Expert Systems With Applications 112 (2018) 229–242

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Retinal vessel segmentation based on Fully Convolutional Neural Networks

Américo Oliveira*, Sérgio Pereira, Carlos A. Silva*

Department of Industrial Electronics, Campus de Azurém, Alameda da Universidade, CMEMS-UMinho Research Unit, University of Minho, Guimarães 4804-533, Portugal

ARTICLE INFO

Article history:
Received 10 January 2018
Revised 16 May 2018
Accepted 14 June 2018
Available online 18 June 2018

Keywords:
Fully Convolutional Neural Network
Stationary Wavelet Transform
Retinal fundus image
Vessel segmentation
Deep learning

ABSTRACT

The retinal vascular condition is a reliable biomarker of several ophthalmologic and cardiovascular diseases, so automatic vessel segmentation may be crucial to diagnose and monitor them. In this paper, we propose a novel method that combines the multiscale analysis provided by the Stationary Wavelet Transform with a multiscale Fully Convolutional Neural Network to cope with the varying width and direction of the vessel structure in the retina. Our proposal uses rotation operations as the basis of a joint strategy for both data augmentation and prediction, which allows us to explore the information learned during training to refine the segmentation. The method was evaluated on three publicly available databases, achieving an average accuracy of 0.9576, 0.9694, and 0.9653, and average area under the ROC curve of 0.9821, 0.9905, and 0.9855 on the DRIVE, STARE, and CHASE_DB1 databases, respectively. It also appears to be robust to the training set and to the inter-rater variability, which shows its potential for real-world applications.

1. Introduction

The retinal vascular tree is the only structure in the human circulatory system that can be directly and non-invasively observed in vivo, besides being easily photographed (Patton et al., 2006). Apart from other applications, as biometric identification, multimodal image registration, or computer-assisted laser surgery (Fraz et al., 2012a), retinal fundus imaging has become essential for diagnosing, treating, and monitoring several disorders, such as arteriosclerosis, hypertension, age-related macular degeneration, diabetic retinopathy, and glaucoma (Abràmoff, Garvin, & Sonka, 2010; Fraz et al., 2012a).

The morphological characteristics of the retinal vasculature, such as angles, branching patterns, length, width, and tortuosity, can play a crucial role in assisting cardiologists and ophthalmologists (Abràmoff et al., 2010; Fraz et al., 2012a). Their quantitative evaluation, however, demands proper segmentation of the vascular tree.

Manually segmenting blood vessels in retinal images is both error-prone and time-consuming, even for experienced physicians. For this reason, automatic and accurate segmentation is vital. This process configures a complex task, not only due to abrupt variations in the attributes (size, shape, intensity levels) and arrangement (branching, crossing) of the vessels but also to the low quality of retinal images (Abràmoff et al., 2010; Fraz et al., 2012a). If lesions occur, the task becomes even more challenging.

1.1. Related work

Several works have been proposed for retinal vessel segmentation in recent decades. At large, all of them may be seen as supervised or unsupervised methods.

Unsupervised methods make use of prior knowledge on the structure of the vessels and usually rely on rule-based schemes. These algorithms often apply matched filtering, morphological processing, vessel tracking, multiscale, or model-based approaches. Matched filtering techniques, as adopted by Hoover, Kouznetsova, and Goldbaum (2000), convolve retinal images with a 2D filter to produce a Gaussian intensity profile of blood vessels. Azzopardi, Strisciuglio, Vento, and Petkov (2015) have employed a combination of shifted filter responses (COSFIRE) to detect retinal vessels or any vessel-like pattern. In Zhang et al. (2016), a 2D image is lifted to a 3D orientation score, and vessels are then enhanced by multiscale derivatives. Present-day approaches seem to reach promising results when dealing with complicated vessel geometries. Still, they may be penalized by the unbalanced filter response to pathological vessels. Strategies based on mathematical morphology combined with matched filtering for centerline detection, as

* Corresponding authors.
E-mail addresses: a68396@alunos.uminho.pt (A. Oliveira), id5692@alunos.uminho.pt (S. Pereira), csilva@dei.uminho.pt (C.A. Silva).

https://doi.org/10.1016/j.eswa.2018.06.034
0957-4174/

in Mendonca and Campilho (2006), or adaptive thresholding, as in Roychowdhury, Koozekanani, and Parhi (2015b) and Neto, Ramalho, Neto, Veras, and Medeiros (2017), can also be found. Vessel tracking techniques, such as those of Liu and Sun (1993) and Yin, Adel, and Bourennane (2012), establish an initial set of points (selected either manually or automatically) and iteratively follow the vessel centerlines to extract the vascular tree. However, besides being poor at detecting not seeded vessel segments, they can miss initially flagged ones too. Multiscale techniques, like those of Wang, Ji, Lin, and Trucco (2013) and Nguyen, Bhuiyan, Park, and Ramamohanarao (2013), try to improve the analysis of blood vessels of varying width by adjusting the scale to the diameter of the vessel, but selecting the parameters of a multiscale filter is not a trivial task. Finally, model-based techniques either apply explicit vessel profile or deformable models to extract the vasculature. Lam, Gao, and Liew (2010) came up with a multi-concavity method to treat, simultaneously, both healthy and pathological images. Zhao, Rada, Chen, Harding, and Zheng (2015) showed an infinite active contour model that combines intensity and local phase information. In general, unsupervised methods may generalize well across images, since they are not learned from a sample of the population as the supervised ones. Still, it may be difficult to encompass both normal and abnormal vasculatures due to their diverse appearing.

Supervised methods use ground truth data to train a classifier to label each pixel as either vessel or background. During the training stage, an optimization algorithm analyses the desired output for each training example and infers a function, which the classifier uses for dealing with unseen examples, in the testing stage. Niemeijer, Staal, van Ginneken, Loog, and Abramoff (2004) started by introducing a vector of features, based on Gaussian functions and their derivatives, for each pixel, and then applied a k-nearest neighbor classifier. Later, Staal, Abràmoff, Niemeijer, Viergever, and van Ginneken (2004) extended this method by introducing ridge profiles in vessel detection. Soares, Leandro, Cesar, Jelinek, and Cree (2006) used a gaussian mixture model classifier and computed a 6-feature set extracted by a Gabor wavelet transform. Marín, Aquino, Gegúndez-Arias, and Bravo (2011) applied neural networks to classify a 7-feature set composed of gray-level and moment invariant-based features. Fraz et al. (2012b) combined the analysis of the gradient vector, line strength measures, and Gabor filter responses to compute a 9-feature set, which was passed to an AdaBoost classifier. Roychowdhury, Koozekanani, and Parhi (2015a), in turn, used a gaussian mixture model to classify a 8-feature set based on each pixel's neighborhood and first and second-order gradient images. Strisciuglio, Azzopardi, Vento, and Petkov (2016) came up with a bank of COSFIRE filters and trained a support vector machine classifier to determine the most effective subset of filters. Orlando, Prokofyeva, and Blaschko (2017) presented a fully connected conditional random field model, whose parameters were learned with a structured output support vector machine. Recently, Zhang et al. (2017) employed a random forest classifier to deal with a 29-feature set built by combining vessel filtering and wavelet transform features. Contrasting with the previous unsupervised approaches, supervised methods can bypass the need for designing different models whenever new segmentation problems arise. Still, results are often strongly affected by the feature engineering stage.

Deep learning-based methods have recently drawn attention since they can automatically learn an increasingly complex hierarchy of features directly from the input data (Bengio, Courville, & Vincent, 2013). This hints a paradigm shift according to which people are now optimizing architectures, instead of designing handcrafted features that may be problem dependent and require expert knowledge. Even if we can trace deep neural networks back to the last century (LeCun et al., 1990), they have recently been applied to several areas, including retinal vessel segmentation. Li et al. (2016), for example, turned the segmentation task into a vessel mapping problem, where the mapping function was learned by a 5-layer deep neural network. In general, this recent trend is due to the fortunate combination of two factors: (1) the amount of data available for training is now massively larger than it was decades ago; and (2) it coincided with the development of more powerful graphical processing units (LeCun, Bengio, & Hinton, 2015).

Convolutional neural networks (CNNs), in particular, have already been used to win quite a few object recognition (Dieleman, Willett, & Dambre, 2015; Krizhevsky, Sutskever, & Hinton, 2012) and biological image segmentation (Ronneberger, Fischer, & Brox, 2015) challenges. In retinal image analysis, recent proposals also employ them. Melinščak, Prentašić, and Lončarić (2015) addressed vessel segmentation using a 10-layer CNN. In Liskowski and Krawiec (2016), a structured prediction scheme was used to highlight context information, while testing a comprehensive set of architectures among which a 7-layer no-pooling CNN was the most successful. Recently, Fu, Xu, Lin, Wong, and Liu (2016) combined a typical 7-layer CNN with a conditional random field, reformulated as a recurrent neural network, to model long-range pixel interactions.

While previous works on deep learning used only raw data, which was processed by cascaded convolutional layers to learn features, it has been recently suggested that deep neural networks can generate more useful features when provided with extra information. This has been, for instance, explored using wavelets for SAR image segmentation (Duan, Liu, Jiao, Zhao, & Zhang, 2017) or in super-resolution (Guo, Mousavi, Vu, & Monga, 2017). We further consider the use of wavelets, combined with a multiscale CNN, for segmenting retinal vessels.

1.2. Motivation and contributions

In a preliminary version of this work (Oliveira, Pereira, & Silva, 2017), we presented a CNN that achieved promising results in retinal vessel segmentation. This paper extends our earlier work. The Stationary Wavelet Transform (SWT) was incorporated to exploit the multiscale nature of the vascular system. An alternative data augmentation strategy was used, and a new multiple prediction scheme, motivated by it, was introduced. Finally, additional ablation studies and implementation details better cover the underlying principles of our method.

The main idea of this approach is to combine information with different levels of abstraction. In a typical CNN, several pooling operators successively reduce the dimensions of the feature maps that are being produced by each convolutional layer. This builds high-level information maps, with low resolution. Although these maps contain more complex and compact features, some details may be lost. Thus, the low-level information available in the shallow layers may help. In other words, we provide detail information to the network, which must learn how to use it, to mitigate the losses of information that may occur during the traditional path.

The main contributions of this paper to the retinal vessel segmentation problem are as follows. We propose rotation operations as the basis of a joint strategy for data augmentation and prediction. Although data augmentation is a well known technique, here we explore the information learned during training, about the arrangement and orientation of the vessels, to refine the segmentation. We also investigate the decomposition of the image through SWT as a way to add new input channels into a Fully Convolutional Neural Network (FCN); this contribution is independent of the architecture, but here we show how to combine a multiscale architecture with the SWT for better handling retinal vessels at different scales.

Our method was evaluated on three publicly available databases: DRIVE (Staal et al., 2004), STARE (Hoover et al., 2000), and CHASE_DB1 (Owen et al., 2009). A worth mentioning method-

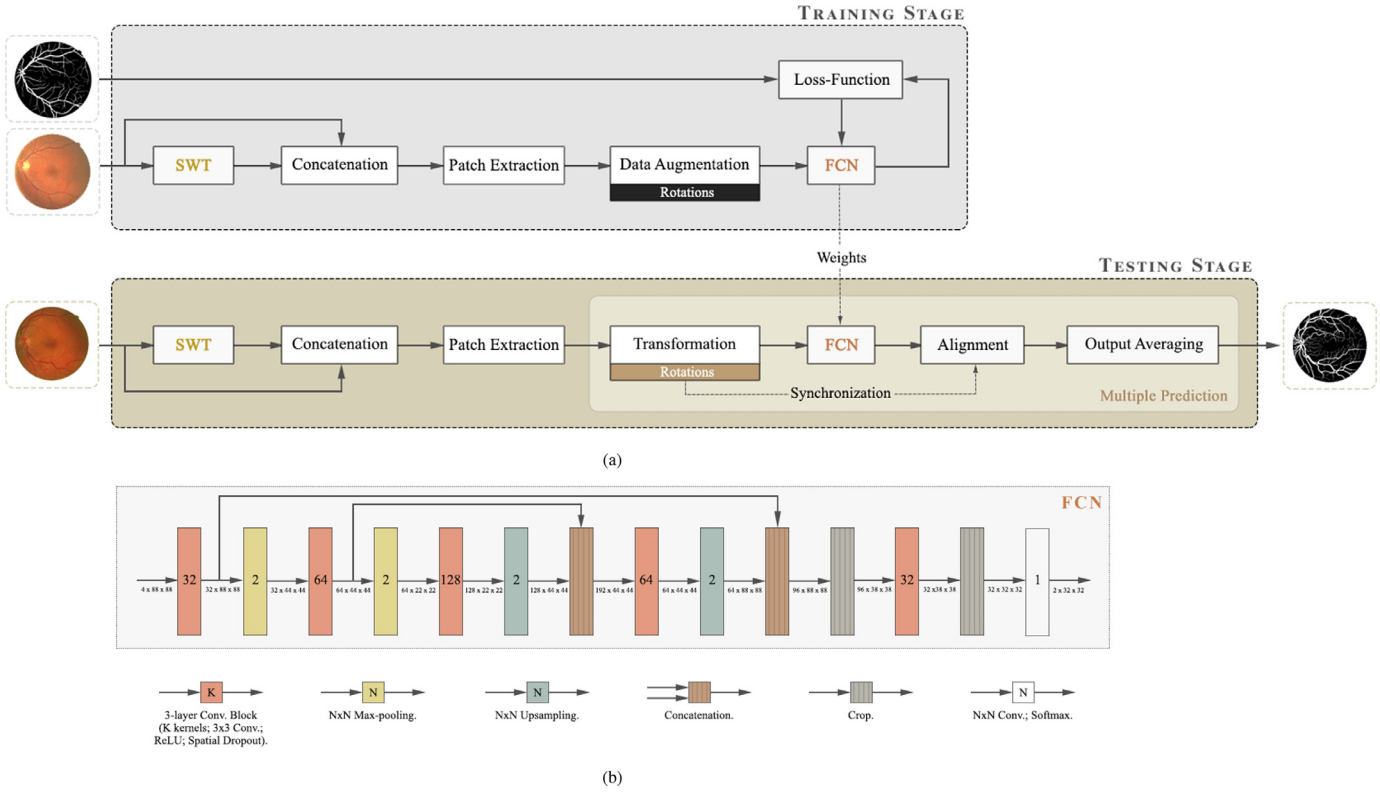


Fig. 1. Overview of the proposed method: (a) Block diagram; (b) FCN architecture.

ological contribution is the comparison between the annotations provided by different physicians and the study of the effects that this inter-rater variability can have on the behavior of the network.

The remaining sections are organized as follows. The proposed method is described in Section 2. The experimental setup is presented in Section 3. Results and discussion can be found in Section 4. Before closing, we come up with the main conclusions in Section 5.

2. Method

Fig. 1a presents an overview of the proposed approach. There are four main stages: input building through the SWT, patch extraction, classification via FCN, and multiple prediction.

2.1. Stationary Wavelet Transform

The SWT (Holschneider, Kronland-Martinet, Morlet, & Tchamitchian, 1990) was initially designed to overcome two drawbacks of the discrete wavelet transform (DWT): (1) DWT is not translation-invariant; and (2) it can only be used on images of dyadic size (Holschneider et al., 1990). Here we propose the SWT as a method to enrich the input of the FCN. We employed the SWT over the DWT since it does not downsample the coefficients, preserving the initial number of pixels. Thus, it allows us to add new extra channels to the input.

Fig. 2 gives some insights on how SWT operates. For simplicity's sake, we show a 2-level decomposition with the 1D transform. $\overline{g_j}$ and $\overline{h_j}$ are, respectively, the high-pass and low-pass filters of each level j. The former returns the detail coefficient d_j; the latter transfers the approximation coefficient a_j to the level $j+1$ for further decomposition (Fig. 2a). This filter pair splits the input signal into two spectral bands (Fig. 2b). At each level $j+1$, the impulse response of each low-pass filter is obtained by upsampling the corresponding response of the previous level j (Holschneider et al., 1990):

$$\overline{h_{j+1}}[k] = \overline{h_j}[k] \uparrow = \begin{cases} \overline{h_j}[\frac{k}{2}] & k \text{ even}, \\ 0 & k \text{ odd}, \end{cases} \tag{1}$$

with the equivalent equation being applied to the high-pass case. Each signal of level $j+1$ is obtained by convolving the corresponding signal of the previous level j with $\overline{g_j}$ or $\overline{h_j}$. These signals keep the original dimensions of the input and are translation invariant. Moreover, they contain multi-resolution information, which is very useful for segmenting images. The reconstruction filters $\widetilde{g_j}$ and $\widetilde{h_j}$ are time-reversed versions of their homologous decomposition filters. When dealing with images, the 2D transform is required. This implies vertical, horizontal, and diagonal versions of the process, resulting in three detail images (dV_j, dH_j, and dD_j) at each level j (Fig. 3).

Throughout this work, we used the Haar wavelet (Haar, 1910), which can be described as:

$$\psi(t) = \begin{cases} 1 & 0 \le t < \frac{1}{2}, \\ -1 & \frac{1}{2} \le t < 1, \\ 0 & otherwise. \end{cases} \tag{2}$$

2.2. Patch extraction

In our approach, the patches fed to the network were obtained differently according to the stage of the algorithm. During training, we extracted 2750, 3250, and 3750 patches from each image of the DRIVE, STARE, and CHASE_DB1 databases, respectively. We noticed that the network benefited from using more patches in larger images, and these values were experimentally found. In addition, we did not apply any restrictions at this stage, so overlapping patches were allowed. During testing, however, we ensured there was no overlap between the output patches, i.e., each pixel was segmented

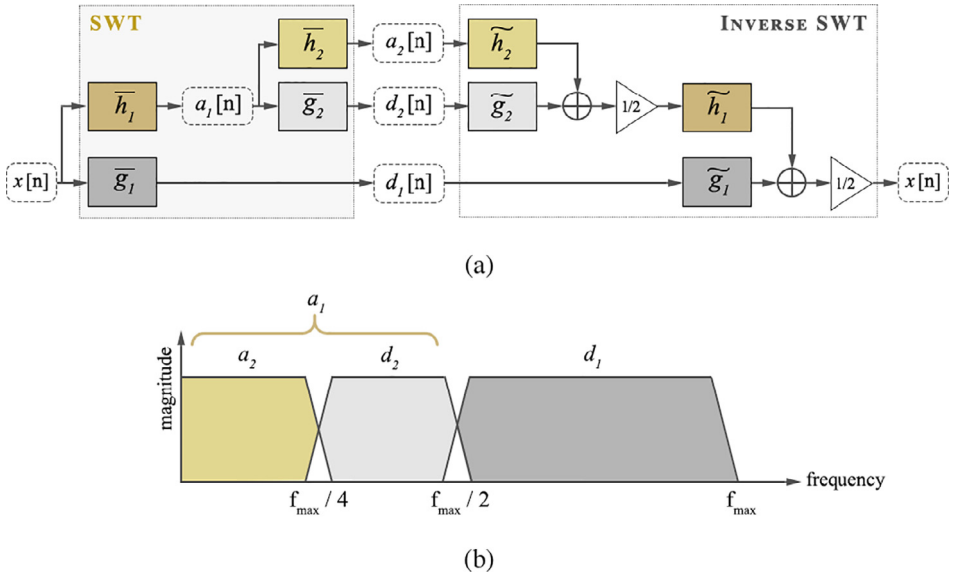


Fig. 2. SWT operation: (a) Filter bank; (b) Spectral analysis.

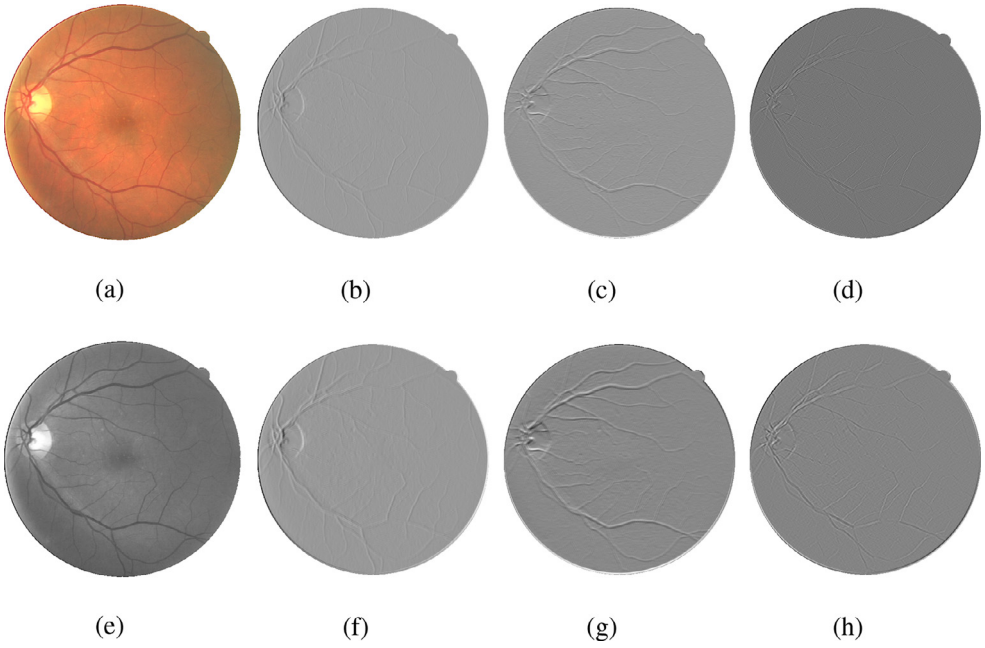


Fig. 3. Application of the SWT to a retinal image: (a) Color image; (b) dV_1; (c) dH_1; (d) dD_1; (e) Green channel; (f) dV_2; (g) dH_2; (h) dD_2.

only once. This was done by zero padding the original image in such a way that its dimension becomes an integer multiple of the output patch size. Furthermore, we applied an overlap-tile strategy (Ronneberger et al., 2015), where each output patch only contains the pixels for which the full context is available in the input image. This was the main reason for the cropping operations, which led to the mismatch of dimensionality between the 88×88 input patch and the 32×32 output patch (Fig. 1b).

2.3. Fully Convolutional Neural Network

CNNs are based on convolutional layers, which receive a stack of input planes and return a group of feature maps. Each feature map results from convolving a single kernel over the inputs, with each layer having a variable number of kernels.

If we denote the kth kernel by W_k, the computation of the associated feature map F_k can be formally described as:

$$F_k = b_k + \sum_c (W_k)_c * X_c \,, \tag{3}$$

where $(W_k)_c$ and X_c are, respectively, the cth channel of both the kernel and the input, $*$ is the convolution operation, and b_k represents the bias term.

From the neural network perspective, each unit of a feature map is the output of a neuron that captures particular features from a restricted region of the previous layer. That region – called the *receptive field* of the neuron – is defined by the kernel size. If a unit has a high value, then the kernel has found a match and the feature encoded by it (like an edge or a curve) is, at least partially, present in the input. Otherwise, there is a less relevant correspon-

dence. As new layers are stacked, and deeper nets appear, more complex features (as motifs, parts, or objects) are recognized.

Among the reasons that make CNNs well-suited to processing visual information, there are two key ideas: local connections and shared weights. Local connectivity means that each hidden unit only looks for its own receptive field, which significantly reduces the number of weights. Weight sharing happens since the same set of weights is convolved over the whole image, which also increases computational efficiency and provides CNNs with translation invariance (LeCun et al., 2015).

In the following lines, we address our decisions regarding some fundamental aspects when dealing with CNNs.

2.3.1. Initialization

We adopted the Xavier initialization (Glorot & Bengio, 2010), which allowed us to maintain the gradients in controlled levels and thus prevent gradient vanishing during back-propagation.

2.3.2. Activation function

Herein, our choice fell on the rectified linear units (ReLU) (Nair & Hinton, 2010), $f(x) = \max(0, x)$, which were found to speed up training in comparison with other nonlinearities, such as the sigmoid or the hyperbolic tangent functions (Krizhevsky et al., 2012).

2.3.3. Pooling

We employed max-pooling, which discards possibly redundant features, making the representation invariant to small details (LeCun et al., 2015).

2.3.4. Upsampling

To return the feature maps to their initial dimensions, we employed nearest neighbor interpolation.

2.3.5. Regularization

Tompson, Goroshin, Jain, LeCun, and Bregler (2015) suggested that when facing natural images with high spatial correlation, standard dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) may be less efficient since neighboring units in the same feature map also become heavily correlated. Thus, we adopted spatial dropout (Tompson et al., 2015), which removes entire feature maps instead of just some nodes. In this way, adjacent units in each feature map are either all neutralized or all active. The same probability p was used in all convolutional blocks, except in the last one where we applied p_{last}.

2.3.6. Architecture

In classic image recognition CNNs (Krizhevsky et al., 2012; Simonyan & Zisserman, 2014), it was common to identify a feature extraction stage based on the convolutional and pooling layers, and a classification stage carried out by a variable number of fully connected (FC) layers. The problem with this type of networks, which we will refer to as FC-CNNs, was that they were forced to take fixed-sized inputs since they had at least one FC layer. In FCNs (Long, Shelhamer, & Darrell, 2015), all the FC layers are replaced by convolutions (often with 1×1 kernels). In this way, a FCN can take an image, of any size, as input, and directly output a set of probability maps with the same dimensions. This, besides bringing advantages regarding versatility, makes FCNs more efficient than FC-CNNs, since it is possible to segment a set of pixels at once, instead of treating each pixel singly.

With the appearance of FCNs, multi-scale architectures, also known as encoder-decoder networks, became more popular. The work by Long et al. (2015) was perhaps the first to investigate the idea of merging feature maps with different levels of abstraction, but several other multi-scale methods for semantic segmentation followed in a short time (Badrinarayanan, Kendall, & Cipolla, 2015; Dai, He, & Sun, 2015; Eigen & Fergus, 2015; Hariharan, Arbeláez, Girshick, & Malik, 2015; Noh, Hong, & Han, 2015). In medical image segmentation, this trend led to the U-net (Ronneberger et al., 2015).

Table 1
Hyperparameters of the proposed FCN.

Stage	Hyperparameter	Value
Initialization	Bias	0.1
	Weights	Xavier
Training	Epochs	20
	Mini-batch size	4
	Spatial dropout p	0.2
	Spatial dropout p_{last}	0.15
	Learning rate decay λ	1×10^{-6}

Our architecture, which is shown in Fig. 1b, is inspired by these previous works. The network is composed of both an encoder and a decoder. In the former, pooling is applied to summarize neighboring features and create high-level representations. In the latter, feature maps are reconstructed, combining those coming from the encoder, through skip connections, with those belonging to low-level scales. While pooling operations are highly suitable for recognizing objects in images and crucial for increasing the receptive field of deeper units, they can significantly harm localization performance. In fact, we seem to face an intriguing problem: high-level information reveals *what*, but low-level information reveals *where* (Long et al., 2015). In other words, building a one-way path capable of achieving satisfactory results may not be feasible. This is the main motivation for the decoder structure since skip connections allow information to propagate directly from shallow, high-resolution, low-level information layers to deep, low-resolution, high-level information ones. Another important aspect is that after each max-pooling in the encoder path we double the number of feature maps, whereas in the decoder path we halve it after each upsampling. This is mainly due to computational reasons, since larger feature maps lead to higher computational load.

In our network, the first channel of the 4-channel input patch resulted from taking the green channel of the retinal image and normalizing it to have zero mean and unit variance. The remaining channels were obtained by applying the SWT over the first and then subjecting them to the same normalization procedure. In Section 4.1, we evaluate our proposal using just the green channel. In Section 4.2, we show the benefits of including the SWT channels. We used small 3×3 kernels in all convolutional layers (except in the 1×1 linear convolution). Stacking smaller kernels can ensure the same effective receptive field of bigger ones while reducing the number of weights (Simonyan & Zisserman, 2014). These and other hyper-parameters are summarized in Table 1. Therein, a mini-batch size of 4 means that four input patches were propagated through the network in each iteration.

2.3.7. Training

During the optimization process, we applied stochastic gradient descent with Nesterov momentum (Nesterov, 1983) to minimize a categorical cross-entropy loss-function:

$$J(w) = -\frac{1}{M}\sum_{j=1}^{M}\sum_{k=1}^{K} y_{j,k}\, ln(\hat{y}_{j,k}), \tag{4}$$

where M and K denote, respectively, the number of pixels and classes, $\hat{y}$ refers to the probabilistic predictions (after the softmax), and y is the target.

Furthermore, both the momentum ν and the learning rate η were changed in specific epochs, according to Table 2. Besides that, we still further decayed the latter one in-between these changes,

Table 2
Schedule of the optimization parameters.

Parameter	Epoch	Value
Learning rate η	1	0.05
	10	0.02
	14	0.002
	18	0.0002
Momentum ν	1	0.2
	10	0.9
	14	0.99

according to:

$$\eta_n = \frac{\eta_{n-1}}{1 + \lambda \times n}, \tag{5}$$

where η_n and η_{n-1} are, respectively, the learning rate in the present and previous updates, and λ is the learning rate decay. Herein, an epoch means a complete pass over all the training samples, while an update refers to the update of the weights after each iteration.

2.4. Data augmentation and multiple prediction

In our data augmentation procedure, each original patch was rotated by 90°, 180°, and 270°. Then, these patches were randomly placed in the training vector. The number of patches was increased by four (since the original one was included too). To avoid interpolations, only multiples of 90° were used. Also, each operation was applied to both the original patch and the respective annotated one to keep the correspondence.

We restricted augmentation to rotations, as they proved to be particularly useful. Even though rotations are common in data augmentation, here we intend to clarify how to use them in the specific context of FCNs. In fact, many valuable proposals that do data augmentation with rotations, in FC-CNNs, can be found in the literature. A good example is to embed the rotational augmentation in the network itself, as in Dieleman, De Fauw, and Kavukcuoglu (2016) and Worrall, Garbin, Turmukhambetov, and Brostow (2016). However, these strategies were developed for FC-CNNs and cannot be applied in FCNs for two main reasons. First, when dealing with FCNs, there is a correspondence between each pixel in the input patch and in the subsequent feature maps; so, if we rotate the feature maps inside the network, as embedded layers, and further add them, as these works suggest for FC-CNNs, we would lose such correspondence. Second, these strategies require the rotated versions to be arranged consecutively, which we show in Section 4.1.1 to be detrimental to the performance.

While for both FC-CNNs and FCNs, a mini-batch contains patches that group together neighbor pixels, a particularity in the training of FCNs is that the computation of the loss takes into account the error of each neighbor pixel in the patch. This seems to indicate that the network will encode context information provided by the combination of the pixels. Considering the positive effects of the rotational augmentation in the generalization capacity of the network, we may conclude that the context provided by the rotations brought new information that was encoded in the network. Although this encoded information was used efficiently in FC-CNNs by works as Dieleman et al. (2016), Worrall et al. (2016), the same was not achieved in FCNs, and we believe this is due to the internal alignment required for a proper use of FCNs. However, this limitation can be surpassed if we apply the rotations differently during training and prediction. In training, we must present the rotated patches randomly in order to effectively train the network; however, in prediction, they should be presented consecutively to ensure that we are able to fuse the information of all possible contexts. With this in mind, we employed the following procedure (recap Fig. 1a). We started by generating three artificial rotated copies of each patch to be segmented, as described above, making up a set of four examples – Transformation. Each example was then segmented and we obtained the respective probability map of each class. Finally, the probability maps of the rotated patches underwent a rotation by the same initial angle, but in the opposite direction – Alignment. This allowed the four maps of each class to be pixel-wise merged by averaging, building the final unique segmentation – Output Averaging.

3. Experimental setup

3.1. Materials

The proposed method was evaluated using three publicly available databases.

The DRIVE database (Staal et al., 2004) consists of 40 images, 7 of which showing pathological signals. Each image has a resolution of 565×584 and 8 bits per channel. The STARE database (Hoover et al., 2000) contains 20 images, 10 of which belonging to sick individuals. These images have a resolution of 700×605 and 24 bits per channel. The CHASE_DB1 database (Owen et al., 2009) has 28 images, collected from both the eyes of 14 children. Each image has a resolution of 999×960.

In the DRIVE case, the global set is divided into the training and testing sets, each of them with 20 images. Thus, the model was evaluated on the testing set. In the remaining two cases, however, there is no clear division. Given this, the model was trained using a stratified k-fold cross-validation, where the original set was partitioned into k equal-sized folds. The validation process was repeated k times, with each fold being retained as testing set, and the remaining $k-1$ folds being used for training. The results were then averaged to produce a single estimation. In the STARE case, we settled $k = 5$, having 5 folds of 4 images. Besides, we stratified the folds by ensuring that half of the images in each of them belong to pathological individuals. In the CHASE_DB1 case, we used $k = 4$, obtaining 4 folds of 7 images. This time, each fold included 3 images of one eye and 4 images of the other.

When using the k-fold cross-validation, the same architecture was used in all folds. The network was trained from scratch in each fold.

For each database, there are two manual segmentations available made by two independent human observers. For the DRIVE and CHASE_DB1 databases, we used the annotations of the first human observer as ground truth. For the STARE database, the gold standard were the annotations from Hoover. The human observer performance was measured using the manual segmentations of the second human observer. The binary masks for DRIVE images are publicly available. For the remaining databases, we have manually created them. These and other supporting materials for this work are publicly available online.[1]

3.2. Evaluation metrics

To enable comparison with other state-of-the-art works, we used four metrics commonly found in the literature: sensitivity (Sn), specificity (Sp), accuracy (Acc), and area under the ROC curve (AUC). The perfect classifier would hit 1 in all of them.

[1] https://github.com/americofmoliveira/VesselSegmentation_ESWA.

Table 3
Segmentation results of each test on the components of the Base System, on DRIVE. Bold values show the best score among all methods; underlined values represent metrics where the null hypothesis can be rejected (p-value < 0.05 when facing the Base System).

Method	Dropout	Patches per image			Prediction	Channels	*Acc*	AUC
		Original	Rotated	Elastic				
No Augmentation	Spatial	3000	–	–	Multiple	1	0.9557	0.9780
Oversampling	Spatial	12000	–	–	Multiple	1	0.9557	0.9785
Elastic Samples	Spatial	3000	–	9000	Multiple	1	0.9558	0.9785
Consecutive Rotations	Spatial	3000	9000	–	Multiple	1	0.9560	0.9795
Simple Prediction	Spatial	3000	9000	–	Simple	1	0.9567	0.9805
Standard Dropout	Standard	3000	9000	–	Multiple	1	0.9564	0.9805
Base System	Spatial	3000	9000	–	Multiple	1	**0.9570**	**0.9815**

3.3. Implementation details

The proposed method was implemented using Keras[2] with TensorFlow[3] backend and cuDNN 5.1. All tests were conducted on a desktop equipped with a NVIDIA GeForce GTX 1070 GPU, an Intel Core i7-6850K CPU @ 3.60 GHz processor, 128 GB of RAM, and running Linux Mint 18 OS.

4. Results and discussion

We begin this section by validating the key components of the *Base System*. Then, we obtain the best model by adding the channels obtained via SWT to it. Having finished these steps, we compare our best model with other state-of-the-art works. The clinical applicability is assessed using a cross-training strategy. Finally, we analyze the behavior of the model when facing the inter-rater variability.

Due to the size of the population and the uncertainty regarding the normality of the data, all tests of statistical significance were computed using the two-sided paired Wilcoxon signed-rank test (Wilcoxon, 1945) (significance level: 0.05).

In Sections 4.1 and 4.2, we performed ablation studies where each component under study was removed or replaced, and the results were recomputed. For this purpose, we used the DRIVE database and focused on *Acc* and AUC. Regarding the tests of statistical significance, we evaluated the following null hypothesis: the results obtained with each variant are not statistically different from those of the *Base System*.

4.1. Validation of the Base System

We started by evaluating our *Base System* regarding data augmentation, prediction and regularization. The results of each variant are shown in Table 3, while the probabilistic predictions can be seen in Fig. 4. All tests were performed under the same conditions, with the only source of variability being the component under study.

4.1.1. Data augmentation

To validate the procedure described in Section 2.4, we studied four alternatives. At first, we reduced the total number of patches per image to 3000, by not performing data augmentation – *No Augmentation*. Then, we increased the total number of patches to 12,000 in three different ways. In the first case, we oversampled the image by extracting 9000 more original patches – *Oversampling*. In the second one, the remaining 9000 patches were artificially created by non-linearly deforming each patch, as described in Oliveira et al. (2017) – *Elastic Samples*. Each set of 3000 elastic patches was obtained using a different (α, σ) combination: (8, 1.5), (16, 2.5), or (32, 3). These values were found manually, ensuring that both the artificial samples and their respective annotations retain a consistent appearance. Finally, we used 9000 rotated patches, as in the *Base System*, but we placed them consecutively (not randomly) – *Consecutive Rotations*.

Considering the *Base System* as the reference, we can see that reducing the number of patches by four strongly deteriorated the results in terms of *Acc* and AUC. Besides this, either when using original or elastic patches to keep the initial number of samples, the differences to the reference remained almost the same. Looking directly at Fig. 4, we notice that these approaches favored the simultaneous appearance of FN and FP, with elastic patches leading to greater tortuosity in the detected vessel segments. Overall, this hints that the network benefited the most from the information encoded by the rotations. Another important note is related to the way those rotations were presented to the network. Recalling Section 2.4, we have seen that some data augmentation strategies designed for FC-CNNs (Dieleman et al., 2016; Worrall et al., 2016) require the rotated versions to be arranged consecutively. The *Consecutive Rotations* test hints that a deterministic proximity between the rotated patches is detrimental to the performance of the FCN, with the network showing more difficulties in detecting vessel segments as can be seen in Fig. 4.

When facing the randomly placed rotations used in the *Base System*, all the alternatives were found to be prejudicial with statistical significance.

4.1.2. Multiple prediction

After verifying the benefits that the rotated patches brought to the model, we settled out to investigate whether these benefits could be extrapolated to the prediction. Our idea was to synchronously excite the network with the original patches and their rotations, performing the average of the outputs, as described in Section 2.4.

Here, we compare the *Base System*, which uses our multiple prediction scheme, with the *Simple Prediction* variant. As can be seen in Table 3, the *Base System* performed better both in terms of *Acc* and AUC. Moreover, comparing the predictions of both approaches (Fig. 4), we notice that the multiple segmentation scheme makes the model less prone to FP, which is particularly important in medical applications. As a point of note, even if the changes in the mean values were slighter, statistically significant differences were found between the two methods.

4.1.3. Regularization

Finally, we looked at the regularization of the model, by comparing the spatial dropout technique, described in Section 2.3.5, with *Standard Dropout*.

[2] https://www.github.com/fchollet/keras.
[3] https://www.github.com/tensorflow/tensorflow.

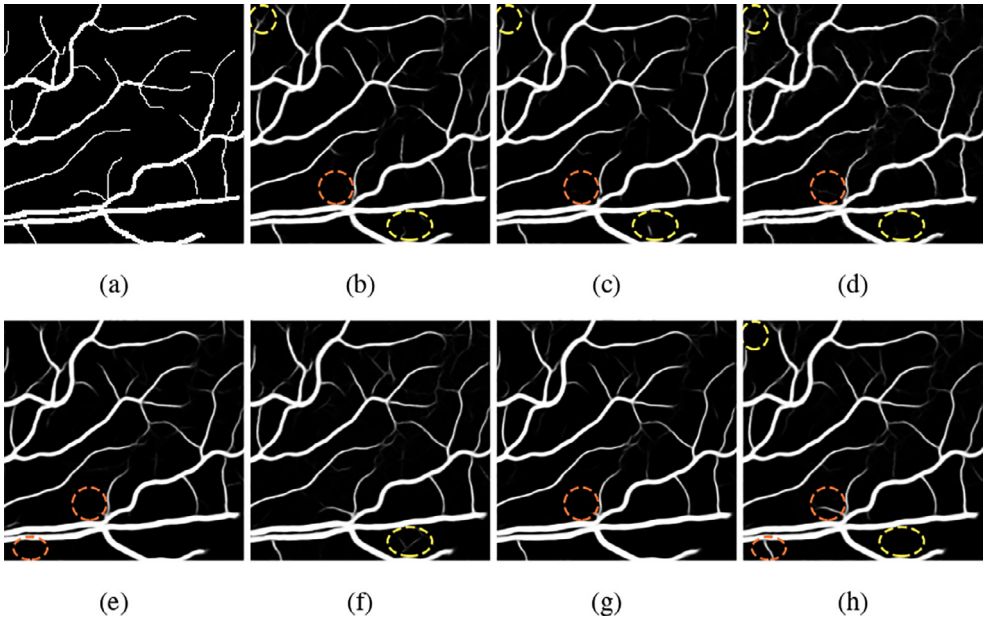


Fig. 4. Effects of the components of the Base System on the probabilistic predictions : (a) Segmentation of the 1st human observer; (b) No Augmentation; (c) Oversampling; (d) Elastic Samples; (e) Consecutive Rotations; (f) Simple Prediction; (g) Standard Dropout; (h) Base System. Yellow markings represent increase of false positives in relation to the Base System, while orange markings symbolize increase of false negatives. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

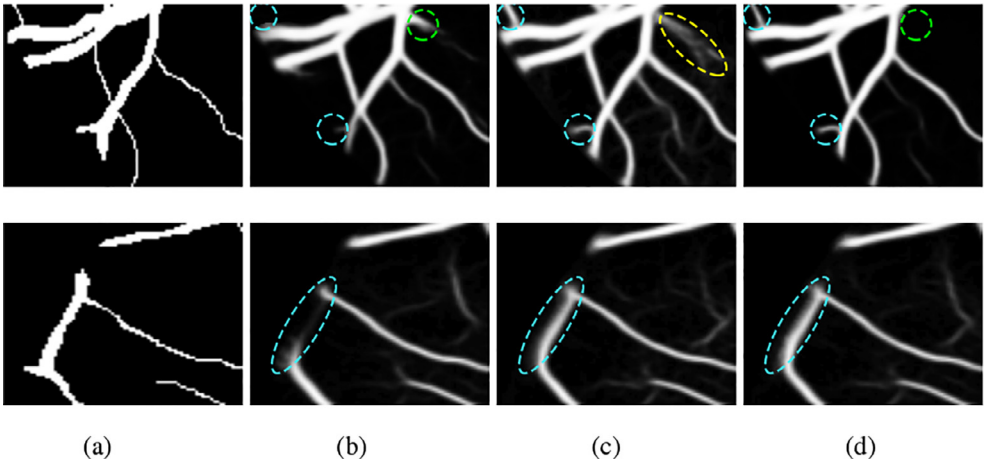


Fig. 5. Effects of the inclusion of the SWT channels on the probabilistic predictions of two different patches: (a) Segmentation of the 1st human observer; (b) Base System; (c) BS + d_1; (d) BS + d_2 (best model). Green markings represent reduction of false positives in relation to the Base System; blue markings indicate reduction of false negatives; yellow markings symbolize increase of false positives. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Still having the *Base System* as a reference, we can see that the standard strategy led to a statistically significant drop in terms of *Acc* and AUC (Table 3). Overall, the predictions are quite similar, but the model seems to detect fewer vessel segments when the standard dropout is applied (Fig. 4).

4.2. Validation of the Stationary Wavelet Transform

Having analyzed our *Base System*, we evaluate the effects of incorporating the SWT into it. The results of each variant are shown in Table 4, while the probabilistic predictions can be seen in Fig. 5. The patches coming from the SWT were concatenated in the input, by varying the total number of input channels. We resorted only to detail coefficients since the goal was to enhance image transitions. We started by concatenating the detail coefficients of the first level in the initial green channel input – $BS + d_1$. Then, we added those of the second level – $BS + d_1 + d_2$. Finally, only the latter were kept – $BS + d_2$.

Analyzing the results of the tests performed, we notice that all the alternative strategies improved the performance of the *Base System*, in terms of *Acc* and AUC. This means that the use of features based on the wavelet decomposition, whose effectiveness for vessel segmentation is well-known (Soares et al., 2006; Zhang et al., 2017), is also beneficial when combined with a deep learning methodology. In particular, we see that the first level SWT coefficients used on $BS + d_1$ were less effective than those of the second level applied on $BS + d_2$. The first level translates spectral information of higher frequencies; this seems to have induced more false positives, as can be seen in Fig. 5. On the other hand, the second level SWT coefficients introduced statistically significant differences to the *Base System*, in both *Acc* and AUC, which seems to reinforce the idea that even deep learning methods can benefit from

Table 4
Segmentation results of each test on the inclusion of SWT, on DRIVE. Bold values show the best score among all methods; underlined values represent metrics where the null hypothesis can be rejected (p-value < 0.05 when facing the Base System).

Method	Dropout	Patches per image			Prediction	Channels	*Acc*	AUC
		Original	Rotated	Elastic				
Base System (BS)	Spatial	3000	9000	–	Multiple	1	0.9570	0.9815
BS + d_1	Spatial	3000	9000	–	Multiple	4	0.9573	0.9818
BS + d_1 + d_2	Spatial	3000	9000	–	Multiple	7	0.9574	0.9820
BS + $\mathbf{d}_2$	Spatial	3000	9000	–	Multiple	4	**0.9576**	**0.9821**

* For concision of notation, d_j represents the set consisting of dV_j, dH_j, and dD_j.

Table 5
Segmentation results on the DRIVE, STARE, and CHASE_DB1 databases. Bold values show the best score among all methods.

			DRIVE				STARE				CHASE_DB1			
	Method	Year	*Sn*	*Sp*	*Acc*	AUC	*Sn*	*Sp*	*Acc*	AUC	*Sn*	*Sp*	*Acc*	AUC
	2nd human observer	–	0.7760	0.9725	0.9473	–	0.8956	0.9381	0.9346	–	0.7705	0.9778	0.9561	–
Unsupervised methods	Hoover et al. (2000)	2000	–	–	–	–	0.6794	0.9560	0.9263	–	–	–	–	–
	Mendonca and Campilho (2006)	2006	0.7344	0.9764	0.9452	–	0.6996	0.9730	0.9440	–	–	–	–	–
	Lam et al. (2010)	2010	–	–	0.9472	0.9614	–	–	0.9567	0.9739	–	–	–	–
	Yin et al. (2012)	2012	0.6522	0.9710	0.9267	–	0.7248	0.9666	0.9412	–	–	–	–	–
	Nguyen et al. (2013)	2013	–	–	0.9407	–	–	–	0.9324	–	–	–	–	–
	Wang et al. (2013)	2013	–	–	0.9461	0.9543	–	–	0.9521	0.9682	–	–	–	–
	Azzopardi et al. (2015)	2015	0.7655	0.9704	0.9442	0.9614	0.7716	0.9701	0.9497	0.9563	0.7585	0.9587	0.9387	0.9487
	Roychowdhury et al. (2015b)	2015	0.7395	0.9782	0.9494	0.9672	0.7317	0.9842	0.9560	0.9673	0.7615	0.9575	0.9467	0.9623
	Zhao et al. (2015)	2015	0.7420	0.9820	0.9540	0.8620	0.7800	0.9780	0.9560	0.8740	–	–	–	–
	Zhang et al. (2016)	2016	0.7743	0.9725	0.9476	0.9636	0.7791	0.9758	0.9554	0.9748	0.7626	0.9661	0.9452	0.9606
	Neto et al. (2017)	2017	0.7806	0.9629	–	–	**0.8344**	0.9443	–	–	–	–	–	–
Supervised methods	Niemeijer et al. (2004)	2004	–	–	0.9416	0.9294	–	–	–	–	–	–	–	–
	Staal et al. (2004)	2004	–	–	0.9441	0.9520	–	–	0.9516	0.9614	–	–	–	–
	Soares et al. (2006)	2006	0.7332	0.9782	0.9466	0.9614	0.7207	0.9747	0.9480	0.9671	–	–	–	–
	Marín et al. (2011)	2011	0.7067	0.9801	0.9452	0.9588	–	–	–	–	–	–	–	–
	Fraz et al. (2012b)	2012	0.7406	0.9807	0.9480	0.9747	0.7548	0.9763	0.9534	0.9768	0.7224	0.9711	0.9469	0.9712
	Roychowdhury et al. (2015a)	2015	0.7249	**0.9830**	0.9519	0.9620	0.7719	0.9726	0.9515	0.9688	0.7201	0.9824	0.9530	0.9532
	Strisciuglio et al. (2016)	2016	0.7777	0.9702	0.9454	0.9597	0.8046	0.9710	0.9534	0.9638	–	–	–	–
	Orlando et al. (2017)	2017	0.7897	0.9684	0.9454	0.9506	0.7680	0.9738	0.9519	0.9570	0.7565	0.9655	0.9467	0.9478
	Zhang et al. (2017)	2017	0.7861	0.9712	0.9466	0.9703	0.7882	0.9729	0.9547	0.9740	0.7644	0.9716	0.9502	0.9706
Deep Learning methods	Melinščak et al. (2015)	2015	0.7276	0.9785	0.9466	0.9749	–	–	–	–	–	–	–	–
	Li et al. (2016)	2016	0.7569	0.9816	0.9527	0.9738	0.7726	0.9844	0.9628	0.9879	0.7507	0.9793	0.9581	0.9716
	Liskowski and Krawiec (2016)	2016	0.7520	0.9806	0.9515	0.9710	0.8145	**0.9866**	**0.9696**	0.9880	–	–	–	–
	Fu et al. (2016)	2016	0.7603	–	0.9523	–	0.7412	–	0.9585	–	0.7130	–	0.9489	–
	Proposed	2018	**0.8039**	0.9804	**0.9576**	**0.9821**	0.8315	0.9858	0.9694	**0.9905**	**0.7779**	**0.9864**	**0.9653**	**0.9855**

domain knowledge. In fact, they allowed to reduce the combination of false positives and false negatives as we can see in Fig. 5 as well. From now on, we will refer to the best model ($BS + d_2$) as *Proposed*.

4.3. Vessel segmentation

The binary segmentations were obtained by thresholding the probability maps at 0.5. We note that only pixels inside the field of view (FOV) were used in calculations.

The results for each database are shown in Table 5. The mean *Acc* values obtained were higher than those of the second observer in all databases. The same occurred regarding *Sp*, which reveals that the network presented few false positives, not signaling areas of injury, or spills, as vessels. In the DRIVE and CHASE_DB1 databases, the mean *Sn* surpassed that of the second observer, which shows that the network also rarely misclassified vessel pixels. In the STARE database, this tendency was not strong enough to level with the second observer, since he systematically marks vessels that the first one does not see.

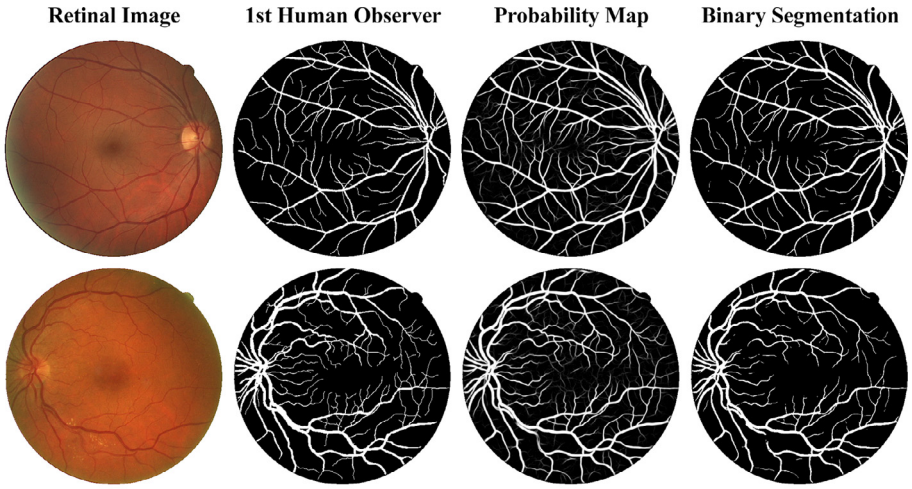
The *Sn*, *Sp*, *Acc*, and AUC values of the best case for the DRIVE database were 0.9119, 0.9742, 0.9667, and 0.9903, while those of the worst case matched 0.7628, 0.9816, 0.9497, and 0.9786, respectively (Fig. 6a). For the STARE database, the values of the best case were 0.8527, 0.9936, 0.9837, and 0.9964, while those of the worst case matched 0.7231, 0.9827, 0.9503, and 0.9791 (Fig. 6b). Finally, for the CHASE_DB1 database, the values of the best case were 0.8541, 0.9844, 0.9744, and 0.9909, while those of the worst case matched 0.8065, 0.9749, 0.9574, and 0.9808 (Fig. 6c).

Regarding computational performance, the training of the network took about 4 h, and each retinal image was fully segmented in approximately 2 s.

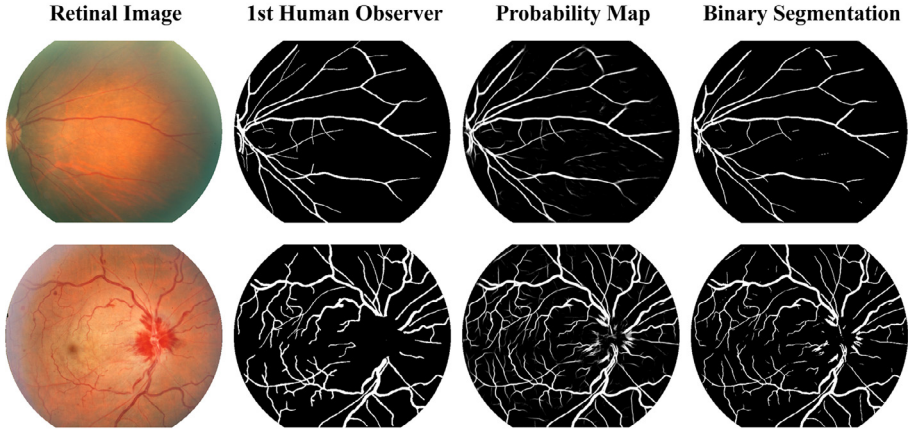
4.4. Comparison with the state-of-the-art

We compare our method with several other state-of-the-art methods in Table 5.

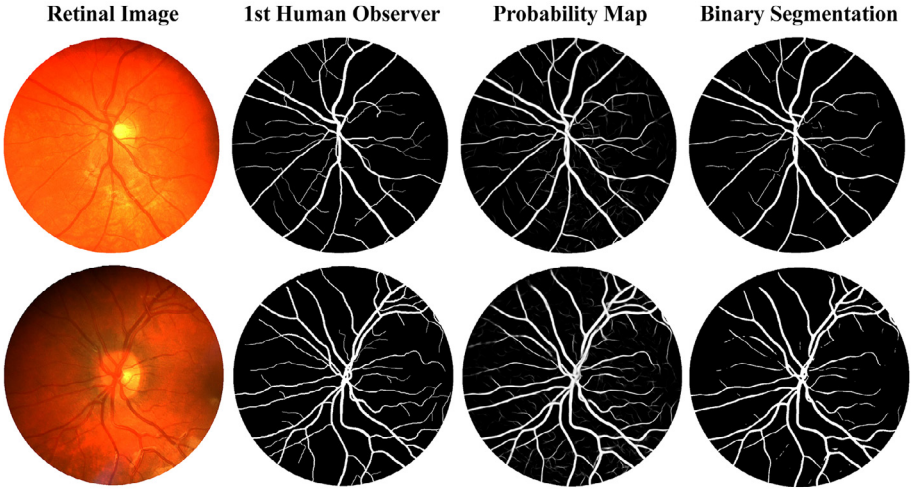
We observe that our method obtained the highest *Sn* in two of the three databases used (DRIVE and CHASE_DB1), being very close to the best result in the third one (STARE), which indicates that it was capable of detecting many vessel pixels; also, considering the high value of *Sp*, this was not obtained by increasing the number of false positives. This is significant, because training a classifier for segmenting retinal vessels becomes more difficult due to the unbalanced classes, where vessel pixels typically correspond only to 10% of the image, and the detection of vessels without increasing false cases is essential for medical applications.



(a)



(b)



(c)

Fig. 6. Segmentation examples for each database: (a) DRIVE; (b) STARE; (c) CHASE_DB1. The first row shows the best case, while the second presents the worst one.

By jointly evaluating *Sn, Sp*, and *Acc*, our method presented the best performance on both the DRIVE and CHASE_DB1 databases. In the DRIVE database, we outperformed all the other works regarding *Sn* and *Acc*. In terms of *Sp*, we stood in sixth; however, knowing that *Acc* combines information from *Sn* and *Sp*, we may conclude that the gain in true detections was much more significant than the inclusion of false detections. In other words, there is a notorious trade-off between *Sn* and *Sp*, with our method having a better balance. In the CHASE_DB1 database, we ranked first in all metrics. Regarding the STARE database, we ranked second in all metrics, only behind Neto et al. (2017) in *Sn* and Liskowski and Krawiec (2016) in *Sp* and *Acc*. By comparing our work with Neto et al. (2017), we see that our *Sp* is much higher so, given the trade-off abovementioned, the slight disadvantage in terms of *Sn* sounds natural. In parallel, if we consider the second row of Fig. 6b, which shows our worst case, we notice that our method failed in rejecting small haemorrhagic blobs, which may have affected *Sp*. This may be explained by our choice of using 5-fold cross-validation for training, which reduced the number of images with pathological signs, contrary to Liskowski and Krawiec (2016), which used leave-one-out cross-validation.

Among the methods based on deep learning, only Fu et al. (2016) also used a multiscale FCN. If we compare this method with our *Base System*, which did not use the extra channels from SWT, we verify that Fu et al. (2016) was surpassed in all available metrics. This means that the simpler structure of our encoder/decoder, when properly trained, was able to outperform a more elaborated architecture that joined a multiscale FCN with a CRF, and was trained end-to-end with a structured loss function. Comparing with all deep learning methods, we verify that our approach presented a much higher *Sn* in all databases, and a much higher *Acc* in DRIVE and CHASE_DB1 databases. In DRIVE, which is perhaps the most used database in this area, we can see that before this work none of the previous best methods in *Sn, Sp*, and *Acc*, was based on deep learning.

Contrary to *Sn, Sp*, and *Acc*, the AUC score is not dependent on the threshold used to produce the final segmentations. Taking this into account, we note that our method achieved the highest AUC value in all databases. Also, to the best of our knowledge, we are the first to report AUC values above 0.98 for all databases.

4.5. Cross-training

In a realistic situation, it is not feasible to retrain the model whenever new images have to be segmented. Additionally, a reliable method must successfully segment each image, even if the acquisition device belongs to a different manufacturer. That being said, robustness to the training set is crucial for the model to be of practical use. In this study, we did cross-training between the DRIVE and STARE databases, since they are the most widely used for this purpose (Fraz et al., 2012b; Li et al., 2016; Marín et al., 2011; Roychowdhury et al., 2015a; Soares et al., 2006; Zhang et al., 2017). The results are shown in Table 6.

The *Acc* average values went down from 0.9576 and 0.9694 to 0.9505 and 0.9597 for the DRIVE and STARE databases, respectively. This means decreases of about 0.7% and 1.0%, against the 0.2% and 0.1% of Roychowdhury et al. (2015a). Regarding AUC, the results fell from 0.9821 and 0.9905 to 0.9748 and 0.9846, by the same order. This implies approximate reductions of 0.7% and 0.6%, against the 0.5% and 1.1% of Fraz et al. (2012b).

As shown in Table 6, in absolute terms, we ranked first in all metrics, except *Sn*, when training on STARE and testing on DRIVE. In the reverse case, something similar happened, but *Sp* replaced *Sn*. We have found that when training on STARE and testing on DRIVE, the network detects fewer thin vessels, so there was a drop in *Sn*. In the reverse case, since the DRIVE database typically has more annotated thin vessels than STARE, *Sn* rose considerably.

4.6. Robustness to the inter-rater variability

The manual annotations provided by physicians are crucial for training and testing supervised methods. Experience may help experts refine their segmentations, but two different observers will always have their own way of approaching the task. There is not always agreement between the experts since some of them systematically see more vessels than others. Moreover, even when the vessel is clear, differences in the estimation of its caliber may arise. For these reasons, inter-rater variability is always present in the evaluation process. Here, we discuss how this variability affects the obtained results.

All the previous results reported in this work were obtained using the annotations of the first human observer as the gold standard for training and testing. In Table 7, otherwise, the gold standard, for testing, were the annotations of the second human observer. Having these as ground truth, we then compared both the first observer and the model, for each database. Regarding the tests of statistical significance, we aimed to reject the following null hypothesis: the results of the *Proposed* method are not statistically different from those of the first human observer.

As shown in Table 7, if we consider the second observer as the gold standard, the model outperformed the first observer regarding *Sp* and *Acc*, in the STARE and CHASE_DB1 databases. According to the definition of *Sp*, this means that it introduced fewer false positives than the first observer himself (in relation to the second one). In other words, some of the pixels marked as background by the model, and rejected by the first observer, ended up being validated by the second one. In the DRIVE database, the model performed better than the first observer in terms of *Sn, Sp*, and *Acc*. This shows that apart from having less false positives, it also presented less false negatives than the first observer. That is, even though the model was trained according to the first observer, it could identify vessel segments that only the second observer saw. Together, the previous results suggest that even when the model was trained according to the first observer, it seemed to be more consistent than him when another observer was considered. To put it another way, the differences established between two independent human observers appeared to be higher than those presented by the model when it was evaluated against a new reference. This reveals a consistency which may be unreachable to a human who always oscillates from day to day and can be affected by fatigue or stress.

Comparing Tables 5 and 7, we can further analyze how the model behaved when we changed the reference from the first to the second observer. For the STARE and CHASE_DB1 databases, the average *Acc* of the model fell from 0.9694 and 0.9653 to 0.9365 and 0.9600, respectively. This means approximate decreases of 3.4% and 0.5%, which sound natural since it was trained according to the first observer. For the STARE database, in particular, the decrease was higher since the second observer systematically marks more vessels than the first one. The DRIVE database case was surprising. The average *Acc* rose from 0.9576 to 0.9639, which corresponds to an approximate increase of 0.7%. This means that despite being trained according to the first observer, the model was closer to the second one.

Except for the STARE database, in both the remaining databases there is no clear difference between the marking pattern of the experts. That is, we can see cases where the first observer marks vessels that the second one does not see and vice versa. This may be deduced from Fig. 7, where we compare the annotations of both observers, for each database. We believe that this noticeable dis-

Table 6
Cross-training segmentation results between the DRIVE and STARE databases. Bold values show the best score among all methods.

Test set (Training set)	Method	*Sn*	*Sp*	*Acc*	AUC
DRIVE (STARE)	Soares et al. (2006)	–	–	0.9397	–
	Marín et al. (2011)	–	–	0.9448	–
	Fraz et al. (2012b)	0.7242	0.9792	0.9456	0.9697
	Roychowdhury et al. (2015a)	–	–	0.9494	–
	Li et al. (2016)	**0.7273**	0.9810	0.9486	0.9677
	Zhang et al. (2017)	–	–	0.9447	0.9593
	Proposed	0.6706	**0.9916**	**0.9505**	**0.9748**
STARE (DRIVE)	Soares et al. (2006)	–	–	0.9327	–
	Marín et al. (2011)	–	–	0.9526	–
	Fraz et al. (2012b)	0.7010	0.9770	0.9495	0.9660
	Roychowdhury et al. (2015a)	–	–	0.9510	–
	Li et al. (2016)	0.7027	**0.9828**	0.9545	0.9671
	Zhang et al. (2017)	–	–	0.9488	0.9676
	Proposed	**0.8453**	0.9726	**0.9597**	**0.9846**

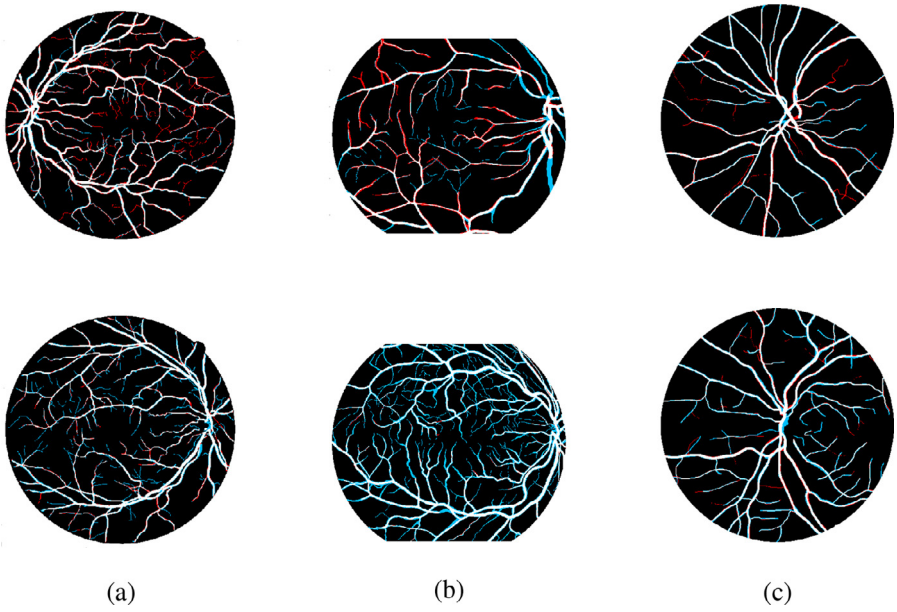


Fig. 7. Comparison between the annotations of the 1st and 2nd human observers for each database: (a) DRIVE; (b) STARE; and (c) CHASE_DB1. The first row shows the comparison for the case where the model improves the most being evaluated according to the 2nd observer (with the *Acc* values for the model/ 1st observer being, from (a) to (c): 0.9692/0.9467, 0.9647/0.9527, and 0.9643/0.9613); the second row refers to the case where it most worsens (with the *Acc* values becoming: 0.9582/0.9446, 0.9028/0.8968, and 0.9492/0.9480). Each color represents a different situation: black – both mark as background; white – both mark as vessel; red – only the 1st observer marks as vessel; blue – only the 2nd observer marks as vessel. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 7
Segmentation results on the DRIVE, STARE, and CHASE_DB1 databases, using the annotations of the 2nd human observer as gold standard for testing. We also evaluate the 1st human observer in relation to the 2nd. Bold values show the best score between the two methods; underlined values represent metrics where the null hypothesis can be rejected (p-value < 0.05 when they are confronted).

Test set	Method	*Sn*	*Sp*	*Acc*
DRIVE	1st human observer	0.8066	0.9674	0.9473
	Proposed	**0.8405**	**0.9814**	**0.9639**
STARE	1st human observer	**0.6439**	0.9883	0.9346
	Proposed	0.6329	**0.9924**	**0.9365**
CHASE_DB1	1st human observer	**0.7974**	0.9736	0.9561
	Proposed	0.7731	**0.9813**	**0.9600**

cordance may become a limiting factor for future improvements since the current results of this field are already extremely accurate.

5. Conclusions

In this paper, we proposed a novel FCN-based method for retinal vessel segmentation. We used rotation operations for data augmentation and introduced a new way to use the information they provide, during training, to strengthen the prediction. Moreover, we investigated the addition of new channels into a FCN through the SWT decomposition. This allowed to improve the performance in terms of Acc and AUC, reducing the combination of FP and FN and suggesting that deep learning methods still leave room for domain knowledge. Results on three publicly available databases showed that we are competitive with state-of-the-art methods. In terms of *Acc*, we rank first on the DRIVE and CHASE_DB1 databases, while being second in STARE. In terms of AUC, we lead on all of them.

Our method proved to be robust to the training set and to the inter-rater variability, which shows its potential for real-world application on screening and diagnostic systems. Due to its simple

convolutional nature, and by using an efficient GPU implementation, it also proves to be fast. Fully segmenting a retinal image takes approximately 2 s.

Based on our vessel segmentation results, it may be possible to extract different biomarkers from retinal images and apply them for clinical purposes. Also, the proposed method can be further applied on other types of medical images for segmenting elongated structures.

Although the proposed approach shows good performance, there are still some aspects that can be further explored in the future. In this work, we showed that extra features were beneficial for the segmentation performance. Hence, domain knowledge may improve Deep Learning-based models. So, as long term research, we will explore further the combination of Deep Learning with domain knowledge. In retinal imaging, we believe that other wavelet families, such as curvelets, may also prove useful. Finally, the amount of training data may limit the capacity of the CNNs. Hence, we expect that as access to medical data becomes easier, we will be able to design different architectures.

Acknowledgments

The authors would like to thank the suggestions of the Anonymous Reviewers that helped to improve this document. This work is supported by FCT with the reference project UID/EEA/04436/2013, by FEDER, Portugal funds through the COMPETE 2020 Programa Operacional Competitividade e Internacionalização (POCI) with the reference project POCI-01-0145-FEDER-006941.

References

Abràmoff, M. D., Garvin, M. K., & Sonka, M. (2010). Retinal imaging and image analysis. *IEEE Reviews in Biomedical Engineering, 3*, 169–208.

Azzopardi, G., Strisciuglio, N., Vento, M., & Petkov, N. (2015). Trainable cosfire filters for vessel delineation with application to retinal images. *Medical Image Analysis, 19*(1), 46–57.

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2015). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv:1511.00561.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35*(8), 1798–1828.

Dai, J., He, K., & Sun, J. (2015). Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision* (pp. 1635–1643).

Dieleman, S., De Fauw, J., & Kavukcuoglu, K. (2016). Exploiting cyclic symmetry in convolutional neural networks. arXiv:1602.02660.

Dieleman, S., Willett, K. W., & Dambre, J. (2015). Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly Notices of the Royal Astronomical Society, 450*(2), 1441–1459.

Duan, Y., Liu, F., Jiao, L., Zhao, P., & Zhang, L. (2017). SAR image segmentation based on convolutional-wavelet neural network and Markov random field. *Pattern Recognition, 64*, 255–267.

Eigen, D., & Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi- convolutional architecture. In *Proceedings of the ieee international conference on computer vision* (pp. 2650–2658).

Fraz, M. M., Remagnino, P., Hoppe, A., Uyyanonvara, B., Rudnicka, A. R., Owen, C. G., & Barman, S. A. (2012a). Blood vessel segmentation methodologies in retinal images – a survey. *Computer Methods and Programs in Biomedicine, 108*(1), 407–433.

Fraz, M. M., Remagnino, P., Hoppe, A., Uyyanonvara, B., Rudnicka, A. R., Owen, C. G., & Barman, S. A. (2012b). An ensemble classification-based approach applied to retinal blood vessel segmentation. *IEEE Transactions on Biomedical Engineering, 59*(9), 2538–2548.

Fu, H., Xu, Y., Lin, S., Wong, D. W. K., & Liu, J. (2016). Deepvessel: Retinal vessel segmentation via deep learning and conditional random field. In *Miccai* (pp. 132–139). Springer.

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks.. In *AISTATS: Vol. 9* (pp. 249–256).

Guo, T., Mousavi, H. S., Vu, T. H., & Monga, V. (2017). Deep wavelet prediction for image super-resolution. In *The IEEE conference on computer vision and pattern recognition (CVPR) workshops*.

Haar, A. (1910). Zur theorie der orthogonalen funktionensysteme. *Mathematische Annalen, 69*(3), 331–371.

Hariharan, B., Arbeláez, P., Girshick, R., & Malik, J. (2015). Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 447–456).

Holschneider, M., Kronland-Martinet, R., Morlet, J., & Tchamitchian, P. (1990). A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets* (pp. 286–297). Springer.

Hoover, A., Kouznetsova, V., & Goldbaum, M. (2000). Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. *IEEE Transactions on Medical Imaging, 19*(3), 203–210.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems* (pp. 1097–1105).

Lam, B. S., Gao, Y., & Liew, A. W.-C. (2010). General retinal vessel segmentation using regularization-based multiconcavity modeling. *IEEE Transactions on Medical Imaging, 29*(7), 1369–1381.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436–444.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., & Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems* (pp. 396–404).

Li, Q., Feng, B., Xie, L., Liang, P., Zhang, H., & Wang, T. (2016). A cross-modality learning approach for vessel segmentation in retinal images. *IEEE Transactions on Medical Imaging, 35*(1), 109–118.

Liskowski, P., & Krawiec, K. (2016). Segmenting retinal blood vessels with deep neural networks. *IEEE Transactions on Medical Imaging, 35*(11), 2369–2380.

Liu, I., & Sun, Y. (1993). Recursive tracking of vascular networks in angiograms based on the detection-deletion scheme. *IEEE Transactions on Medical Imaging, 12*(2), 334–341.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *CVPR* (pp. 3431–3440).

Marín, D., Aquino, A., Gegúndez-Arias, M. E., & Bravo, J. M. (2011). A new supervised method for blood vessel segmentation in retinal images by using gray-level and moment invariants-based features. *IEEE Transactions on Medical Imaging, 30*(1), 146–158.

Melinščak, M., Prentašić, P., & Lončarić, S. (2015). Retinal vessel segmentation using deep neural networks. In *VISAPP*.

Mendonca, A. M., & Campilho, A. (2006). Segmentation of retinal blood vessels by combining the detection of centerlines and morphological reconstruction. *IEEE Transactions on Medical Imaging, 25*(9), 1200–1213.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In *ICML* (pp. 807–814).

Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate o (1/k2). In *Soviet mathematics doklady: Vol. 27* (pp. 372–376).

Neto, L. C., Ramalho, G. L., Neto, J. F. R., Veras, R. M., & Medeiros, F. N. (2017). An unsupervised coarse-to-fine algorithm for blood vessel segmentation in fundus images. *Expert Systems with Applications, 78*, 182–192.

Nguyen, U. T., Bhuiyan, A., Park, L. A., & Ramamohanarao, K. (2013). An effective retinal blood vessel segmentation method using multi- line detection. *Pattern Recognition, 46*(3), 703–715.

Niemeijer, M., Staal, J., van Ginneken, B., Loog, M., & Abramoff, M. D. (2004). Comparative study of retinal vessel segmentation methods on a new publicly available database. In *Medical imaging: Vol. 5370* (pp. 648–656). SPIE.

Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision* (pp. 1520–1528).

Oliveira, A., Pereira, S., & Silva, C. A. (2017). Augmenting data when training a cnn for retinal vessel segmentation: How to warp? In *ENBENG* (pp. 1–4). IEEE.

Orlando, J. I., Prokofyeva, E., & Blaschko, M. B. (2017). A discriminatively trained fully connected conditional random field model for blood vessel segmentation in fundus images. *IEEE Transactions on Biomedical Engineering, 64*(1), 16–27.

Owen, C. G., Rudnicka, A. R., Mullen, R., Barman, S. A., Monekosso, D., Whincup, P. H., et al. (2009). Measuring retinal vessel tortuosity in 10-year-old children: validation of the computer-assisted image analysis of the retina (CAIAR) program. *Investigative Ophthalmology & Visual Science, 50*(5), 2004–2010.

Patton, N., Aslam, T. M., MacGillivray, T., Deary, I. J., Dhillon, B., Eikelboom, R. H., et al. (2006). Retinal image analysis: Concepts, applications and potential. *Progress in Retinal and Eye Research, 25*(1), 99–127.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *MICCAI* (pp. 234–241). Springer.

Roychowdhury, S., Koozekanani, D. D., & Parhi, K. K. (2015a). Blood vessel segmentation of fundus images by major vessel extraction and subimage classification. *IEEE Journal of Biomedical and Health Informatics, 19*(3), 1118–1128.

Roychowdhury, S., Koozekanani, D. D., & Parhi, K. K. (2015b). Iterative vessel segmentation of fundus images. *IEEE Transactions on Biomedical Engineering, 62*(7), 1738–1749.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-image recognition. arXiv:1409.1556.

Soares, J. V., Leandro, J. J., Cesar, R. M., Jelinek, H. F., & Cree, M. J. (2006). Retinal vessel segmentation using the 2d gabor wavelet and supervised classification. *IEEE Transactions on Medical Imaging, 25*(9), 1214–1222.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting.. *Journal of Machine Learning Research, 15*(1), 1929–1958.

Staal, J., Abràmoff, M. D., Niemeijer, M., Viergever, M. A., & van Ginneken, B. (2004). Ridge-based vessel segmentation in color images of the retina. *IEEE Transactions on Medical Imaging, 23*(4), 501–509.

Strisciuglio, N., Azzopardi, G., Vento, M., & Petkov, N. (2016). Supervised vessel delineation in retinal fundus images with the automatic selection of b-cosfire filters. *Machine Vision and Applications, 27*(8), 1137–1149.

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., & Bregler, C. (2015). Efficient object localization using convolutional networks. In *CVPR* (pp. 648–656).

Wang, Y., Ji, G., Lin, P., & Trucco, E. (2013). Retinal vessel segmentation using multi-wavelet kernels and multi hierarchical decomposition. *Pattern Recognition, 46*(8), 2117–2133.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin, 1*(6), 80–83.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., & Brostow, G. J. (2016). Harmonic networks: Deep translation and rotation equivariance. arXiv:1612.04642.

Yin, Y., Adel, M., & Bourennane, S. (2012). Retinal vessel segmentation using a probabilistic tracking method. *Pattern Recognition, 45*(4), 1235–1244.

Zhang, J., Chen, Y., Bekkers, E., Wang, M., Dashtbozorg, B., & ter Haar Romeny, B. M. (2017). Retinal vessel delineation using a brain-inspired wavelet transform and random forest. *Pattern Recognition, 69*, 107–123.

Zhang, J., Dashtbozorg, B., Bekkers, E., Pluim, J. P., Duits, R., & ter Haar Romeny, B. M. (2016). Robust retinal vessel segmentation via locally adaptive derivative frames in orientation scores. *IEEE Transactions on Medical Imaging, 35*(12), 2631–2644.

Zhao, Y., Rada, L., Chen, K., Harding, S. P., & Zheng, Y. (2015). Automated vessel segmentation using infinite perimeter active contour model with hybrid region information with application to retinal images. *IEEE Transactions on Medical Imaging, 34*(9), 1797–1807.