Zhiliang Zeng, Ying Kin Yu, and Kin Hong Wong,"Tracking by recognition using neural network",19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2018) , Novotel Ambassador Busan, Korea.June 27-29, 2018.

Tracking by recognition using neural network

Zhiliang Zeng*, Ying Kin Yu, and Kin Hong Wong
Department of Computer science and Engineering
The Chinese University of Hong Kong
*E-mail: zlzeng@cse.cuhk.edu.hk

***Abstract*—Vision-based object tracking is a challenging problem. In the tracking process, the object is usually first recognized in a given image. Then a bounding box is used to describe the position of the target object. Normally, a vector $[x, y, width, height]$ is adopted to represent the bounding box. Under this viewpoint, the tracking problem can be treated as a regression problem if we handle the image sequence frame by frame. Due to the recent advancement in machine learning, many researchers apply neural networks to solve the visual tracking problem. This greatly improves the accuracy of bounding box prediction. Actually, the neural network based approaches are more suitable for end-to-end systems. In this paper, we propose to train and use a single neural network to tackle the tracking task. With the cropped candidate image patch as the input to the network, the output is the bounding box that indicates the target position. In our network, we first have a mask map to identify the target. It is a binary image and is divided into two classes. The positive class denotes the foreground while the negative class denotes the background. The mask map is then used for the estimation of the bounding box vector. The task now becomes an image mapping problem. We have achieved a good balance between accuracy and computational efficiency. Our tracker can reach an average speed of 178 frames per second(fps) and a maximum of 334 fps in the OTB benchmark.**

I. Introduction

Visual object tracking is challenging and also one of the most important components in computer vision. Given a sequence of images, the goal of a tracking algorithm is to recognize the target from the background and then locate its positions in the video frame by frame. To represent the location of the target object, one can use a bounding box, which is a vector denoted by $[x, y, w, h]$ or $[y, x, h, w]$. Here, (x, y) represents the position and (w, h) represents the size of the bounding box.

Although the visual tracking problem has been explored for many decades, it still remains a difficult task because the appearance of the target may change overtime. To better recognize the target, early work relies on predefined features to describe the object, for example TLD in [11]. Some non-neural network based methods also make use of the predefined features like KCF in [9]. However, the appearance may change dramatically due to deformation, illumination variations or occlusion. In addition, the creation of predefined features is dependent on human experience. It turns out that this kind of features cannot represent the target object well.

Due to the recent progress on using neural networks to solve computer vision tasks, such as edge detection [14] [25], image classification [19] [12], object detection and semantic segmentation [7] [15], and saliency detection [22], the neural network demonstrates its great ability to extract high-level features. Many researchers proposed to replace the user-defined features with high-level features in visual tracking and achieved significant improvements on the performances [2] [6] [10] [13] [21] [16] [17] [20] [5].

The state-of-the-art neural networks are able to achieve high accuracy in predicting the bounding box vector. However, a huge volume of data is required to train the networks. They are usually not available in the visual tracking problem. To address this issue, MDNet [17] uses a large number tracking video sequences as the training data to train the network. Unfortunately, the large dataset increases the training time tremendously. So the most recent neural networks require much more time to train before it can be applied to tracking. Since the neural network is originally designed for the classification problem, many neural network based methods handle visual tracking as object classification, for examples MDNet [17] and SANet [5]. To get the final bounding box prediction, an additional regression model is required to be trained to convert the prediction into a bounding box vector. This is the well-known bounding box regression method proposed in [7].

With the success of VGG [19] and AlexNet [12], the neural network has already demonstrated its ability to solve a regression problem. It is able to convert an image matrix into a vector that encodes the class information. Neural network can also be used to perform pixel-wise classification [15], which is a kind of image mapping problem. In this paper, we devise a neural network that convert an image into mask map to tackle the visual tracking problem. An overview of our approach is shown in Fig 1.

In our method, cropping is first applied to produce the candidate input patch, which consists of the target and some background context. Then we convert the bounding box vector into a mask map. The mask map is the labels of our network. During training, we first train the network to be a pixel-wise binary classifier using the input patch and mask map. After training, the network should have the ability to convert one image patch into a desired mask map. We then compute a new bounding box vector based on the predicted mask map. To refine the new bounding box vector, which is enclose it to the target bounding box, a bounding box regression method is used to update the predict vector.

Our paper is organized as follows. Section 2 discusses the background of our work. In Section 3, the theory and design methodology are described. The implementation details

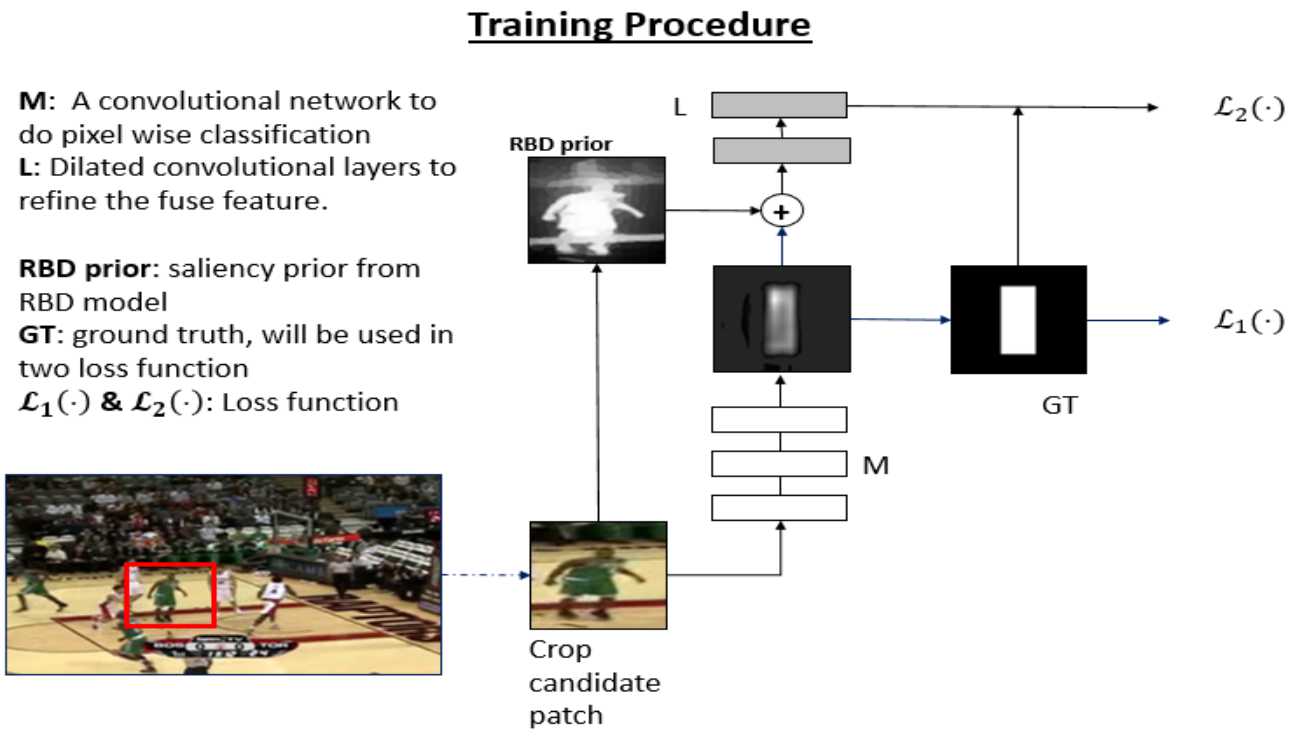


Fig. 1. Here is an overview of our training procedure. We first crop the candidate patch from the original image as input. With the initial bounding box vector, we create a mask map. To refine the mask map prediction, we combine the saliency prior computed by RBD [26] with the network prediction, then we use a sigmoid cross entropy loss to supervised the fuse feature.

and experimental results are illustrated in Section 4. The conclusion is found in Section 5.

II. Related work

The problem of visual object tracking has been extensively studied [18]. Existing tracking algorithms can be divided into two categories. One is non-neural network based methods. Some examples are the correlation filter in [9] and the machine learning method in [11]. Most of them, on the other hand, are built based on neural networks. They are embedded with high-level features extracted by the network in their systems. Here, we focus our review on this kind of visual trackers.

The predefined features are unable to describe the target object well during the whole tracking process because its appearance may change overtime. Researchers applied neural network to visual tracking so as to use high-level features to represent the target object. A number of methods in [6] [23] [13] have been proposed to tackle the visual tracking problem using neural networks. However, all these methods suffer from the same problem that they are lacking of training data. To address this issue, transfer learning has been applied in [2] [10] [16] [21]. The networks that are pre-trained on other large-scale image classification dataset are transferred to handle the visual tracking problem. Even with insufficient training data, the model can still extract good features to represent the target to maintain the accuracy.

However, the features extracted from image classification by transfer learning are not perfect for the visual tracking application due to the fundamental differences between two distinct tasks. To solve this problem, Nam annd Han [17] proposed to use a large number of annotated video sequences to train the network. They also adopted multi domain learning [3] to improve the feature extraction performance.

Actually, target tracking is a sequential problem but the convolutional network structure, which is originally designed for object recognition, cannot capture any temporal relations of features among image frames. Another network structure known as recurrent neural network (RNN) [4] is designed to learn the temporal information. Recently, Fan and Ling proposed the SANet [5] to embed the RNN structure into the convolutional layer. SANet uses RNN to separate objects from similar distracters of intra-class [5]. Indeed, SANet does not really exploit the temporal information to improve the prediction results.

The state-of-the-art networks focus on improving the prediction accuracy, such as MDNet [17] and SANet [5]. These networks can achieve a precision of 0.948 and 0.95 in OTB2013 benchmark. However, they are not computational efficient and are unable to run in real-time. In this paper, we attempt to speed up the algorithm with a little tradeoff in the tracking precision. We explore a simple network structure based on classical VGG [19] network to extract the features. We append the dilated convolutional layers at the end of our network so

that we can better capture the target by the mask map, we can get a bounding box vector based on the high activate region of the mask map, then an bounding box regression model is used to refine the extracted bounding box vector. Even without exploiting temporal information in a video sequence as in the SANet, we can still get an accurate result using the tracking by recognition framework.

III. Theory

Some previous solutions for the image classification problem [19] [12] show that neural network is good at handling the regression problem. Neural network is also suitable to learn a mapping between two images [15]. Inspired by these researches, we devise a neural network that do the image mapping to tackle the visual tracking problem.

To achieve the goal, we first construct a mask map, which is a binary map, from the bounding box vector. One denotes the positive class while zero represents the negative class. It is then used as an image label. To improve the accuracy of trained network, cropping is applied to reduce the complexity of the background, combining the saliency prior to better capture the target location and size. We build a network consisting of three convolutional layers to perform feature extraction and image mapping. After acquiring the predicted mask map, it is used to estimate the bounding box vector.

The whole process is made up of two phases. Firstly, the network is trained using the first image given the bounding box vector, which represents the selected target. After training, the network is ready to produce the mask map with the following image frame that encodes the location and size of the target object. In the tracking phase, we use the previous bounding box vector to crop the next frame to acquire the input image, which is then fed into the pre-trained model in phase one. A proper mask map is generated and is used to estimate the bounding box vector, then a bounding box regression model is used to refine the bounding box vector. Since the pipeline is relatively simple, it is computational efficient in both of the training and tracking phase.

A. Producing the candidate patch

Image cropping is applied to produce a candidate patch at the beginning of the tracking process. The candidate patch is then input to our network. The reason for using an image patch as input rather than the whole image is that the target usually occupies a small region in the original image. Recognizing the target in a such a situation is more difficult and require more computation time. On the other hand, if we first crop a patch of image that contains mainly the target and a little portion of the background, the complexity of recognizing the target can be greatly reduced. We only require to distinguish the target from the background without explicitly locating the target in a small part of the original image.

The cropped region in the current image frame is predicted based on the bounding box vector in the last frame. We assume that the target motion is relatively small. The location of the target in the next frame is not far away from its last bounding box position. To make sure that the candidate patch contains the target and some background context, a scalar value is applied to enlarge the bounding box area. In this way, the desired image patch can be extracted.

The details of our method is as follows. We first assume to have an initial bounding box vector, i.e. $[x, y, w, h]$. We use equation 1 below to compute the cropping area.

$$\begin{aligned} crop\_width &= \sigma \times w \\ crop\_height &= \sigma \times h \end{aligned} \tag{1}$$

Here, σ is a scalar used to increase the size of the cropping area. Since it is easier to handle input of the same size in a neural network, the bilinear resize method is applied to resize the cropped image patch. The resized image has a size of $[batch\_size, height, width, channel]$, which is $[1 \times 128 \times 128 \times 3]$ in our experiment.

B. Combind saliency prior

Since we use a shallow network here, to better train the network, we also try to combined with saliency map. We use RBD model [26] to extract the saliency map as prior and feed it into the network, combinding the network prediction and the saliency prior, then we append two context layer to refine the fuse map, after refined, the new predict mask map will better capture the target location. Based on the target location in the mask map, we update the previous bounding box, which mean we replace the $[x, y]$ of the previous bounding box with current target location, so that the updated bounding box enclose the target capture by the mask map.

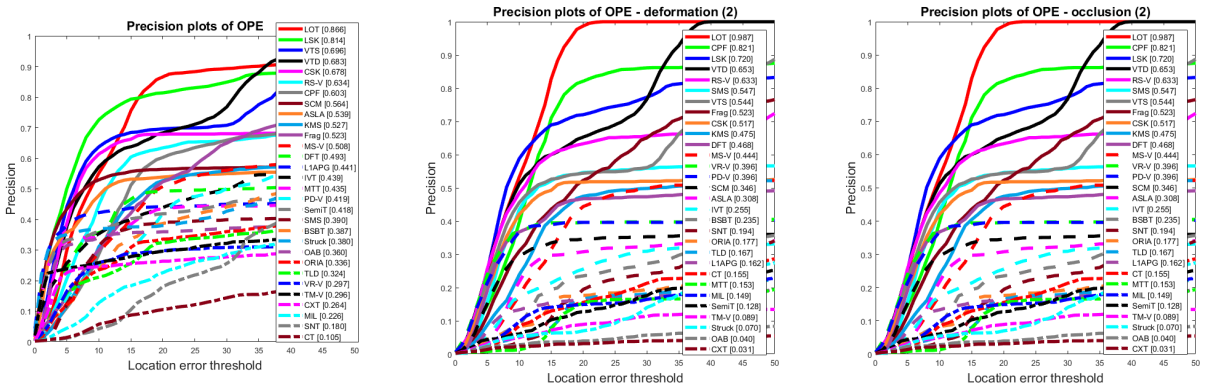
C. Bounding box regression

The bounding box regression model is first introduced in the object detection problem [7]. It has been extended to visual tracking tasks [17] [5], in which the bounding box is refined to tightly enclose the target. We use exactly the same training method which proposed on [7].

The difference from the one in [7] in a way that our network does not require any region proposal to enclose the given ground truth. Instead, we make use of the updated bounding box predicted from the last frame and is described in equation 2 below.

$$p_x^f = p_w^{f-1} \times x_L + p_x^{f-1} \tag{2}$$
$$p_y^f = p_h^{f-1} \times x_L + p_y^{f-1} \tag{3}$$
$$p_w^f = p_w^{f-1} \times exp(x_L) \tag{4}$$
$$p_h^f = p_h^{f-1} \times exp(x_L) \tag{5}$$

Here, $P^{f-1} = [p_i^{f-1}, where\ i = x, y, w, h]$ denotes the predicted bounding box vector from the last frame indexed by $f-1$. x_L denotes the feature from the last layer of the network. No activation function is applied on feature x_L. So it is still a linear regression problem from equation 2.



(a) (b) (c)

Fig. 2. Precision plots showing the evaluation results of our method compared to the others. SNT is the shortform of our algorithm. We choose the performance scores in the Precision Plots of OPE at the location error threshold of 20.

D. Network Structure

Our neural network contains three convolutional layers as the feature extraction part. It is similar to the previous approaches in [17] [5]. The task to convert an image patch into a mask map in our network is relatively simple. It does not require too many neurons to achieve the goal or otherwise it is a waste of computation resources. The previous networks contain two [5] or more fully connected layers [17] However, the fully connected part has more free parameters to be computed than the convolutional layers. This in turn increases the network storage requirement. Our network is used to predict mask map, to make the network more computational efficient, we don't use any fully connected layer. This is a tradeoff between speed and accuracy.

E. Fuse loss

In our network, we have to compute two loss values to optimize the network towards ground truth. For image mapping, we use the classical sigmoid cross entropy loss function to compute the loss value as expressed in equation 6. For the refining part, we also use the sigmoid cross entropy loss, but the input feature is combinded with prior map, which is expressed in equation 7.

$$\begin{aligned}\mathcal{L}_{mapping} = & -m \cdot log(z_L) \\ & -(1-m) \cdot log(1-z_L)\end{aligned} \tag{6}$$

Here, m denotes the mask map label. z_L represents the output from the last convolutional layer of the network.

$$\begin{aligned}\mathcal{L}_{refine} = & -m \cdot log(z_{L2}) \\ & -(1-m) \cdot log(1-(z_{L2}))\end{aligned} \tag{7}$$

Here, $z_{L2} = \sigma((z_L + rbd\_prior), \theta)$, combined with the saliency prior computed by RBD model, then feed it to the dilated convolutional layer, which represented it as function $\sigma(\cdot)$, θ denoted the parameters, to refine the final mask map prediction.

Our final loss function is shown in equation 8.

$$\mathcal{L}_{final} = \mathcal{L}_{mapping} + \mathcal{L}_{refine} \tag{8}$$

IV. Experiments and Results

A. Implementation details

The proposed method was implemented in Python based on tensorflow framework [1]. We used three convolutional layers for feature extraction and each convolutional layer was appended with one maximum pooling layer to reduce the image resolution. The learning rate was initialized as $1e-3$. We adopted the training data from the popular OTB benchmark [24]. There are totally 100 video sequences. To simplify the training procedure, we made use of the first video frame for training and the bounding box vector as the label. The major difference of our network and the state-of-the-art methods, such as SANet [5], MDNet [17], is that they used the whole image sequence for training to try to achieve the highest model accuracy. On the other hand, we only adopted the first frame of these video footages, i.e. 100 frames, in the training stage to maximize the computational efficiency.

B. Evaluation results on OTB benchmark

Precision plots are shown in figure 2 to compare the results of our tracker with other existing methods. Since our goal is to optimize the computational efficiency, the accuracy of our tracker was not as good as some of the state-of-the-art techniques. The proposed algorithm still performed better than a number of traditional methods. We can conclude that we have achieved a good balance between precision and speed.

When performing a forward pass to track a specific target, our approach was able to achieve an average of 178 frames per second (fps). Our method could reach a maximum frame rate of 334fps for some cases in the OTB dataset [24]. Note that we also try to combine a saliency prior which is computed by RBD model [26] to refine the mask map, since the RBD model is not a real time model, so the efficiency will reduce to $\frac{1}{4}$ fps. Methods like MDNet [17], SANet [5] and BranchOut [8] were indeed very slow. They could hardly operate at a speed over

1fps as reported in the literature. This is actually a tradeoff between algorithm speed and accuracy.

V. Conclusion

We present a vision-based object tracker with a simple neural network architecture in this paper. Here, we mainly concentrate on the improvement of computational efficiency on tracking rather than the prediction accuracy. This is different from the goal of previous researches that make use of convolutional neural networks. Beside the simple structure, a small dataset is adopted to reduce the training time. This is actually a tradeoff between algorithm speed and accuracy. We have achieved an average image frame rate of 178fps. Although our algorithm underperformed the state-of-the-art methods in the literature, it was still more accurate than a number of traditional approaches. In the future, we will further improve the precision by limiting the computation time.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 4

[2] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016. 1, 2

[3] L. Duan, I. W. Tsang, D. Xu, and T.-S. Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *ICML*, 2009. 2

[4] J. L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990. 2

[5] H. Fan and H. Ling. Sanet: Structure-aware network for visual tracking. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, volume 00, pages 2217–2224, July 2017. 1, 2, 3, 4

[6] J. Fan, W. Xu, Y. Wu, and Y. Gong. Human tracking using convolutional neural networks. *IEEE Trans. Neural Networks*, 21(10):1610–1623, 2010. 1, 2

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1, 3

[8] B. Han, H. Adam, and J. Sim. Branchout: Regularization for online ensemble tracking with cnns. In *CVPR*, 2017. 4

[9] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(3):583–596, 2015. 1, 2

[10] S. Hong, T. You, S. Kwak, and B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. In *ICML*, 2015. 1, 2

[11] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(7):1409–1422, July 2012. 1, 2

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 3

[13] H. Li, Y. Li, and F. Porikli. Deeptrack: Learning discriminative feature representations online for robust visual tracking. *IEEE Trans. Image Processing*, 25(4):1834–1848, 2016. 1, 2

[14] Y. Liu, X. H. Ming-Ming Cheng, K. Wang, and X. Bai. Richer convolutional features for edge detection. In *CVPR*, 2017. 1

[15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1, 3

[16] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015. 1, 2

[17] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016. 1, 2, 3, 4

[18] R. P. Pflugfelder. Siamese learning visual tracking: A survey. *CoRR*, abs/1707.00569, 2017. 2

[19] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1, 2, 3

[20] R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. In *CVPR*, 2016. 1

[21] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *ICCV*, 2015. 1, 2

[22] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan. Saliency detection with recurrent fully convolutional networks. In *ECCV*, 2016. 1

[23] N. Wang and D.-Y. Yeung. Learning a deep compact image representation for visual tracking. In *NIPS*, 2013. 2

[24] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 4

[25] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, pages 1395–1403, 2015. 1

[26] W. Zhu, S. Liang, Y. Wei, and J. Sun. Saliency optimization from robust background detection. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2814–2821, June 2014. 2, 3, 4