Estimating Ejection Fraction and Left Ventricle Volume Using Deep Convolutional Networks

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 present a fully automated method to estimate the ejection fraction, the end-systolic and end-diastolic volumes from cardiac MRI images. These values can be manually measured by a cardiologist but the process is slow and time consuming. The method is based on localizing the left ventricle of the image. Then, the slices are cleaned, re-ordered, and preprocessed using the DICOM meta fields. The end-systolic and end-diastolic images for each slice are identified. Finally, the end-systolic and end-diastolic images are passed to a neural network to estimate the volumes.

Keywords: Localization · Detection · Cardiac MRI · Left ventricle · Automatic ejection fraction estimation

1 Introduction

Cardiologists can assess cardiac function by analyzing the end-systolic and end-diastolic volumes, and ejection fraction. These values can be manually measured by a cardiologist but the process is slow and time consuming. We introduce an automated method that can estimate these values. We developed our method on a data set with 700 training studies and tested it on 440 testing studies. The data set is compiled by the National Institutes of Health and Children's National Medical Center [1].

In general, left ventricle volume can be estimated by detecting it and segmenting its cavity [3,7,9,10]. Once the cavity (the blood pool) is segmented, the volume can be calculated by summing up the sub-volumes of all slices according to simpson's rule.

In this paper, we take a slightly different approach and estimate the volume using a convolutional neural network. A Convolutional Neural Network (convnet or CNN) is a special type of neural network that contains some layers with restricted connectivity. CNNs have been producing excellent results on many classification tasks. This is all thanks to the availability of large training data sets [2,14], powerful hardware, regularization techniques such as Dropout [6,16], initialization methods [4], ReLU activations [12], and data augmentation. Since 2012, many networks that can perform classification were introduced [8,15,17].

A. Campilho and F. Karray (Eds.): ICIAR 2016, LNCS 9730, pp. 678–686, 2016.
DOI: 10.1007/978-3-319-41501-7_76

In general, CNNs can produce excellent results on many classification tasks if large amounts of training data is available. However, this requirement can be alleviated if the data is cleaned, standardized, and transformed such that to minimize viewpoint variance. The total size of the data set described in this paper is 1140 studies (700 training studies and 440 testing studies). On this data set with only 700 training studies, we were able to estimate the volume with around +/− 15 ml error. Furthermore, we were able to estimate the ejection fraction with +/− 5 % error. The ejection fraction is one of the most important cardiac measurements since it describes how good the heart is in pumping blood.

First, we localize the left ventricle (Sect. 2). Then, the data is preprocessed, cleaned and standardized (Sect. 3). Finally, the volume of the left ventricle is estimated (Sect. 4). In Sect. 5, we present the results. We conclude our work in Sect. 6.

2 Left Ventricle Localization

For each patient, there are 8–16 short axis views (slices). These views show cross-sections of the left ventricle at different levels. Each slice contains around 30 images spanning the cardiac cycle (one heartbeat). The end-diastolic volume is the volume when the left ventricle is fully expanded while the end-systolic volume is when the left ventricle is contracted. To estimate these two volumes, for each slice, we need to identify the image with the largest blood pool area (end-diastole) and the one with the smallest area (end-systole).

In order to do that, we first localize the left ventricle and crop the image to get rid of the background pixels. We train a localization network on the training images and on the corresponding masks. The masks indicate where the left ventricle is in the image. Figure 1 shows the summary of the localization.

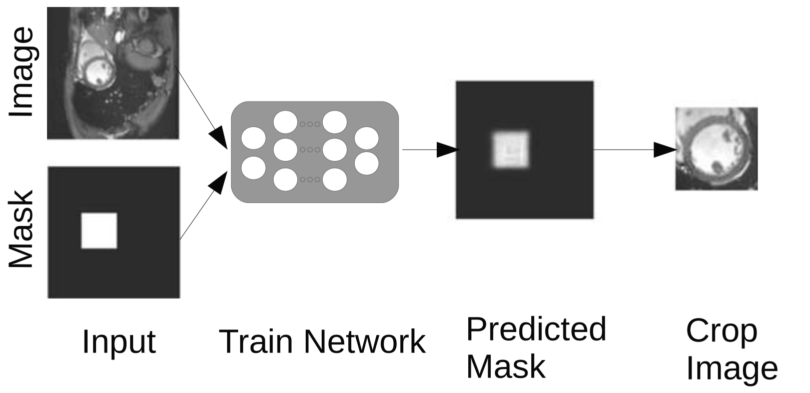

Fig. 1. Model Overview: A set of training images along with masks are used to train the network. Once the network is trained, it can be used to predict a mask which identifies the location of the region of interest. Finally, this predicted mask is thresholded (using Otsu [13]) and used to crop the image.

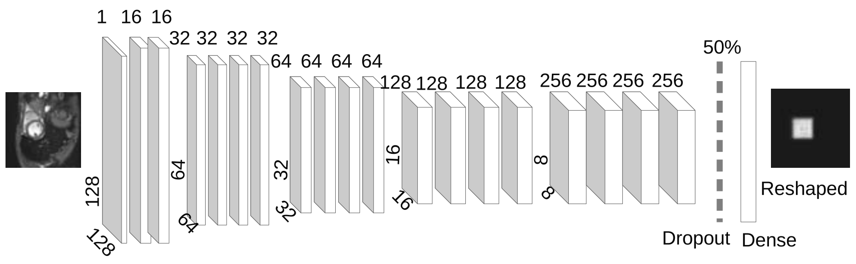

Fig. 2. Localization architecture: The input of the architecture is an image of size 128×128. In this figure, the output of the network is a layer with $128 \times 128 = 16384$ possible classes. The output layer is simply a flattened mask and reshaping this layer gives us back the predicted mask. The pixels with the highest intensities represent the location of the region of interest.

The architecture of the localization network (shown in Fig. 2) is similar to many classification networks. The image goes through many convolutional and maxpooling layers. The output layer has $h \times w$ number of units where h and w are the dimensions of the input image. Reshaping the output layer produces the predicted mask. Once thresholded, the predicted mask can be used to predict the location of the left ventricle. In the predicted mask, the pixels with high intensity values correspond to where the network predicts the left ventricle to be whereas pixels with values close to 0 correspond to background pixels.

Each convolutional layer is followed by ReLU activation [12]. 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 pixels of the input mask are supposed to be 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 $\sum_{i=1}^{H} \sum_{j=1}^{W} y_{ij}$ sums up to 1, $pixel_{ij}$ is the pixel value at row i and column j.

3 Preprocessing

Once the left ventricle is localized (Fig. 3), the meta fields are used to standarize and clean the data. First, we re-size all images using the pixel spacing and slice location meta fields. The images in this data set have different pixel spacing. This ensures that all images sizes are measured in the same units despite the difference in pixel spacing. Equations 2 and 3 ensure that the volume of the blood pool inside the left ventricle is directly proportional its area inside the image.

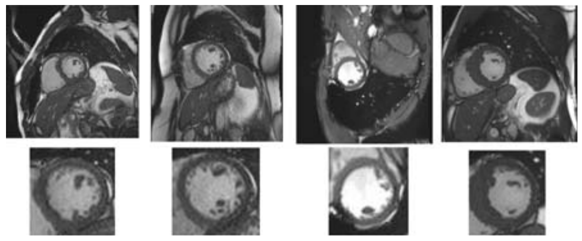

Fig. 3. A random sample of MRI images showing the heart. In the lower row, the left ventricle is localized and cropped.

$$w_{new} = w_{old} \times p_w \times \frac{\sqrt{\Delta s}}{\sqrt{10}} \tag{2}$$

$$h_{new} = h_{old} \times p_h \times \frac{\sqrt{\Delta s}}{\sqrt{10}} \tag{3}$$

where w_{old} is the old image width, w_{new} is the new image width, h_{old} is the old image height, h_{new} is the new image height, respectively. p_w and p_h are the pixel spacing for the width and height. Δs is the slice height. Since most slice heights in the data set are 10 mm, we normalize each equation by $\sqrt{10}$.

Before passing the slices to the neural network to estimate the volume, the slices needs to be standarized. Many slices are in arbitrary order in the study. Therefore, we use the meta field *slicelocation* to sort all slices. This sorted almost all slices from the top of the heart to the bottom. However, since the field *slice-location* is based on an unkown reference point, for some studies the slices were reversed (from the bottom of the heart to the top). Therefore, we trained a small neural network (Fig. 4) to predict the slice location. We mainly use this network

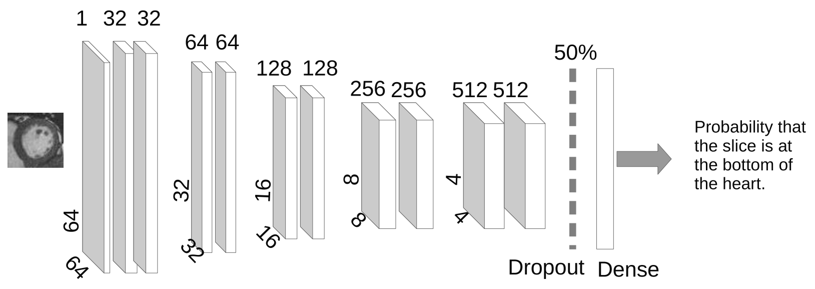

Fig. 4. Slice Localizer Network: this network outputs a probability that a certain image in a slice is located at the bottom (or base) of the heart. In other words, if the image is of a slice at the base of the heart, the network will output a high probability. If the image is showing a slice near the top of the heart, the network will output a low probability.

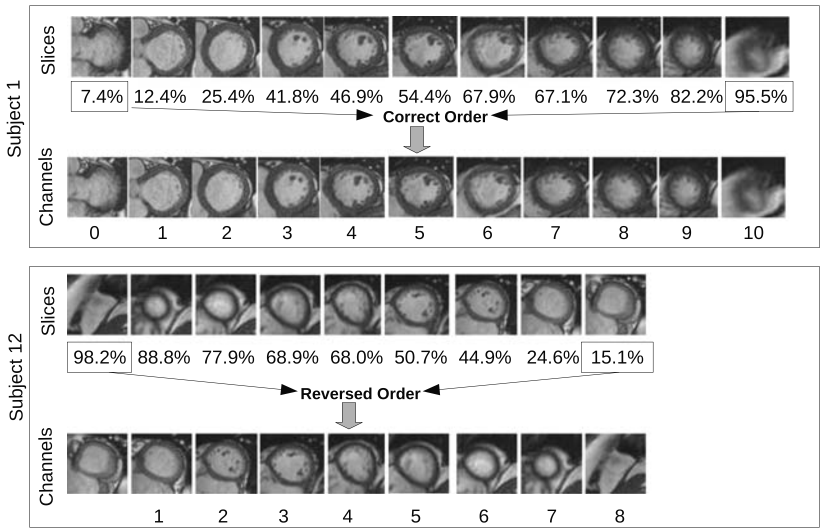

Fig. 5. Ordering the Slices: this figure shows the end-diastolic image from each slice in subjects 1 and 12. The slices of subject 1 are in the correct order (from the top of the heart to the bottom). On the other hand, the slices of subject 12 are in reverse order. The slice localizer is trained to output the probability that the image is showing the top of the heart or the bottom. For instance, in subject 12, the slice localizer network states that the image in the first slice has a 98.2 % of being in the bottom of the heart. In addition, the last slice has a 15.1 % probability of being a slice in the bottom of the heart. Therefore, we should reverse the slices to make them consistent. This process ensures that for all subjects, the slices show the heart from top to bottom.

to predict the top and bottom slices of the heart, if these appear to be in the wrong order, we reverse them.

Figure 5 shows the end-diastolic image from each slice in subjects 1 and 12. The slices of subject 1 are in the correct order (from the top of the heart to the bottom). On the other hand, the slices of subject 12 are in reverse order. The slice localizer network is trained to output the probability that the image represent a slice at the top of the heart or the bottom. For instance, in subject 12, the slice localizer network states that the image in the first slice has a 98.2 % probability of being in the bottom of the heart. In addition, the last slice has a 15.1 % probability of being a slice in the bottom of the heart. Therefore, we should reverse the slices to make them consistent. This process ensures that for all subjects, the slices show the heart slices from top to bottom. This is important because the volume estimator network (Sect. 4) accepts each slice as a separate channel and expects the slices to be ordered and consistent.

4 Volume Estimation

In Sect. 2, we localized the left ventricle in the image. After that (in Sect. 3), the images were re-sized using the pixel spacing meta field in the DICOM files to ensure that the left ventricle cavity area is consistent across all images. Furthermore, a slice localization network was used to predict the slice location. The slices were ordered so that they are consistent before passing them to the volume estimation network.

The end systole and end diastole for each slice are defined as the images with the smallest and largest blood pool, respectively. Therefore, for each slice, we threshold the images and the end systolic and end diastolic images are chosen to be the ones with the smallest and largest white pixels in one slice, respectively. The end systolic and end diastolic images from each slice form the channels of the input we pass to the volume estimation architecture.

As shown in Fig. 6, the input to the network is a set of images with size 96×96 pixels. The input has 36 channels (18 channels are for the diastole images and 18 channels for the systole images). In other words, we extract 1 systole image and 1 diastole image from each slice (a slice is a set of 30 images representing one heartbeat). We assume that the maximum number of slices possible is 18. After that, the input is passed through multiple sets of convolutional layers and maxpooling (8 convolutional layers & 1 maxpooling; 8 convolutional layers & 1 maxpooling; 4 convolutional layers & 1 maxpooling; 4 convolutional layers & 1 maxpooling; 2 convolutional layers & 1 maxpooling). The convolutional kernel size is (5,5). The activation for all convolutional layer is leaky

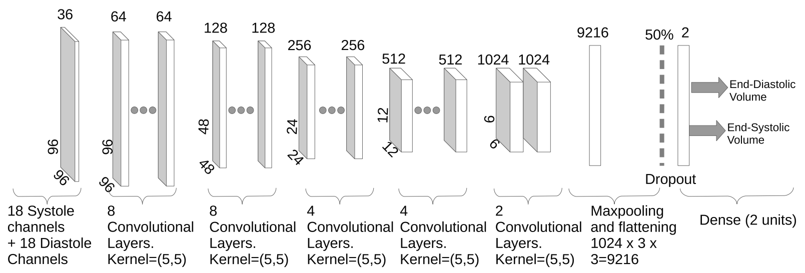
36
64
64
128
128
256
256
512
512
1024
1024
9216
50%
2
96
96
96
96
48
48
24
24
12
12
6
6
End-Diastolic Volume
End-Systolic Volume
Dropout
18 Systole channels + 18 Diastole Channels
8 Convolutional Layers. Kernel=(5,5)
8 Convolutional Layers. Kernel=(5,5)
4 Convolutional Layers. Kernel=(5,5)
4 Convolutional Layers. Kernel=(5,5)
2 Convolutional Layers. Kernel=(5,5)
Maxpooling and flattening 1024 x 3 x 3=9216
Dense (2 units)

Fig. 6. Volume Estimation Architecture: the input to the network is a set of images with size 96×96 pixels. The input has 36 channels (18 channels are for the diastole images and 18 channels for the systole images). In other words, we extract 1 systole image and 1 diastole image from each slice (a slice is a set of 30 images representing one heartbeat). We assume that the maximum number of slices possible is 18. After that, the input is passed through multiple sets of convolutional layers and maxpooling (8 convolutional layers & 1 maxpooling; 8 convolutional layers & 1 maxpooling; 4 convolutional layers & 1 maxpooling; 4 convolutional layers & 1 maxpooling; 2 convolutional layers & 1 maxpooling). The convolutional kernel size is (5,5). The network is regularized with dropout. The output layer contains 2 output units: one unit returns the predicted end diastole volume and the other returns the predicted end systole volume.

rectification (with leakiness = 0.1) [5,11]. The network is regularized with dropout. The output layer contains 2 output units: one unit returns the predicted end diastole volume and the other returns the predicted end systole volume. We optimize the root mean square error between the true volumes and the predicted volumes.

In addition to regularizing the network with dropout, we augment the data by performing random transformations. These transformations include randomly rotating the input between −20 and 20 degrees. We also perform random horizontal and vertical flipping. Finally, we perform random Gaussian blurring (up to $\sigma = 1.0$).

5 Results

The model was trained on a laptop with the graphics card GTX980M (4 GB dedicated RAM). Figure 7 shows the training and validation loss while training. The gap between the training loss and the validation loss remains almost the same throughout training. After 100 epochs, the validation error drops from 40 ml to around 15 ml. Each epoch takes around 4 min to complete (total training time is 6.7 h). On the other hand, predicting the volumes is done in real time.

The ejection fraction is probably the most important quantity since it measures the fraction of outbound blood pumped from the heart with each heartbeat. In general, low ejection fraction is an indication of heart problems. This quantity can be calculated as shown in Eq. 4

$$EF = 100 \times \frac{(V_D - V_S)}{V_D} \tag{4}$$

where V_D and V_S are the end diastolic and end systolic volumes, respectively.

Table 1 shows a summary of the volume errors along with the ejection fraction. The end diastolic volume error in milliliter (15.82 ml) appears to be higher

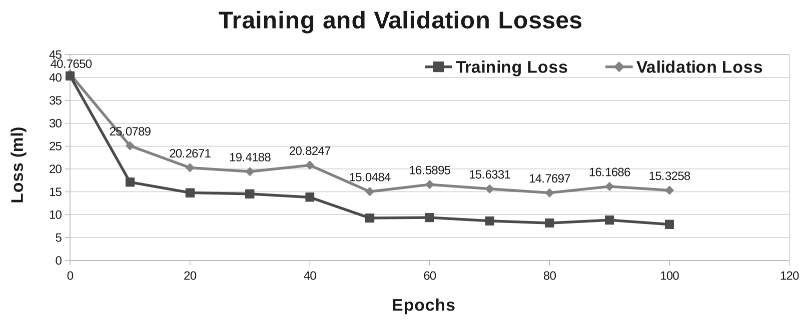

Fig. 7. Training and Validation Loss: the figure shows how the training and validation loss (error) improves over the course of training. The loss (or the error) is measured in milliliter (ml).

Table 1. Errors Summary - the table displays a summary of the volume errors in milliliter and as a percentage. In addition, the ejection fraction error is shown. Abbreviations: ESV is the End Systolic Volume error. EDV is the End Diastolic Volume error. EF is the Ejection Fraction error.

EDV(ml)	ESV(ml)	EDV(%)	ESV(%)	EF(%)
15.82	11.16	17.67	20.49	5.64

than the end systolic volume error (11.16 ml). However, when normalizing the errors by the true volume, we can see that the end diastolic percentage error (17.67 %) is better than the end systolic percentage error (20.49 %). Finally, the ejection fraction error is on average 5.64 %.

6 Conclusion

We described a model based on deep learning that can estimate the volume of the left ventricle. First, the left ventricle is localized. Then, the slice location is predicted and the slices of the study are ordered and preprocessed. After that, we identify the end systolic and end diastolic images for each slice. The end systolic and end diastolic images for each slice are stacked together and passed to a network that can estimate the end diastolic and end systolic volumes.

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, 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 19 Mar 2016
2. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2009, pp. 248–255. IEEE (2009)
3. Germano, G., Kiat, H., Kavanagh, P.B., Moriel, M., Mazzanti, M., Su, H.T., Train, K.F.V., Berman, D.S.: Automatic quantification of ejection fraction from gated myocardial perfusion spect. J. Nucl. Med. **36**(11), 2138 (1995)
4. 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)
5. Graham, B.: Spatially-sparse convolutional neural networks. arXiv preprint (2014). arXiv:1409.6070
6. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors (2012). arXiv:1207.0580

7. Kaus, M.R., von Berg, J., Weese, J., Niessen, W., Pekar, V.: Automated segmentation of the left ventricle in cardiac MRI. Med. Image Anal. **8**(3), 245–254 (2004)
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. Lin, X., Cowan, B.R., Young, A.A.: Automated detection of left ventricle in 4D MR images: experience from a large study. In: Larsen, R., Nielsen, M., Sporring, J. (eds.) MICCAI 2006. LNCS, vol. 4190, pp. 728–735. Springer, Heidelberg (2006)
10. Lynch, M., Ghita, O., Whelan, P.F.: Automatic segmentation of the left ventricle cavity and myocardium in MRI data. Comput. Biol. Med. **36**(4), 389–407 (2006)
11. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of ICML, vol. 30 (2013)
12. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814 (2010)
13. Otsu, N.: A threshold selection method from gray-level histograms. Automatica **11**(285–296), 23–27 (1975)
14. 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)
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014). arXiv:1409.1556
16. 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)
17. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions (2014). arXiv:1409.4842