal of this model comparison is to show the significant gap between the standard RGB detector and a multi-spectral one. Also, the designed model can be used with any additional datasets for human detection tasks, e.g. for office lighting control systems based on human detection. The example of detection results are presented in Fig. 15.

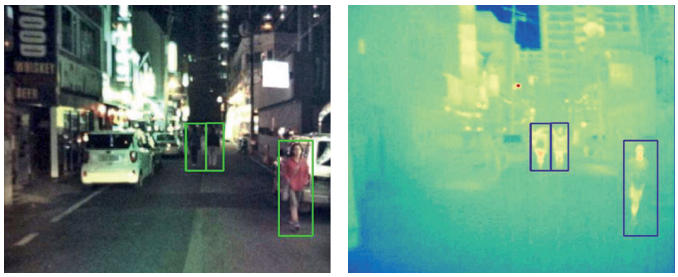

Fig. 15. The example of detection results with multi-spectral SSD architecture on KAIST dataset

3.3.2. Multi-spectral U-Net for semantic segmentation. Remote sensing is another interesting domain with multi-spectral data. A significant amount of satellite imagery has given a radical improvement of planet understanding. It has enabled for researchers to achieve better results in various applications from mobilizing resources during disasters to monitoring effects of global warming. What is often taken for granted is that advancements such as those have relied on labeling and feature construction e.g. of building footprints and roadways, fully manually or through imperfect semi-automated methods.

The dataset from DSTL Satellite Imagery Feature Detection competition [25] provides labels and various satellite multi-spectral images, which will be used for evaluation of fusion architectures for semantic segmentation of buildings. This dataset contains 60 images which cover 1×1 km area. Images, provided in different bands, such as RGB, full multi-spectral (400–1040 nm), and short-wave infrared band (1195–2365). The full range of bands is presented in Fig. 16. In present work only RGB and near infrared bands are used.

The core idea of multi-spectral extension is the same as for object detection: we add additional subnetwork in a siamese way, which utilizes near-infrared channel of the multi-spectral band. The multi-spectral extension for U-Net [14] architecture based on convolution fusion is presented in Fig. 17. The loss function for this task is binary cross-entropy, which provides per-pixel optimization for input images. All weights of both architectures were optimized with Adam [19] optimizer, with learning rate 0.001. The standard architecture was trained in

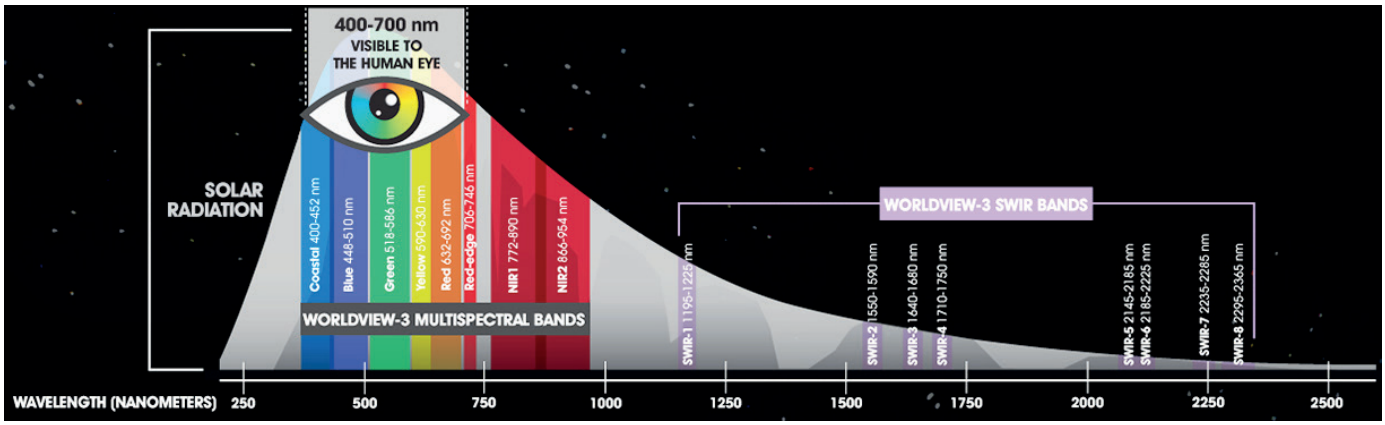

Fig. 16. General overview of spectral bands that are used in remote sensing [38]

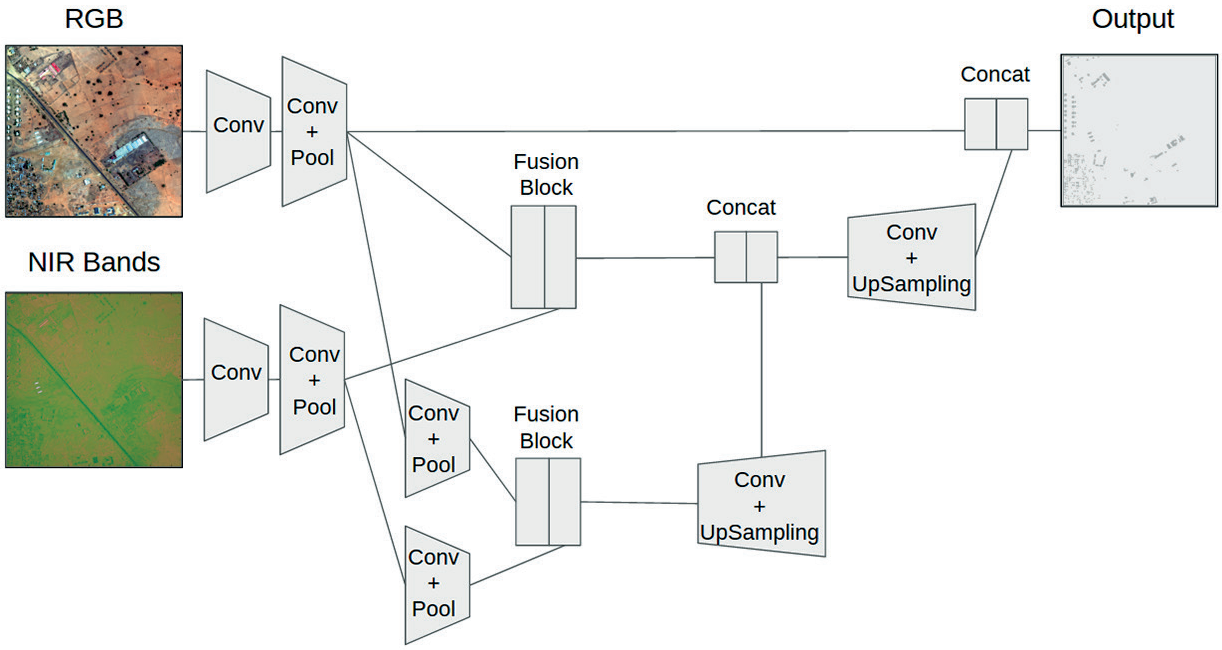

Fig. 17. Multispectral U-net based architecture for semantic segmentation. Only first two layers are presented, while full network utilized the same number of down-sampling stages as original U-net

the same way as the architecture with multi-spectral extension. Comparison results on the test set are presented in Table 7. Evaluation metric is Jaccard index, also known as intersection over union, which can be interpreted as similarity or diversity measure between a finite number of sets.

Table 7
Semantic segmentation of satellite images

Method	Intersection over Union
U-NET	0.516
Multispectral U-NET	0.5784

Intersection over union is used to evaluate how well neural network maps predicted mask with ground truth label mask for building class. To improve predicted results several morphological operations were used, providing better building shapes. We used the most common morphological operations such as erosion, dilation, opening, and closing. The difference between them is showed in Fig. 18.

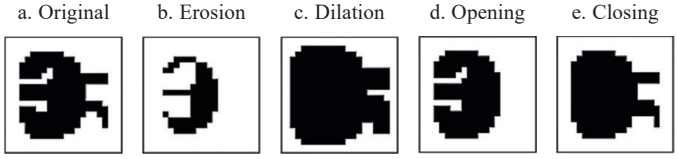

Fig. 18. Different morphological operation for post-processing stage of semantic segmentation [39]. Closing operation helps to fill holes inside predicted building significantly improving final score

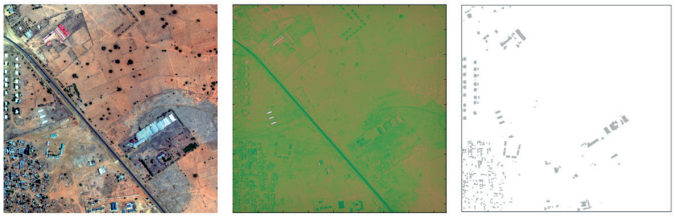

Fig. 19. Semantic segmentation results using multi-spectral U-Net architecture. Left: RGB satellite input image. Middle: Infrared satellite input image. Right: Semantic segmentation results for buildings

3.3.3. Air quality prediction using heterogeneous data. In this subsection multi-modal architecture is designed for air-quality prediction. Specifically, a neural network is designed to predict CO level in the Moscow city. In this setting, convolution neural network performs approximation of mesoscale atmospheric model for 1-hour prediction. This model (so-called full-physics model) describes transport, microphysics and chemistry processes in the atmosphere with indirect and nonlinear effects. Physical block is based on a full number of hydrodynamic equations. The chemical block contains about 200 reactions. This model is similar to numerical weather modeling. It is well known that such models require a lot of computational resources at each time step. Thus usually powerful clusters of powerful machines are used for simulations and in practice we need to decrease computation time of the model. We can consider this problem as an approximation problem (also known as regression problem), which arises quite often in industrial design, and solutions of such problems are conventionally referred to as surrogate models [27]: we collect

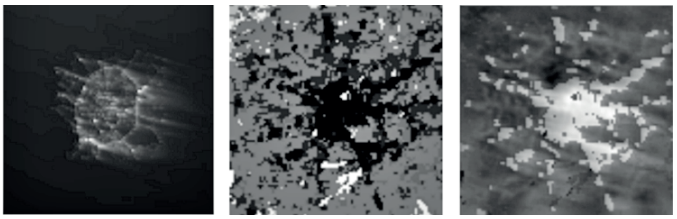

Fig. 20. Input data for air quality prediction network. Left: CO concentration. Middle: Land-use index. Right: Surface temperature

input-output data from simulations, construct a surrogate model and use it instead of full-scale simulations. The most common application of surrogate modeling in engineering is in connection to engineering optimization [26]. Input data is presented in Fig. 20, it is provided as images with information about CO concentration, land-use index and surface temperature. Hence, we can construct a surrogate model for this task using the same architecture as previously designed multi-spectral models. We estimate accuracy of the surrogate model via the root-mean-square loss function (RMSE).

Table 8
Predicted rmse error for different fusion blocks

Fusion block type	RMSE
Sum	1.25
Max	0.93
Convolution	0.65

Multi-modal architecture for air quality prediction is presented in Fig. 21. Several fusion blocks, located in different stages of the network, provide a fusion of various feature maps, corresponding to different types of input data.

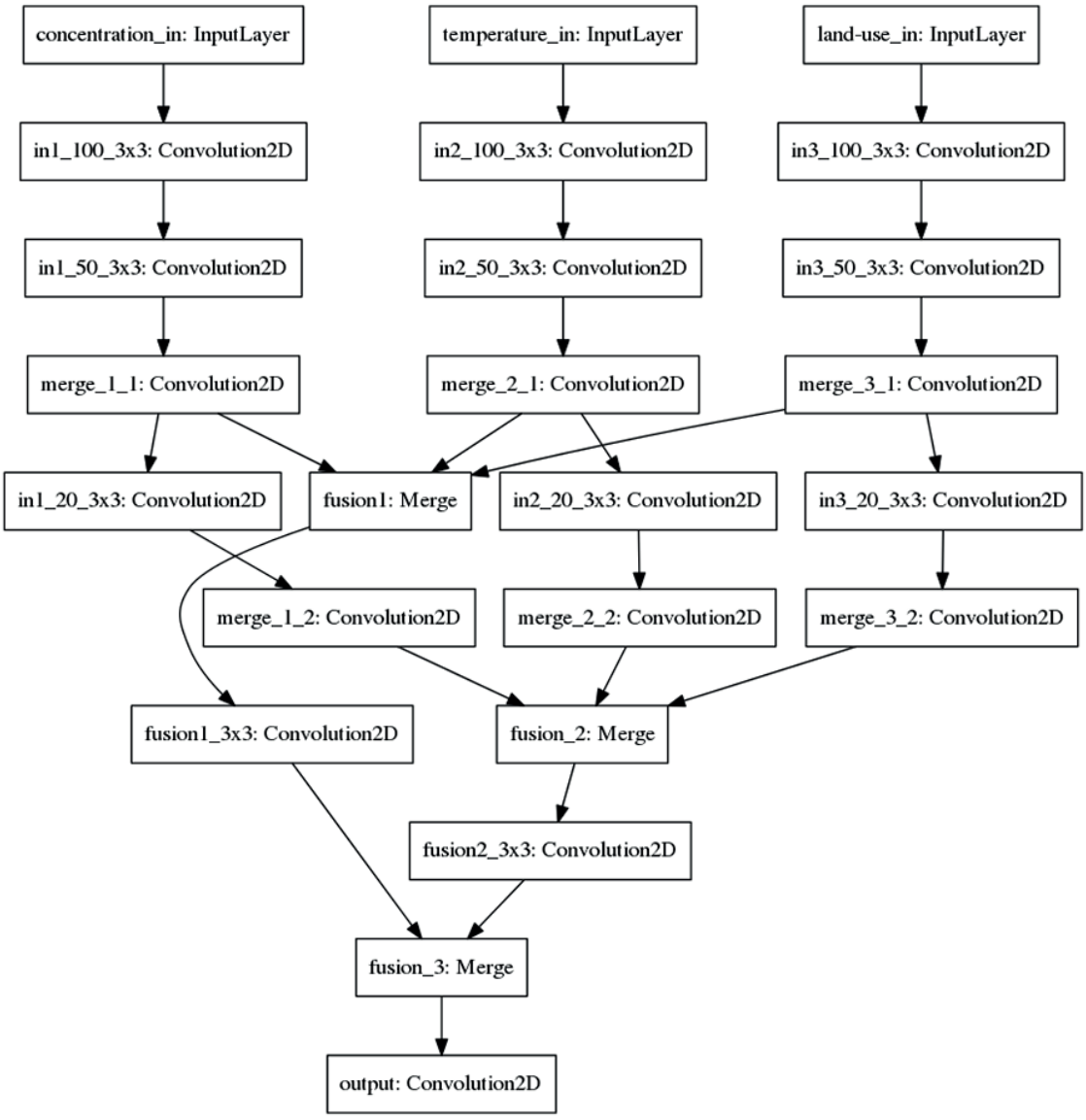

Fig. 21. Multi-modal architecture structure used for air quality prediction. Convolution layers with different number of filters are used. Pooling layers are not used since we need to preserve the same spatial dimension of feature maps

4. Implementation details

4.1. Deep learning framework. All presented experiments in this work were made with help of MXNet framework [31], in Python programming language. MXNet perfectly fits with the objective of embedding system computing.

4.1.1. Hyper-parameters setting. For successful training, procedure hyper-parameters should be correctly configured. For all experiments, hyper-parameters were manually set, according to training logs about loss function behavior and standard best practices. The initial parameters are learning rates, L2 regularization strength, mini-batch size, loss function and type of optimizer. The Adaptive Moment Estimator (Adam) was chosen for all experiments. Similar to most other optimizers, it calculates an adaptive learning rate for all parameters in the network. By storing an exponentially decaying average of past gradients, it has been shown that Adam performs better than other algorithms such as RMSProp [19]. The resulting hyper-parameters are presented in Table 9.

Table 9
Used hyper-parameters for different models

Experiment	Learning rate	Regularization	Batch size	Loss function
Fast SSD (ResNet-18)	4×10^{-2}	10^{-5}	32	cross entropy + L1
Multi-spectral SSD	10^{-3}	10^{-5}	16	cross entropy + L1
U-Net	10^{-2}	10^{-3}	10	binary cross entropy
Multi-spectral U-Net	10^{-3}	10^{-3}	8	binary cross entropy

4.1.2. Hardware. The Intel Core i5 CPU with 8GB of memory was used in earlier stages of this work, resulting in impractical time consuming training process. Eventually, a single GPU NVIDIA GeForce Titan X was used resulting in massive speed increase. The massive parallelization in GPUs is a key that allows building fast training pipelines. Final training time for object detection models were approximately 1–1.5 days and for segmentation 2–2.5 days.

5. Conclusions

The results of Section 4 have much scope for improvement:

- **Improving the dataset.** In fact the dataset [9] used for multi-spectral object detection is fairly small, and for a specific task, e.g. multi-spectral human detection in the office environment, it is highly recommended to perform data acquisition process and re-train proposed convolutional neural network architectures. During the image acquisition process it is necessary to draw attention to image registration, because even small misalignments in multi-spectral-visible image pair lead to significant decrease of detection accuracy. Also, more complex data augmentation strategies can be used. Concerning segmentation task, publicly available resources of satellite images can be leveraged at a larger scale, to obtain more representative satellite imagery of different locations for training the pipeline.
- **Detection pipeline improvements.** According to the results, only near real-time performance was achieved on embedded systems. Hence, it is necessary to improve the quality of base feature extractor, non-maximum suppression, and default box generation strategy. To improve features quality produced by current ResNet-18 feature extractor, it is possible to use collective residual unit networks, which use collective tensor factorization, described in [32]. Also, it is necessary to try recently presented neural network architectures specifically designed for embedded systems such as [15]. Also, non-maximum suppression can be integrated into the learning pipeline as an additional neural network as discussed in [33]. Another possibilities could be to use sparse convolutions [34] to process data with multiple modalities, provided in different channels, to use ensembles for increasing accuracy and robustness of results [35], as well as more efficient heuristics for initialization of neural network parameters [36]. In case of surrogate models construction for data with multi-model input so-called variable fidelity surrogate modeling approaches [28–30] and learning with privileged information can be used [37].
- **Hyper-parameters tuning.** Since the manual approach was used for hyper-parameters tuning in most of the experiments, it is highly encouraged to explore algorithmic approaches to exploring hyper-parameter space.

A primary motivation for undertaking this work was to gain an understanding for designing, implementing and evaluating a deep learning pipeline with the focus on heterogeneous data, e.g. infrared images.

Thus as a result we defined a set of computer vision problems, including multi-spectral object detection for human detection, semantic segmentation of satellite images for building identifications. In particular, we analyzed and compared object detection algorithms, including analysis how such algorithms can be used for multi-spectral detection on embedded systems. We presented necessary steps for launching state-of-the-art single object detection algorithms on low-powered devices. Single Shot Multibox detection framework (SSD) was successfully adopted for the near real-time inference on Raspberry Pi, which was used as a baseline embedded system. We discussed implementation details, describing the architecture, used deep learning framework and evaluation results for the estimation of multi-modal features influence on computer vision task. Convolution fusion block, as a key element of multi-spectral architectures, showed the best performance overall presented fusion rules. Several experiments were conducted to estimate influence of heterogeneous data sources in object detection, semantic segmentation and air quality prediction tasks. According to results, there was an improvement in the quality of proposed models, compared with standard models, which operated with normal data.

The first version of the proposed multispectral architecture was developed in the framework of the Skoltech-MIT NGP Burnaev/Solomon project.

Acknowledgements. The work was supported by the MES Russian Federation under the grant 14.756.31.0001.

References

[1] R. Gade and T.B. Moeslund, "Thermal cameras and applications: a survey", *Machine vision and applications* 25 (1), 245–262 (2014).
[2] J.RR. Uijlings and et al., "Selective search for object recognition", *Int. J. of Computer Vision* 104 (2), 154–171 (2013).
[3] R. Girshick and et al., "Rich feature hierarchies for accurate object detection and semantic segmentation", *CVPR* (2014).
[4] R. Girshick and et al., "fast R-CNN", *ICCV* (2015).
[5] J. Redmon and et al., "You only look once: Unified, real-time object detection", *CVPR* (2016).
[6] S. Ren and et al., "Faster r-cnn: Towards real-time object detection with region proposal networks", *NIPS* (2015).
[7] W. Liu and et al., "SSD: Single shot multibox detector", *ECCV*, Springer, 21–37 (2016).
[8] J. Huang and et al., "Speed/accuracy trade-offs for modern convolutional object detectors", *arXiv:1611.10012* (2016).
[9] S. Hwang and et al., "Multispectral pedestrian detection: Benchmark dataset and baseline", *CVPR* (2015).
[10] P. Dollar and et al., "Fast feature pyramids for object detection", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36 (8), 1532–1545 (2014).
[11] J. Wagner and et al., "Multispectral pedestrian detection using deep fusion convolutional neural networks", *ESANN* (2016).
[12] A. Krizhevsky, I. Sutskever, and G.E. Hinton, "Imagenet classification with deep convolutional neural networks", *NIPS* (2012).
[13] J. Liu and et al., "Multispectral deep neural networks for pedestrian detection", *arXiv:1611.02644* (2016).
[14] O. Ronneberger, P. Fischer, and T. Brox. "U-net: Convolutional networks for biomedical image segmentation", *Int. Conf. on Medical Image Computing and Computer-Assisted Intervention*, Springer (2015).
[15] A.G. Howard and et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications", *arXiv:1704.04861* (2017).
[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv:1409.1556* (2014).
[17] F.N. Iandola and et al., "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size", *arXiv:1602.07360* (2016).
[18] K. He and et al., "Deep residual learning for image recognition", *CVPR* (2016).
[19] D. Kingma and B. Jimmy, "Adam: A method for stochastic optimization", *arXiv:1412.6980* (2014).
[20] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks", *AISTATS*, 9 (2010).
[21] W. Liu, A. Rabinovich, and A.C. Berg, "Parsenet: Looking wider to see better", *arXiv:1506.04579* (2015).
[22] O. Russakovsky and et al., "Imagenet large scale visual recognition challenge", *Int. J. of Computer Vision*, 115 (3), 211–252 (2015).
[23] A. Eitel and et al., "Multimodal deep learning for robust rgbd object recognition", *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, IEEE (2015).
[24] M. Everingham and et al., "The pascal visual object classes challenge: A retrospective", *Int. J. of Computer Vision* 111 (1), 98–136 (2015).
[25] Kaggle: DSTL Satellite Imagery Feature Detection Dataset. https://www.kaggle.com/c/ dstl-satellite-imagery-feature-detection/data
[26] S. Grihon, E. Burnaev, M. Belyaev, and P. Prikhodko, "Surrogate Modeling of Stability Constraints for Optimization of Composite Structures", *Surrogate-Based Modeling and Optimization. Engineering applications*, Eds. by S. Koziel, L. Leifsson. Springer, 359–391 (2013).
[27] M. Belyaev, E. Burnaev, E. Kapushev, M. Panov, P. Prikhodko, D. Vetrov, and D. Yarotsky, "GTApprox: Surrogate modeling for industrial design", *Advances in Engineering Software* 102, 29- 39 (2–16)
[28] E. Burnaev and A. Zaytsev, "Minimax approach to variable fidelity data interpolation", *PRML, Volume 54: Artificial Intelligence and Statistics*, 54, 652–661 (2017).
[29] E. Burnaev and A. Zaytsev, "Large Scale Variable Fidelity Surrogate Modeling", *Ann Math Artif Intell*, 1–20 (2017).
[30] E. Burnaev and A. Zaytsev, "Surrogate modeling of mutlifidelity data for large samples", *J. of Communications Technology and Electronics*, 60 (12), 1348–1355 (2015).
[31] DMLC: MXNet for Deep Learning. https://github.com/ dmlc/ mxnet
[32] C. Yunpeng and et al. "Sharing Residual Units Through Collective Tensor Factorization in Deep Neural Networks", *arXiv preprint arXiv:1703.02180* (2017).
[33] J. Hosang, B. Rodrigo, and B. Schiele, "A Convnet for Non-maximum Suppression", *German Conf. on Pattern Recognition*, Springer (2016).
[34] A. Notchenko, E. Kapushev, and E. Burnaev, "Large Scale Shape Retrieval with Sparse 3D Convolutional Neural Networks", *Proc. of 6th Int. Conf. on Analysis of Images, Social Networks and Texts (AIST-2017), LNCS 10716*, 236–245 (2018).
[35] E. Burnaev and P. Prikhod'ko, "On a method for constructing ensembles of regression models", *Automation and Remote Control*, 74 (10), 1630–1644 (2013).
[36] E. Burnaev and P. Erofeev, "The Influence of Parameter Initialization on the Training Time and Accuracy of a Nonlinear Regression Model", *J. of Communications Technology and Electronics*, 61 (6), 646–660 (2016).
[37] E. Burnaev and D. Smolyakov, "One-Class SVM with Privileged Information and Its Application to Malware Detection", *ICDMW*, 273–280 (2016).
[38] World View-3 Satellite Sensor Specifications. http: //www.satimagingcorp.com/satellite-sensors/ worldview-3/
[39] S.W. Smith: The scientist and engineer's guide to digital signal processing, California Technical Pub, 1997.