3D Deep Learning for Efficient and Robust Landmark Detection in Volumetric Data

Yefeng Zheng, David Liu, Bogdan Georgescu, Hien Nguyen, and Dorin Comaniciu

Imaging and Computer Vision, Siemens Corporate Technology, Princeton, NJ, USA
yefeng.zheng@siemens.com

Abstract. Recently, deep learning has demonstrated great success in computer vision with the capability to learn powerful image features from a large training set. However, most of the published work has been confined to solving 2D problems, with a few limited exceptions that treated the 3D space as a composition of 2D orthogonal planes. The challenge of 3D deep learning is due to a much larger input vector, compared to 2D, which dramatically increases the computation time and the chance of over-fitting, especially when combined with limited training samples (hundreds to thousands), typical for medical imaging applications. To address this challenge, we propose an efficient and robust deep learning algorithm capable of full 3D detection in volumetric data. A two-step approach is exploited for efficient detection. A shallow network (with one hidden layer) is used for the initial testing of all voxels to obtain a small number of promising candidates, followed by more accurate classification with a deep network. In addition, we propose two approaches, i.e., separable filter decomposition and network sparsification, to speed up the evaluation of a network. To mitigate the over-fitting issue, thereby increasing detection robustness, we extract small 3D patches from a multi-resolution image pyramid. The deeply learned image features are further combined with Haar wavelet features to increase the detection accuracy. The proposed method has been quantitatively evaluated for carotid artery bifurcation detection on a head-neck CT dataset from 455 patients. Compared to the state-of-the-art, the mean error is reduced by more than half, from 5.97 mm to 2.64 mm, with a detection speed of less than 1 s/volume.

1 Introduction

There are many applications of automatic anatomical landmark detection in medical image analysis. For example, they can be used to align an input volume to a canonical plane on which physicians routinely perform diagnosis and quantification [1, 2]; A detected vascular landmark provides a seed point for automatic vessel centerline extraction and lumen segmentation [3]. Various landmark detection methods have been proposed in the literature. Most of the state-of-the-art algorithms [1–3] apply machine learning on a set of handcrafted image features. However, in practice, we found some landmark detection problems (e.g., carotid artery bifurcation landmarks in this work) are still too challenging to be solved with the current technology.

Recently, deep learning [4] has demonstrated great success in computer vision with the capability to learn powerful image features from a large training set. However, several

N. Navab et al. (Eds.): MICCAI 2015, Part I, LNCS 9349, pp. 565–572, 2015.
DOI: 10.1007/978-3-319-24553-9_69

challenges are present in applying deep learning to 3D landmark detection. Normally, the input to a neural network classifier is an image patch, which increases dramatically in size from 2D to 3D. For example, a patch of 32×32 pixels generates an input of 1024 dimensions to the classifier. However, a $32 \times 32 \times 32$ 3D patch contains 32,768 voxels. Such a big input feature vector creates several challenges. First, the computation time of a deep neural network is often too slow for a real clinical application. The most widely used and robust approach for object detection is the *sliding-window* based approach, in which the trained classifier is tested on each voxel in the volume. Evaluating a deep network on a large volume may take several minutes. Second, a network with a bigger input vector requires more training data. With enough training samples (e.g., over 10 million in ImageNet), deep learning has demonstrated impressive performance gain over other methods. However, the medical imaging community is often struggling with limited training samples (often in hundreds or thousands) due to the difficulty to generate and share images. Several approaches can tackle or at least mitigate the issue of limited training samples. One approach is to reduce the patch size. However, a small patch may not contain enough information for classification. Alternatively, instead of sampling a 3D patch, we can sample on three orthogonal planes [5] or even a 2D patch with a random orientation [6]. Although they can effectively reduce the input dimension, there is a concern on how much 3D information is contained in 2D planes.

In this work we tackle the above challenges in the application of deep learning for 3D anatomical structure detection (focusing on landmarks). Our approach significantly accelerates the detection speed by about 20 times, resulting in an efficient method that can detect a landmark in less than one second. We apply a two-stage classification strategy (as shown in Fig. 1). In the first stage, we train a shallow network with only one small hidden layer. This network is applied to test all voxels in the volume in a sliding-window process to generate 2000 candidates for the second stage classification. The second network is much bigger with three hidden layers (each has 2000 nodes) to obtain more discriminative power. The weights of a node in the first hidden layer are often treated as a filter (3D in this case). The response of the first hidden layer over the volume can be calculated as a convolution with the filter. Here, a neighboring patch is shifted by only one voxel; however, the response needs to be re-calculated from scratch. In this work we approximate the weights as separable filters using tensor decomposition. Therefore, a direct 3D convolution is decomposed as three one-dimensional convolutions along the x, y, and z axis, respectively. Previously, such approximation has been exploited for 2D classification problems [7, 8]. However, in 3D, the trained filters are more difficult to be approximated as separable filters. We propose a new training cost function to enforce smoothness of the filters so that they can be approximated with high accuracy. The second big network only applies on a small number of candidates that have little correlation. Separable filter approximation does not help to accelerate classification. However, many weights in a big network are close to zero. We propose to add L1-norm regularization to the cost function to drive majority of the weights (e.g., 90%) to zero, resulting in a sparse network with increased classification efficiency.

The power of deep learning is on the automatic learning of a hierarchical image representation (i.e., image features). Instead of using the trained network as a classifier, we can use the responses at each layer (including the input layer, all hidden layers, and the

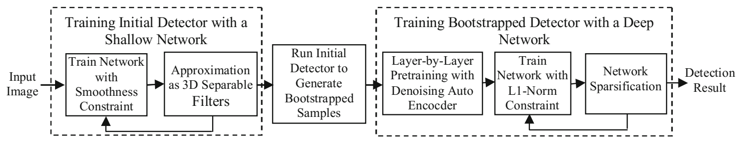
Training Initial Detector with a Shallow Network
Input Image
Train Network with Smoothness Constraint
Approximation as 3D Separable Filters
Run Initial Detector to Generate Bootstrapped Samples
Training Bootstrapped Detector with a Deep Network
Layer-by-Layer Pretraining with Denoising Auto Encoder
Train Network with L1-Norm Constraint
Network Sparsification
Detection Result

Fig. 1. Training procedure of the proposed deep network based 3D landmark detection method

output layer) as features and feed them into other state-of-the-art classifiers (e.g., boosting). After years of feature engineering, some handcrafted features have considerable discriminative power for some applications and they may be complimentary to deeply learned features. In this work we demonstrate that combining deeply learned features and Haar wavelet features, we can reduce the detection failures.

2 Efficient Detection with Neural Networks

Training Shallow Network with Separable Filters. A fully connected multilayer perceptron (MLP) neural network is a layered architecture. Suppose the input is a n_0-dimensional vector $[X_1^0, X_2^0, \ldots, X_{n_0}^0]$. The response of a node X_j^1 of the first hidden layer is

$$X_j^1 = g\left(\sum_{i=1}^{n_0} W_{i,j}^0 X_i^0 + b_j^0\right), \tag{1}$$

for $j = 1, 2, \ldots, n_1$ (n_1 is the number of nodes in the first hidden layer). Here, $g(.)$ is a nonlinear function, e.g., the sigmoid function in this work. $W_{i,j}^0$ is a weight and b_j^0 is a bias term. If we denote $\mathbf{X}^0 = [X_1^0, \ldots, X_{n_0}^0]^T$ and $\mathbf{W}_j^0 = [W_{1,j}^0, \ldots, W_{n_0,j}^0]^T$, Eq. (1) can be re-written as $X_j^1 = g\left((\mathbf{W}_j^0)^T \mathbf{X}^0 + b_j^0\right)$. Multiple layers can be stacked together using Eq. (1) as a building block. For a binary classification problem as this work, the output of the network can be a single node $\hat{X}$. Suppose there are L hidden layers, the output of the neural network is $\hat{X} = g\left((\mathbf{W}^L)^T \mathbf{X}^L + b^L\right)$. During network training, we require the output to match the class label Y (with 1 for the positive class and 0 for negative) by minimizing the squared error $E = ||Y - \hat{X}||^2$.

In object detection using a sliding window based approach, for each position hypothesis, we crop an image patch (with a pre-defined size) centered at the position hypothesis. We then serialize the patch intensities into a vector as the input to calculate response $\hat{X}$. After testing a patch, we shift the patch by one voxel (e.g., to the right) and repeat the above process again. Such a naive implementation is time consuming. Coming back to Eq. (1), we can treat the weights of a node in the first hidden layer as a filter. The first term of the response is a dot-product of the filter and the image patch intensities. Shifting the patch over the whole volume is equivalent to convolution using the filter. Therefore, alternatively, we can perform convolution using each filter $\mathbf{W}_j^0$ for $j = 1, 2, \ldots, n_1$ and cache the response maps. During object detection, we can use the cached maps to retrieve the response of the first hidden layer.

Although such an alternative approach does not save computation time, it gives us a hint for speed-up. With a bit abuse of symbols, suppose $\mathbf{W}_{x,y,z}$ is a 3D filter with size $n_x \times n_y \times n_z$. Let's further assume that $\mathbf{W}_{x,y,z}$ is separable, which means we can find three one-dimensional vectors, $\mathbf{W}_x, \mathbf{W}_y, \mathbf{W}_z$, such that

$$\mathbf{W}_{x,y,z}(i,j,k) = \mathbf{W}_x(i).\mathbf{W}_y(j).\mathbf{W}_z(k) \tag{2}$$

for any $i \in [1, n_x]$, $j \in [1, n_y]$, and $k \in [1, n_z]$. The convolution of the volume with $\mathbf{W}_{x,y,z}$ is equivalent to three sequential convolutions with $\mathbf{W}_x$, $\mathbf{W}_y$, and $\mathbf{W}_z$ along its corresponding axis. Sequential convolution with one-dimensional filters is much more efficient than direct convolution with a 3D filter, especially for a large filter. However, in reality, Eq. (2) is just an approximation of filters learned by a neural network and such a rank-1 approximation is poor in general. In this work we search for S sets of separable filters to approximate the original filter as

$$\mathbf{W}_{x,y,z} \approx \sum_{s=1}^{S} \mathbf{W}_x^s.\mathbf{W}_y^s.\mathbf{W}_z^s. \tag{3}$$

Please note, with a sufficient number of separable filters (e.g., $S \geq min\{n_x, n_y, n_z\}$), we can reconstruct the original filter perfectly.

To achieve detection efficiency, we need to cache $n_1 \times S$ filtered response maps. If the input volume is big (the size of a typical CT scan in our dataset is about 300 MB) and n_1 is relatively large (e.g., 64 or more), the cached response maps consume a lot of memory. Fortunately, the learned filters $\mathbf{W}_1^0, \ldots, \mathbf{W}_{n_1}^0$ often have strong correlation (i.e., a filter can be reconstructed by a linear combination of other filters). We do not need to maintain different filter banks for each $\mathbf{W}_i^0$. The separable filters in reconstruction can be drawn from the same bank,

$$\mathbf{W}_i^0 \approx \sum_{s=1}^{S} c_{i,s}.\mathbf{W}_x^s.\mathbf{W}_y^s.\mathbf{W}_z^s. \tag{4}$$

Here, $c_{i,s}$ is the combination coefficient, which is specific for each filter $\mathbf{W}_i^0$. However, $\mathbf{W}_x^s$, $\mathbf{W}_y^s$, and $\mathbf{W}_z^s$ are shared by all filters. Eq. (4) is a rank-S decomposition of a 4D tensor $[\mathbf{W}_1^0, \mathbf{W}_2^0, \ldots, \mathbf{W}_{n_1}^0]$, which can be solved using [9].

Using 4D tensor decomposition, we only need to convolve the volume S times (instead of $n_1.S$ times using 3D tensor decomposition) and cache S response maps. Suppose the input volume has $N_x \times N_y \times N_z$ voxels. For each voxel, we need to do $n_x n_y n_z$ multiplications using the original sliding window based approach. To calculate the response of a hidden layer with n_1 nodes, the total number of multiplications is $n_1 n_x n_y n_z N_x N_y N_z$. Using the proposed approach, to perform convolution with S set of separable filters, we need do $S(n_x + n_y + n_z)N_x N_y N_z$ multiplications. To calculate the response of n_1 hidden layer nodes, we need to combine the S responses using Eq. (4), resulting in $n_1 S N_x N_y N_z$ multiplications. The total number of multiplications is $S(n_x + n_y + n_z + n_1)N_x N_y N_z$. Suppose $S = 32$, $n_1 = 64$, the speed-up is 103 times for a $15 \times 15 \times 15$ patch.

To achieve significant speed-up and save memory footprint, we need to reduce S as much as possible. However, we found, with a small S (e.g., 32), it was more difficult to approximate 3D filters than 2D filters [7, 8]. Non-linear functions $g(.)$ are exploited in neural networks to bound the response to a certain range (e.g., [0, 1] using the sigmoid function). Many nodes are saturated (with an output close to 0 or 1) and once a node is saturated, its response is not sensitive to the change of the weights. Therefore, a weight can take an extremely large value, resulting in a non-smooth filter. Here, we propose to modify the objective function to encourage the network to generate smooth filters

$$E = ||Y - \hat{X}||^2 + \alpha \sum_{i=1}^{n_1} ||\mathbf{W}_i^0 - \overline{\mathbf{W}_i^0}||^2. \tag{5}$$

Here, $\overline{\mathbf{W}_i^0}$ is the mean value of the weights of filter $\mathbf{W}_i^0$. So, the second term measures the variance of the filter weights. Parameter α (often takes a small value, e.g., 0.001) keeps a balance between two terms in the objective function.

The training of the initial shallow network detector is as follows (as shown in the left dashed box of Fig. 1). 1) Train a network using Eq. (5). 2) Approximate the learned filters using a filter bank with S ($S = 32$ in our experiments) sets of separable filters to minimize the error of Eq. (4). The above process may be iterated a few times (e.g., three times). In the first iteration, the network weights and filter bank are initialized with random values. However, in the following iterations, they are both initialized with the optimal values from the previous iteration.

Training Sparse Deep Network. Using a shallow network, we can efficiently test all voxels in the volume and assign a detection score to each voxel. After that, we preserve 2000 candidates with the largest detection scores. The number of preserved candidates is tuned to have a high probability to include the correct detection (e.g., hypotheses within one-voxel distance to the ground truth). However, most of the preserved candidates are still false positives. In the next step, we train a deep network to further reduce the false positives. The classification problem is now much tougher and a shallow network does not work well. In this work we use a big network with three hidden layers, each with 2000 nodes. Even though we only need to classify a small number of candidates, the computation may still take some time since the network is now much bigger. Since the preserved candidates are often scattered over the whole volume, separable filter decomposition as used in the initial detection does not help to accelerate the classification. After checking the values of the learned weights of this deep network, we found most of weights were very small, close to zero. That means many connections in the network can be removed without sacrificing classification accuracy. Here, we apply L1-norm regularization to enforce sparse connection

$$E = ||Y - \hat{X}||^2 + \beta \sum_{j=1}^{L} \sum_{i=1}^{n_j} ||\mathbf{W}_i^j||. \tag{6}$$

Parameter β can be used to tune the number of zero weights. The higher β is, the more weights converge to zero. With a sufficient number of training epochs, part of weights converges exactly to zero. In practice, to speed up the training, we periodically check the magnitude of weights. The weights with a magnitude smaller than a threshold are set to zero and the network is refined again. In our experiments, we find that 90% of the weigths can be set to zero after training, without deteriorating the classification accuracy. Thus, we can speed up the classification by roughly ten times.

The proposed acceleration technologies can be applied to different neural network architectures, e.g., a multilayer perceptron (MLP) and a convolutional neural network (CNN). In this work we use the MLP. (We also tried the CNN and achieved similar,

Table 1. Quantitative evaluation of carotid artery bifurcation detection accuracy on 455 CT scans based on a four-fold cross validation. The errors are reported in millimeters.

	Mean	Std	Median	80^{th} Percentile
Haar + PBT	5.97	6.99	3.64	7.84
Neural Network (Single Resolution)	4.13	9.39	1.24	2.35
Neural Network (Multi-Resolution)	3.69	6.71	1.62	3.25
Network Features + PBT	3.54	8.40	1.25	**2.31**
Haar + Network + PBT	**2.64**	**4.98**	**1.21**	2.39

but not superior, detection accuracy.) The shallow network is trained directly with back-propagation and the deep network is trained using the denoising auto-encoder criterion [4]. The right dashed box of Fig. 1 shows the training procedure of the sparse deep network.

3 Robust Detection by Combining Multiple Features

To train a robust neural network based landmark detector on a limited training samples, we have to control the patch size. The optimal patch size was searched and we found a size of $15 \times 15 \times 15$ achieved a good trade-off between detection speed and accuracy. However, a small patch has a limited field-of-view, thereby may not capture enough information for classification. In this work we extract patches on an image pyramid with multiple resolutions. A small patch in a low-resolution volume has a much larger field-of-view at the original resolution. To be specific, we build an image pyramid with three resolutions (1-mm, 2-mm, and 4-mm resolution, respectively). The intensities of patches from multiple resolutions are concatenated into a long vector to feed the network. As demonstrated in Section 4, a multi-resolution patch can improve the landmark detection accuracy.

Deep learning automatically learns a hierarchical representation of the input data. Representation at different hierarchical levels may provide complementary information for classification. Furthermore, through years' of feature engineering, some hand-crafted image features can achieve quite reasonable performance on a certain task. Combining effective hand-crafted image features with deeply learned hierarchical features may achieve even better performance than using them separately.

In this work we propose to use probabilistic boosting-tree (PBT) [10] to combine all features. A PBT is a combination of a decision tree and AdaBoost, by replacing a weak classification node in the decision tree with a strong AdaBoost classifier. Our feature pool is composed of two types of features: Haar wavelet features $(h_1, h_2, \ldots, h_m)$ and neural network features r_i^j (where r_i^j is the response of node i at layer j). If $j = 0$, r_i^0 is an input node, representing the image intensity of a voxel in the patch. The last neural network feature is actually the response of the output node, which is the classification score by the network. This feature is the strongest feature and it is always the first selected feature by the AdaBoost algorithm.

4 Experiments

The carotid artery is the main vessel supplying oxygenated blood to the head and neck. The common carotid artery originates from the aortic arch and runs up toward the head

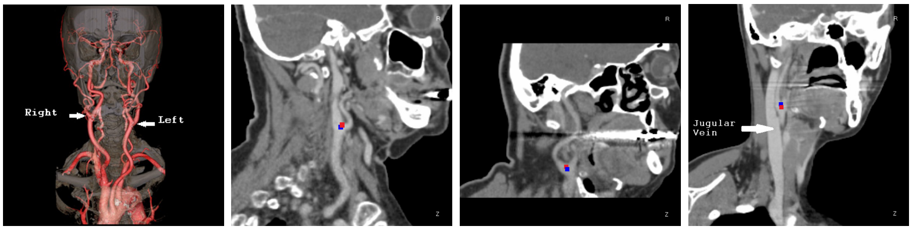

Fig. 2. Carotid artery bifurcation landmark detection in head-neck CT scans. The first column shows a 3D visualization of carotid arteries with white arrows pointing to the left and right bifurcations (image courtesy of http://blog.remakehealth.com/). The right columns show a few examples of the right carotid artery bifurcation detection results with the ground truth labeled as blue dots and detected landmarks in red.

before bifurcating to the external carotid artery (supplying blood to face) and internal carotid artery (supplying blood to brain). Examination of the carotid artery helps to assess the stroke risk of a patient. Automatic detection of this bifurcation landmark provides a seed point for centerline tracing and lumen segmentation, thereby making automatic examination possible. However, as shown in the left image of Fig. 2, the internal/external carotid arteries further bifurcate to many branches and there are other vessels (e.g., vertebral arteries and jugular veins) present nearby, which may cause confusion to an automatic detection algorithm.

We collected a head-neck CT dataset from 455 patients. Each image slice has 512×512 pixels and a volume contains a variable number of slices (from 46 to 1181 slices). The volume resolution varies too, with a typical voxel size of $0.46 \times 0.46 \times 0.50\,mm^3$. A four-fold cross validation is performed to evaluate the detection accuracy and determine the hyper parameters, e.g., the network size, smoothness constraint α in Eq. (5), sparsity constraint β in Eq. (6). There are two carotid arteries (left vs. right) as shown in Fig. 2. Due to the space limit, we only report the bifurcation detection accuracy of the right carotid artery (as shown in Table 1) with different approaches. The detection accuracy of the left carotid artery bifurcation is similar.

For each approach reported in Table 1, we follow a two-step process by applying the first detector to reduce the number of candidates to 2000, followed by a bootstrapped detection to further reduce the number of candidates to 250. The final detection is picked as the candidate with the largest vote from other candidates. Previously, Liu et al. [3] used Haar wavelet features + boosting to detect vascular landmarks and achieved promising results. Applying this approach on our dataset, we achieve a mean error of 5.97 mm and the large mean error is caused by too many detection outliers. The neural network based approach can significantly improve the detection accuracy with a mean error of 4.13 mm using a $15 \times 15 \times 15$ patch extracted from a single resolution (1 mm). Using patches extracted from an image pyramid with three resolutions, we can further reduce the mean detection error to 3.69 mm. If we combine features from all layers of the network using the PBT, we achieve slightly better mean accuracy of 3.54 mm. Combining the deeply learned features and Haar wavelet features, we achieve the best detection accuracy with a mean error of 2.64 mm. We suspect that the improvement comes from the complementary information of the Haar wavelet features and neural network features. Fig. 2 shows the detection results on a few typical datasets.

The proposed method is computationally efficient. Using the speed-up technologies presented in Section 2, it takes 0.92 s to detect a landmark on a computer with a six-core 2.6 GHz CPU (without using GPU). For comparison, the computation time increases to 18.0 s if we turn off the proposed acceleration technologies. The whole training procedure takes about 6 hours and the sparse deep network consumes majority of the training time.

5 Conclusions

In this work we proposed 3D deep learning for efficient and robust landmark detection in volumetric data. We proposed two technologies to speed up the detection using neural networks, namely, separable filter decomposition and network sparsification. To improve the detection robustness, we exploit deeply learned image features trained on a multi-resolution image pyramid. Furthermore, we use the boosting technology to incorporate deeply learned hierarchical features and Haar wavelet features to further improve the detection accuracy. The proposed method is generic and can be re-trained to detect other 3D landmarks.

References

1. Zhan, Y., Dewan, M., Harder, M., Krishnan, A., Zhou, X.S.: Robust automatic knee MR slice positioning through redundant and hierarchical anatomy detection. IEEE Trans. Medical Imaging 30(12), 2087–2100 (2010)
2. Schwing, A.G., Zheng, Y.: Reliable extraction of the mid-sagittal plane in 3D brain MRI via hierarchical landmark detection. In: Proc. Int'l Sym. Biomedical Imaging, pp. 213–216 (2014)
3. Liu, D., Zhou, S., Bernhardt, D., Comaniciu, D.: Vascular landmark detection in 3D CT data. In: Proc. of SPIE Medical Imaging, pp. 1–7 (2011)
4. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. The Journal of Machine Learning Research 11, 3371–3408 (2010)
5. Prasoon, A., Petersen, K., Igel, C., Lauze, F., Dam, E., Nielsen, M.: Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network. In: Mori, K., Sakuma, I., Sato, Y., Barillot, C., Navab, N. (eds.) MICCAI 2013, Part II. LNCS, vol. 8150, pp. 246–253. Springer, Heidelberg (2013)
6. Roth, H.R., et al.: A new 2.5D representation for lymph node detection using random sets of deep convolutional neural network observations. In: Golland, P., Hata, N., Barillot, C., Hornegger, J., Howe, R. (eds.) MICCAI 2014, Part I. LNCS, vol. 8673, pp. 520–527. Springer, Heidelberg (2014)
7. Rigamonti, R., Sironi, A., Lepetit, V., Fua, P.: Learning separable filters. In: Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 2754–2761 (2013)
8. Denton, E., Zaremba, W., Bruna, J., LeCun, Y., Fergus, R.: Exploiting linear structure within convolutional networks for efficient evaluation. In: Advances in Neural Information Processing Systems, pp. 1–11 (2014)
9. Acar, E., Dunlavy, D.M., Kolda, T.G.: A scalable optimization approach for fitting canonical tensor decompositions. Journal of Chemometrics 25(2), 67–86 (2011)
10. Tu, Z.: Probabilistic boosting-tree: Learning discriminative methods for classification, recognition, and clustering. In: Proc. Int'l Conf. Computer Vision, pp. 1589–1596 (2005)