Object Detection and Localization Using Deep Convolutional Networks with Softmax Activation and Multi-class Log Loss

AbdulWahab Kabani and Mahmoud R. El-Sakka(✉)

Department of Computer Science, The University of Western Ontario,
London, ON, Canada
{akabani5,melsakka}@uwo.ca

Abstract. We introduce a deep neural network that can be used to localize and detect a region of interest (ROI) in an image. We show how this network helped us extract ROIs when working on two separate problems: a whale recognition problem and a heart volume estimation problem. In the former problem, we used this network to localize the head of the whale while in the later we used it to localize the heart left ventricle from MRI images. Most localization networks regress a bounding box around the region of interest. Unlike these architecture, we treat the problem as a classification problem where each pixel in the image is a separate class. The network is trained on images along with masks which indicate where the object is in the image. We treat the problem as a multi-class classification. Therefore, the last layer has a softmax activation. Furthermore, during training, the mutli-class log loss is minimized just like any classification task.

Keywords: Localization · Detection · Recognition · Artificial neural networks · Deep learning · Convolutional neural network · Image classification

1 Introduction

A Convolutional Neural Network (convnet or CNN) is a special type of neural network that contains some layers with restricted connectivity. Such networks were introduced a long time ago [5] and achieved excellent results on the famous MNIST data set [9]. However, it took them few years to outperform the state of the art methods in visual recognition challenges. Currently, CNNs can produce the state of the art performance in many classification tasks. Such a success is driven by the availability of large training data sets [3,13], powerful hardware, regularization techniques such as Dropout [7,17], initialization methods [6], ReLU activations [10], and data augmentation. Since 2012, many networks that can perform classification were introduced [8,15,18].

Typically, CNNs have been used for classification tasks. However, they can also be used for detection and localization [4,12,14,16,19]. A common way to

A. Campilho and F. Karray (Eds.): ICIAR 2016, LNCS 9730, pp. 358–366, 2016.
DOI: 10.1007/978-3-319-41501-7_41

localize an object in the image is to treat the problem as a regression task. In this setting, a bounding box center (or top left corner) is regressed along with the height and width of the bounding box. In this paper, we propose an architecture that treats the problem as a classification task. Pixels are classified as to whether they are inside the bounding box or not. The network is trained on the images and on masks. The mask of one image has the same size as the image and indicates where the bounding box is (or the pixels that belong to the target object are). We experiment with two types of images: right whale aerial images and heart MRI images.

The localization we describe in this paper helped us improve the classification performance on these data sets. In general, we find localization to be very helpful when the number of training images is very small. This is because localizing a region of interest reduces the number of degrees of freedom by removing background pixels that are unrelated to the classification task. This also reduces the amount of time needed for training the classifier. Furthermore, removing background pixels and training a classifier to classify only the region of interest is very important when the amount of RAM in the GPU card is limited. Image subsampling can be used to reduce the size of the image in order to fit it in the GPU RAM. However, subsampling may not be suitable in images where the region of interest (ROI) is very small with respect to the image size. This is because image subsampling may shrink the ROI to a point where it is difficult for the classifier to learn useful features. On the other hand, localization can reduce the input image size by extracting the ROI and removing the unrelated background pixels.

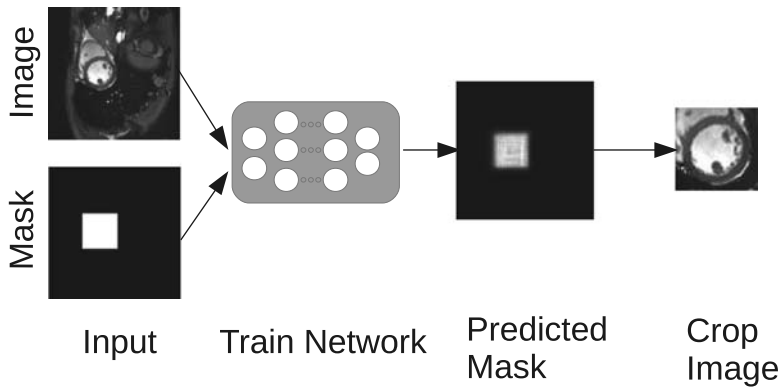

Fig. 1. Model overview: A set of training images along with masks are used to train the network. Once the network is trained, it can be used to predict a mask which identifies the location of the region of interest. Finally, this predicted mask is thresholded (using otsu [11]) and used to crop the image.

Figure 1 shows an overview of the method. Training images and corresponding masks are used to train a neural network. Once the network is trained, the

network can be used to predict masks. Once the predicted mask is thresholded, it can be used to find the region of interest in the original image. In Sect. 2, we introduce the architecture of the network. We describe how we trained the network in Sect. 3. In Sect. 4, we present our experimentation with two data sets: a north american right whale data set and short axis heart MRI data set. We conclude our work in Sect. 5.

2 Architecture

The architecture of the localization network (shown in Fig. 2) is similar to many classification networks. The main difference is the last layer, which is a flattened 2D mask predicted by the network. The training images along with their corresponding masks are re-sized to a certain size. In general, we resize the images by taking the mean of the width and height of images in the data set. We resized the images to 128×128 and 112×224 for the MRI images and the whale images, respectively. The input image and the corresponding mask should have the same size. Furthermore, the last layer is a fully connected with $height \times width$ neurons. Reshaping the output layer to 2D produces the predicted masks. The network parameters are summarized in Table 1.

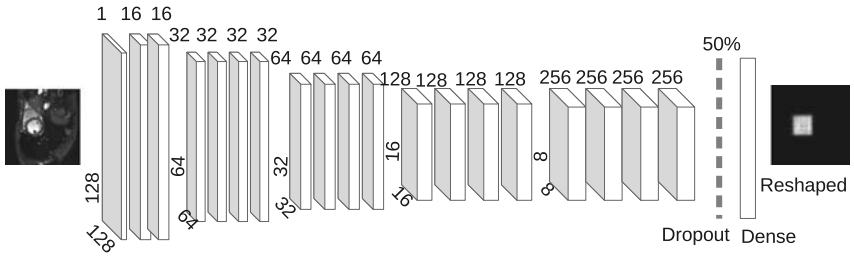

Fig. 2. Localization architecture: The input of the architecture is an image of size 128×128 for the Heart MRI images (described in this Figure.) Note that for the whale images (not shown in this Figure), we used a different input size 112×224. Besides the difference in input size, the number of layers and other parameters are the same for both data sets. In this figure, the output of the network is a layer with $128 \times 128 = 16384$ possible classes. The output layer is simply a flattened mask and reshaping this layer gives us back the predicted mask. The pixels with the highest intensities represent the location of the region of interest.

Each convolutional layer is followed by ReLU activation [10]. The output layer has a softmax activation to ensure that the sum of all pixels in the predicted mask is 1 and that the value of one pixel is between 0 and 1. Maxpooling is used to detect features at different scales. The network is regularized with a 50 % dropout rate.

The most important layer in this network is probably the output layer. It is crucial to design the network such that the number of parameters in the last

Table 1. Localization network architecture: this tables shows the type of layer, the kernel size, shape of the layer, and the number of parameters in each layer.

No	Type	kernel	Shape	Parameters
1	Convolution	(7,7)	(16, 128, 128)	800
2	Convolution	(7,7)	(16, 128, 128)	12,560
3	MaxPooling	_	(16, 64, 64)	0
4	Convolution	(5,5)	(32, 64, 64)	12,832
5	Convolution	(5,5)	(32, 64, 64)	25,632
6	Convolution	(5,5)	(32, 64, 64)	25,632
7	Convolution	(5,5)	(32, 64, 64)	25,632
8	MaxPooling	_	(32, 32, 32)	0
9	Convolution	(5,5)	(64, 32, 32)	51,264
10	Convolution	(5,5)	(64, 32, 32)	102,464
11	Convolution	(5,5)	(64, 32, 32)	102,464
12	Convolution	(5,5)	(64, 32, 32)	102,464
13	MaxPooling	_	(64, 16, 16)	0
14	Convolution	(5,5)	(128, 16, 16)	204,928
15	Convolution	(5,5)	(128, 16, 16)	409,728
16	Convolution	(5,5)	(128, 16, 16)	409,728
17	Convolution	(5,5)	(128, 16, 16)	409,728
18	MaxPooling	_	(128, 8, 16)	0
19	Convolution	(5,5)	(256, 8, 8)	819,456
20	Convolution	(5,5)	(256, 8, 8)	1,638,656
21	MaxPooling	_	(256, 4, 4)	0
23	Flatten	_	4096	0
24	Dropout	_	4096	0
25	Dense	_	16384	67,125,248

layer is as large as possible but also appropriate for the GPU RAM available. Since the output layer is fully connected, it has the largest number of parameters in the network. It is important to make sure that the number of pixels in the input image and the input mask equals the number of units in the output layer.

3 Training and Localizing

The network is trained on two types of images: north atlantic whale images [2] and heart MRI images [1]. We minimize the mutli-class logloss (categorical cross-entropy). Initially, the learning rate is set at a relatively high value 0.01 and gradually reduced if the validation loss does not improve. Because the last layer in the network is a softmax layer, the pixels of the predicted mask are probabilities.

Therefore, it is crucial that pixels of the input mask are also probabilities (sum up to 1 and their range is between 0 and 1). Therefore, we standardize each pixel in the input mask by Eq. 1:

$$y_{ij} = \frac{pixel_{ij}}{\sum_{i=1}^{H}\sum_{j=1}^{W} pixel_{ij}} \tag{1}$$

where y_{ij} is the normalized pixel value such that $y_{ij} \in [0, 1]$ and $\sum_{i=1}^{H}\sum_{j=1}^{W} y_{ij} = 1$, $pixel_{ij}$ is the pixel value at row i and column j.

Once the training is complete, the output layer is reshaped to have a 2D shape. Then, this reshaped layer (our predicted mask) can be thresholded using otsu. The image is thresholded as shown in Eq. 2:

$$I(i,j)_{thresholded} = \begin{cases} 1, & \text{if } I(i,j) \geq threshold_{Otsu} \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

Then, the predicted mask is resized back to the original size of the image. Finally, the predicted mask can be used to extract the region of interest from the original image.

During training, the images are transformed in order to alleviate overfitting. These transformations are summarized in Table 2.

Table 2. Data augmentation: Random transformations along with parameters. These transformations are applied randomly to each image before sending it to the GPU. Important note: the same transformation should be applied to both the image and corresponding mask.

Transformation	Parameters
Horizontal Flip	Randomness = 50 %
Vertical Flip	Randomness = 50 %
Horizontal Shift	Up to 20 % of width
Vertical Shift	Up to 20 % of height
Gaussian Blurring	Up to $\sigma = 1$

It is very important to note that the same transformation should be applied to both the image and corresponding mask. Otherwise, the network will never converge and the training will fail. It is worth mentioning that the network can also learn the object scale if there is a variation in scale in the training data. If there is no scale variation in the training data and scale learning is desired, the training images and masks can be transformed to simulate scale variations.

4 Experiments

All experiments were ran on a laptop with GTX980M graphics with 4 GB RAM. Training the localizer takes around 160 s per epoch (around 4.5 h to train 100 epochs).

Figure 3 shows the multi-class logarithmic loss (also known as categorical cross-entropy) progress while training for both the training and validation images. The equation for this metric is:

$$logloss = -\frac{1}{n}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} log(p_{ij}) \tag{3}$$

where N is the number of images in the data set, M is the number of pixels in the mask. y_{ij} is a pixel in the true input mask i. y_{ij} has a 0 value if it corresponds to the background and a higher value if it is inside the ROI. p_{ij} is the corresponding predicted pixel value.

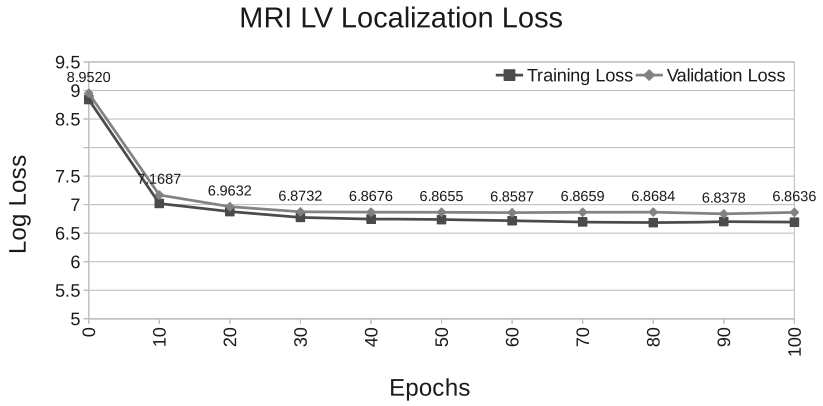

Fig. 3. The loss curves when training the network to localize the left ventricle in heart MRI images. Since the validation and training losses are very close to each other, increasing the capacity of the network is likely to improve results at the expense of longer training time. (Color figure online)

Since the validation and training losses are very close to each other, increasing the capacity of the network is likely to improve results at the expense of longer training time. It is worth mentioning that using an architecture similar to the Oxford Visual Geometry Group (VGG) net [15] can usually lead to better results at the expense of time. The VGG [15] architecture is a very deep network with small kernel sizes and it is usually used for classification. Modifying our network architecture to be similar to VGG can lead to better results at the expense of longer training times. Since localization is usually a preprocessing step before classification, we opted for an architecture that can converge fast and produce good results within a decent amount of training time. If better localization results are desired, the network can be made deeper with smaller kernel sizes.

It may be difficult to understand how good the localizer is by analyzing the log loss progress. The lower the loss, the better the localizer. However, it may not be clear how good the localizer is. Figure 4 shows how the performance

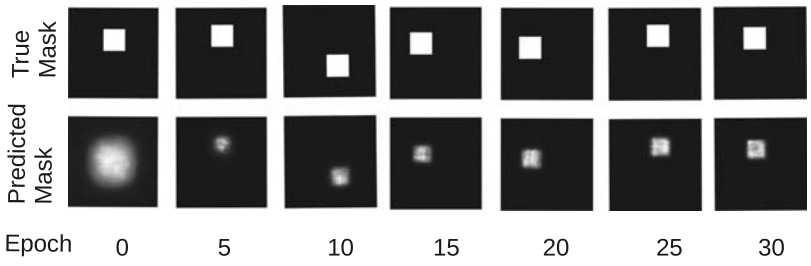

Fig. 4. This figure shows how the network performance in tracking the location of the left ventricle improves as it is being trained. The same image is shown across different epochs. Due to data augmentation, the true (and predicted) location of the left ventricle changes. At epoch 0, the network predicts the location of the left ventricle to be in the middle of the image. Later, the network gradually becomes capable at predicting the location of the left ventricle.

of the network improves while training. Figure 4 shows only one image from the validation set but due to data augmentation, the true location of region of interest changes during each epoch. At the beginning of training, the network predicts the region of interest to be in the middle. Gradually, we can see that the network is starting to predict the correct location and size of the region of interest.

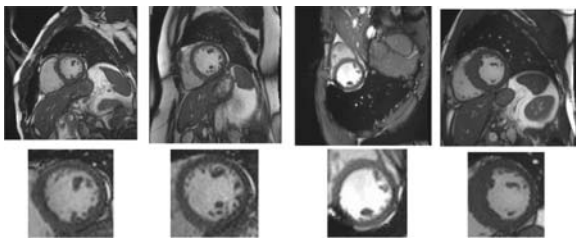

Fig. 5. A random sample of MRI images showing the heart. In the lower row, the left ventricle is localized and cropped.

Figures 5 and 6 show the result of localizing the left ventricle in MRI images and the right whale heads in right whale images. The performance of this network seems to be very robust. When scanning the validation set for wrong localization, we could not find any examples where the network completely returned the wrong region of interest. However, for the MRI images, we did notice that when the left ventricle is very small and at end-systole, some regions of interest were larger than they should be.

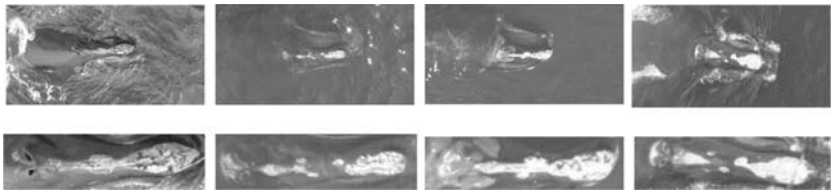

Fig. 6. A random sample of right whale images along with the resulting head crops. These head crops can be passed to a right whale classifier to recognize the whale.

5 Conclusion

We introduced a network that can be used to localize a region of interest. Unlike many localization networks, we do not regress a bounding box. Instead, the network is trained using the training images and the corresponding masks. The network predicts a mask, which is then thresholded and used to extract the region of interest from the original image. It is worth mentioning that this network can learn a region of interest of any shape (rectangle, triangle, circle, etc.). It can also be used for supervised segmentation of an arbitrary shaped object.

Acknowledgements. This research is partially funded by the Natural Sciences and Engineering Research Council of Canada (NSERC). This support is greatly appreciated. We would also like to thank kaggle, the National Oceanic Atmospheric Administration Fisheries for providing the whale data set. We would also like to thank Booz Allen Hamilton, and the National Heart, Lung, and Blood Institute (NHLBI) for providing the MRI images.

References

1. Data science bowl cardiac challenge data. https://www.kaggle.com/c/second-annual-data-science-bowl. (Accessed on 19 March 2016)
2. Right whale recognition. https://www.kaggle.com/c/noaa-right-whale-recognition. (Accessed on 19 January 2016)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 248–255. IEEE (2009)
4. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2147–2154 (2014)
5. Fukushima, K.: Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybern. **36**(4), 193–202 (1980)
6. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: International Conference on Artificial Intelligence and Statistics, pp. 249–256 (2010)

7. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)
8. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp. 1097–1105 (2012)
9. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
10. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML 2010), pp. 807–814 (2010)
11. Otsu, N.: A threshold selection method from gray-level histograms. Automatica **11**(285–296), 23–27 (1975)
12. Szegedy, C., Toshev, A., Erhan, D.: Deep neural networks for object detection. In: Advances in Neural Information Processing Systems, pp. 2553–2561 (2013)
13. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015)
14. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv:1312.6229 (2013)
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
16. Song, H.O., Girshick, R., Jegelka, S., Mairal, J., Harchaoui, Z., Darrell, T.: On learning to localize objects with minimal supervision. arXiv preprint arXiv:1403.1024 (2014)
17. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
18. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. arXiv preprint arXiv:1409.4842 (2014)
19. Szegedy, C., Toshev, A., Erhan, D.: Deep neural networks for object detection. In: Advances in Neural Information Processing Systems, pp. 2553–2561 (2013)