City Research Online

City, University of London Institutional Repository

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

M2U-Net: Effective and Efficient Retinal Vessel Segmentation for Real-World Applications

Tim Laibacher
City, University of London
tim.laibacher@city.ac.uk

Tillman Weyde
City, University of London
t.e.weyde@city.ac.uk

Sepehr Jalali
University College London
s.jalali@ucl.ac.uk

Abstract

In this paper, we present a novel neural network architecture for retinal vessel segmentation that improves over the state of the art on two benchmark datasets, is the first to run in real time on high resolution images, and its small memory and processing requirements make it deployable in mobile and embedded systems.

The M2U-Net has a new encoder-decoder architecture that is inspired by the U-Net. It adds pretrained components of MobileNetV2 in the encoder part and novel contractive bottleneck blocks in the decoder part that, combined with bilinear upsampling, drastically reduce the parameter count to 0.55M compared to 31.03M in the original U-Net.

We have evaluated its performance against a wide body of previously published results on three public datasets. On two of them, the M2U-Net achieves new state-of-the-art performance by a considerable margin. When implemented on a GPU, our method is the first to achieve real-time inference speeds on high-resolution fundus images. We also implemented our proposed network on an ARM-based embedded system where it segments images in between 0.6 and 15 sec, depending on the resolution. Thus, the M2U-Net enables a number of applications of retinal vessel structure extraction, such as early diagnosis of eye diseases, retinal biometric authentication systems, and robot assisted microsurgery.

1. Introduction

The retinal vasculature in each human eye is unique and, in the absence of pathologies, stays unaltered throughout the lifetime [34, 41]. This trait makes the automatic segmentation of the vessel structure suitable for two main applications:

First, in the early detection of diseases that affect the vessel structure, such as diabetic retinopathy and wet age-related macular degeneration. Diabetic retinopathy can cause the growth of new blood vessels. Wet macular degeneration can result in atherosclerosis, which can lead to the narrowing of blood vessels and can affect the arteries to vein ratio [2, 20].

Second, in biometric authentication systems, where the vessel structure is commonly used as a feature pattern by itself [17, 31] or as a preprocessing step to extract feature points based on landmarks like bifurcations and crossovers of retinal vessels [3, 4, 34]. To achieve retinal identification, good segmentation of the principal vessels of each individual is necessary [17]. An emerging third application is in robotic intraocular microsurgery, where the instruments location relative to the retinal vessel structure is tracked in real time as proposed by Braun *et al*. [11].

For real-world deployments of vessel segmentation methods, not only the segmentation quality, but also model size and computational requirements in terms of memory and processing power are important factors. In a real-world retinal authentication system for example, it would be infeasible to deploy expensive and power-hungry server grade GPUs, often consuming more than 200W, at every point of authentication. Additionally, the usage of cloud-computing resources in such a setting is prohibitive due to higher latency and privacy concerns, arising from the processing of large amounts of sensitive personal biometric data in the form of retina fundus images. By deriving feature patterns of the retina vasculature directly at the point of authentication, without the actual fundus images leaving the device, the risk of leaking or exposing this sensitive information can be reduced.

In this work we propose a novel neural network architecture, called M2U-Net, for retinal vasculature segmentation. M2U-Net reaches state of the art quality and is small and fast enough for use in embedded and mobile environments, enabling use-cases in biometrics and mobile diagnosis. Our specific contributions are as follows:

- The M2U-Net architecture, which builds on pretrained components of MobileNetV2 [39], is inspired by the U-Net architecture [37] and introduces contracting bottleneck blocks.

- The proposed network is tested on three publicly available retina fundus image datasets with annotated vessel ground truth labels: DRIVE [43], CHASE_DB1 [18] and HRF [23].

- We demonstrate that M2U-Net achieves new state-of-the-art results on HRF and CHASE_DB1 and new state of the art for mobile/embedded implementations on DRIVE. Our work is the first to reach super-human performance on CHASE_DB1 and real-time inference speeds on HRF and CHASE_DB1.

The remainder of this paper is structured as follows: Section 2 reviews related work with a focus on deep neural networks and efficient models. Section 3 introduces the M2U-Net architecture. Section 4 describes the implementation details and datasets. Section 5 describes the results together with an additional ablation study. Section 6 provides the conclusion with further research avenues.

2. Related work

The task of retinal blood vessel segmentation falls into the computer vision subcategory of semantic segmentation, which in recent years has seen tremendous improvements in performance thanks to the introduction of novel deep neural network architectures [27, 36, 37]. Similarly, recent state-of-the-art methods in retinal blood vessel segmentation that focus on segmentation quality are dominated by various variations of deep neural networks [26, 29, 45].

Liskowski and Krawiec [26] propose a patch-based approach, where a network that consists of a stack of convolutional layers followed by three fully-connected layers is trained with small patches of the input fundus image. In contrast, Maninis *et al*. [29] and Yan *et al*. [45] train their networks with complete fundus images.

Maninis *et al*. [29] extract intermediate feature maps of a VGG-16 network [42], pretrained on ImageNet, which are upsampled via transposed convolutions and concatenated before applying a final 1×1 convolutional layer. Their method, called DRIU, achieves a state-of-the-art Dice score of 0.822 on DRIVE. Yan *et al*. [45] train the U-Net model [37] with a joint-loss by appending two separate branches, one with a pixel-wise and one with a segment-level loss, that are trained simultaneously.

While these supervised methods accomplish good segmentation results, their computational requirements are substantial and as a consequence are commonly implemented on high-performance server-grade GPUs such as the NVIDIA TITAN class GPUs in [26, 29, 45].

Additionally they either fail to reach the performance of unsupervised methods on very high-resolution datasets or, as a result of their computational requirements, can only be trained with small patches of the complete input image that fit into memory and thereby further increase the time it takes

to segment the complete fundus image. For example, on the high resolution HRF dataset, the unsupervised method introduced by Annunziata *et al*. [5] achieves a state-of-the art Dice score of 0.7578, while the best-performing supervised method [45] achieves a Dice score of 0.7212.

Furthermore unsupervised methods are still dominant in works that focus on embedded or mobile systems and on execution speed [6, 8, 9, 24]. These methods often rely on matched filtering, contour tracing and morphological transformation techniques. In [6], a combination of matched filtering and contour tracing is proposed and implemented on GPU, realizing an execution time of 10ms on DRIVE. FPGA implementations of matched filtering based methods are introduced in [8] and [24] that accomplish execution times of 2ms and 52.3 ms respectively. In addition to morphological transformations, Bibiloni *et al*. [9] use CLAHE and hysteresis thresholding to segment the retinal vasculature and report single-core execution speeds of 37ms on DRIVE, using a Intel i5-3340 CPU.

To the best of our knowledge, [44] is the only existing work that has been implemented and tested on a mobile device, a Samsung Galaxy S5. The algorithm they introduced is based on visual saliency and incorporates orientation, morphological, spectral and intensity features and has an execution time of 118s on DRIVE images.

3. Model architecture

In this work we address this gap between unsupervised and supervised methods in retinal vessel segmentation by introducing a light-weight supervised method that can be deployed in embedded systems, can be trained on very high resolution images and produces competitive segmentation quality.

Like the U-Net architecture, our proposed network can be conceptually divided into two parts: a contracting encoder part and an expanding decoder part. For the encoder part, we employ the first fourteen layers of MobileNetV2 that were pretrained on ImageNet. The key building blocks, introduced by Sandler *et al*. [39] in the MobileNetV2 network, are inverted bottleneck blocks with Stride=1 and Stride=2, which are illustrated in Figure 1. Bottlenecks blocks with Stride=1 contain a residual connection if the number of input channels d' is equal to the number of output channels d''. Bottleneck blocks with Stride=2 do not have a residual connection. Both types of blocks start with a 1×1 convolution that expands the number of channels d by a factor t, followed by a depthwise separable convolution [40] that consists of a depthwise convolution and a pointwise 1×1 convolution.

In the decoder part we make use of the same Stride=1 bottleneck blocks as used in the encoder, but this time with a contracting factor of $t = 0.15$ instead of using an expansion factor of $t = 6$. These contracting bottleneck blocks do not
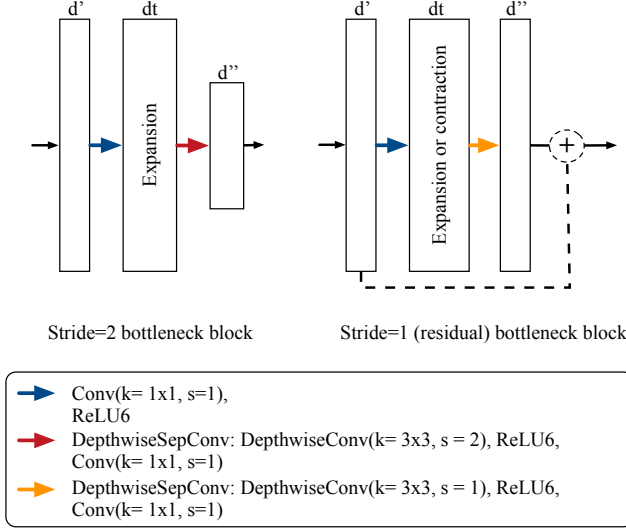
Figure 1: Bottleneck blocks: each box represents a multi-channel feature map. The height corresponds to the resolution, the width to the number of channels. Best viewed in color.

Input	Operator	t	c	n	s	Params
3×544^2	conv	-	32	1	2	928
32×272^2	dwisesep	1	16	1	1	896
16×272^2	bottleneck	6	24	1	2	5,136
16×136^2	resbottleneck	6	24	1	1	8,832
24×136^2	bottleneck	6	32	1	2	10,000
32×68^2	resbottleneck	6	32	2	1	29,696
32×68^2	bottleneck	6	64	1	2	21,056
64×34^2	resbottleneck	6	64	3	1	162,816
64×34^2	bottleneck	6	96	1	1	66,624
96×34^2	resbottleneck	6	96	2	1	236,544
96×34^2	upconcat	-	128	1	-	-
128×68^2	bottleneck	0.15	64	1	1	4,023
64×68^2	upconcat	-	88	1	-	-
88×136^2	bottleneck	0.15	44	1	1	1973
44×136^2	upconcat	-	60	1	-	-
60×272^2	bottleneck	0.15	30	1	1	987
30×272^2	upconcat	-	33	1	-	-
33×544^2	bottleneck	0.15	1	1	1	237
1×544^2	sigmoid	-	1	1	-	-

Table 1: M2U-Net: Each row corresponds to an operation that uses stride s, expansion factor t, is repeated n times and outputs c channels. The encoder part consists of an initial convolution (**conv**) and a depthwise separable convolution (**dwisesep**) followed by a stack of **(residual) bottleneck** blocks. The encoder part consists of repeated instances of upsampling and concatenation operations (**upconcat**) followed by **bottleneck** operations. Encoder bottleneck blocks expand with a factor $t = 6$, decoder bottleneck blocks contract with a factor $t = 0.15$. Batch normalization is used after each convolutional operation. A final **sigmoid** layer converts the raw logits into probabilities.

contain a residual connection. The contracting bottleneck blocks allow us to further reduce the amount of trainable parameters. Additionally we follow the recommendation of Fauw *et al.* [16] and use parameter-free bilinear upsampling instead of transposed convolutional operations to upsample the spatial resolution of feature maps. Table 1 and Figure 2 describe and illustrate the proposed architecture in more detail [1].

4. Implementation details

4.1. Training setup and datasets

The proposed architecture is implemented in PyTorch. We use AdamW [28] as optimization method with a learning rate of 0.001. A combined binary cross-entropy and Jaccard loss function L_{JBCE} with a weighting factor $w = 0.3$ as proposed in [21] is utilized:

$$L_{JBCE} = L_{BCE} + w * (1 - J), \qquad (1)$$

where L_{BCE} is the binary cross entropy loss:

$$L_{BCE} = -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\hat{y}_i) + (1 - y_i)log(1 - \hat{y}_i)) \quad (2)$$

and J the Jaccard index adapted for non-discrete objects:

$$J = \frac{1}{n}\sum_{i=1}^{n}\frac{y_i\hat{y}_i}{y_i + \hat{y}_i - y_i\hat{y}_i}, \qquad (3)$$

---

[1] A PyTorch implementation is available at <https://github.com/laibe/M2U-Net>

n denotes the number of pixels in a given image, the network's prediction probability output of the pixel belonging to the vessel class is denoted by $\hat{y}$ and the ground-truth by y.

For the DRIVE dataset we adopt the training-test split as proposed by the authors of the dataset (20 training and 20 test images) [43]. For CHASE_DB1 we follow the suggestion of [18], where the first 8 images form the training set and the remaining 20 the test set. This split is also used in [33].

For HRF we adopt the split as proposed by Orlando *et al.* [33] and adapted in [45], who introduced the only supervised methods that have so far been evaluated on this dataset. The training set contains the first five images of each category (healthy, diabetic retinopathy and glaucoma); the remaining 30 images define the test set.

Examples of training images of DRIVE, CHASE_DB1 and HRF together with their resolution are shown in Figure 3. DRIVE and CHASE_DB1 images contain large black borders that do not contain any valuable information. To de-
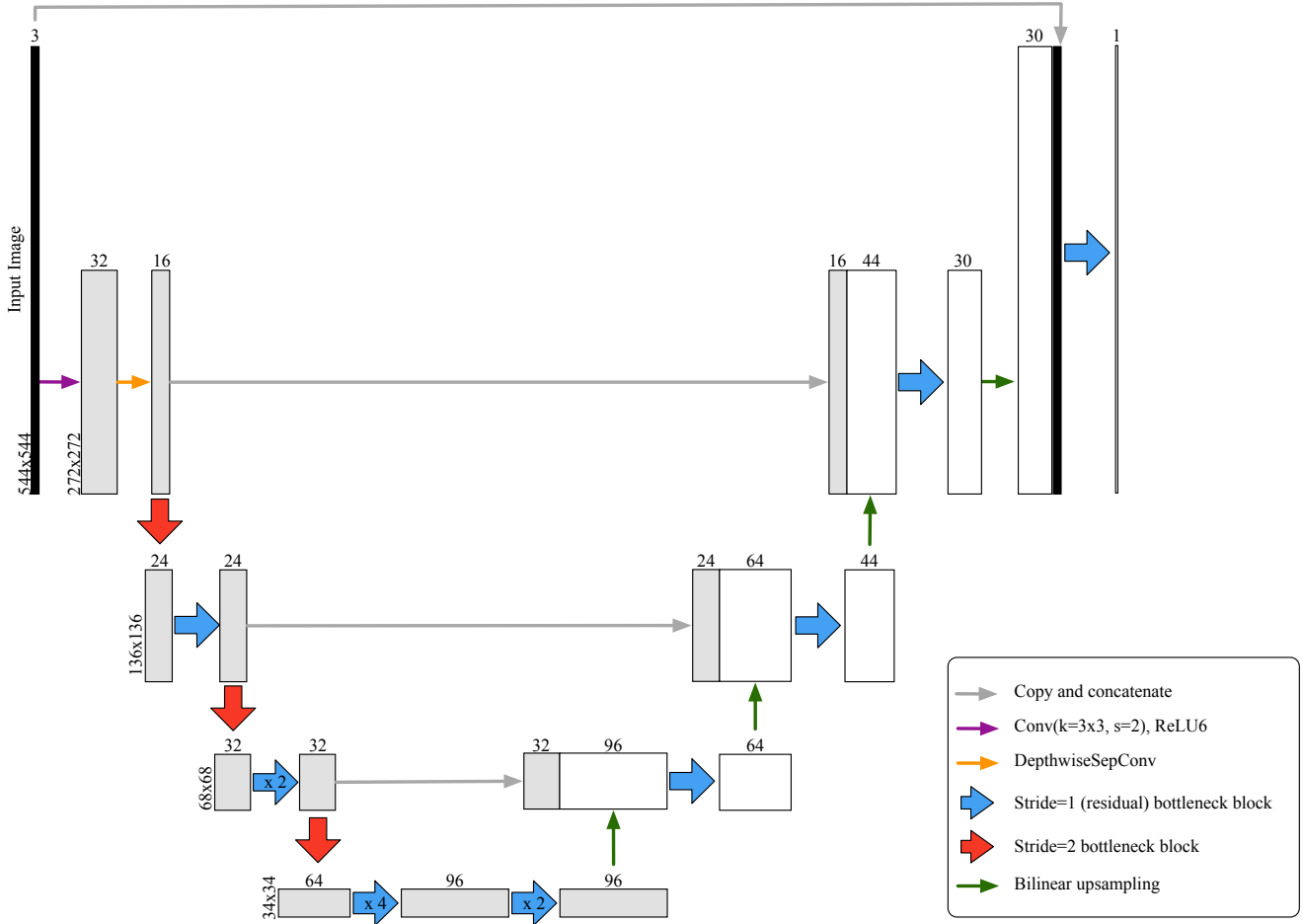
Figure 2: M2U-Net: Example for an input image of dimension $3 \times 544 \times 544$. Each white and grey rectangular box represents a multi-channel feature map. For illustration purposes the height corresponds to the resolution and the width to the number of channels. The encoder part feature maps are highlighted in grey. Best viewed in color.

crease the number of background pixels and to ensure that the input resolution is a multiple of 16, a requirement for M2U-Net, U-Net [37] and ERFNet [36], we perform the following crops:

- On DRIVE we take a 544×544 center crop.

- On CHASE_DB1 we crop 18 pixels on the left and 21 on the right, resulting in a resolution of 960×960.

- On HRF we do not perform cropping as the original resolution of 2336×3504 is already a multiple of 16.

No other preprocessing is conducted and during training, a set of random augmentations are applied: rotations, horizontal and vertical flips, elastic distortions and changes in brightness, contrast, saturation and hue. Image augmentations are commonly used in biomedical image analysis tasks, especially when working with small datasets, as they

can improve accuracy and generalization [14]. We limit the rotations to a range from -15 to $+15$ degrees. Brightness, contrast, saturation and hue are changed by a small random factor in the range $\mathbb{R} \cap [1 - c, 1 + c]$, where c determines the freedom of change. The following values are selected: $c_{brightness} = 0.3$, $c_{contrast} = 0.3$, $c_{saturation} = 0.02$ and $c_{hue} = 0.02$. Elastic transformations are governed by the grid size and the magnitude, for which we selected values of 8×8 and 1 respectively [10].

Since our datasets contain a very small number of training images we derive separate validation sets from the training sets, using the same random augmentations. During training, the model with the highest dice score on the validation set is selected. Training is stopped after 300 epochs.

We compare the performance of M2U-Net versus the performance of ERFNet [36] and U-Net [37] that are trained with identical training parameters. ERFNet is a recent ef-

Figure 3: Examples of training images ($H \times W$). **Top left**: DRIVE (584×565); **top right**: CHASE_DB1 (960×999); **bottom**: HRF (2336×3504).

ficient real-time semantic segmentation method, originally applied on urban scene segmentation tasks. We deviate from the original U-Net implementation by using zero-padding to ensure the resolution of the final feature map is equal to the resolution of the input image. It should also be noted that in contrast to ERFNet and M2U-Net that utilize pretrained encoders, the U-Net was trained from scratch and acts as a baseline only.

On DRIVE we use a batch-size of 4 for ERFNet and M2U-Net and a batch size of 2 for U-Net due to it's higher memory footprint. On CHASE_DB1 we use a batch-size of 2 for ERFNet and M2U-Net. The U-Net can not be trained on CHASE_DB1 and HRF without reverting to a patch-based training approach, due to insufficient GPU memory in our training setup (11 GB). Similarly the memory requirement of ERFNet for HRF is too large for our setup. As a consequence we train only M2U-Net on HRF, using a batch-size of 1.

4.2. Evaluation metrics

Since in our segmentation task the number of non-vessel pixels is far larger than the number of vessel pixels, accuracy is not a suitable evaluation measurement. Instead we follow Maninis *et al*. [29] and plot precision (Pr) and recall

(Re) at various thresholds of the probability map output and compute the Dice score (also referred to as F1-score) as our key summary evaluation criteria. It is commonly expressed in terms of true positives (TP), false negatives (FN) and false positives (FP):

$$Dice = \frac{2 * TP}{2 * TP + FN + FP} = 2\frac{Pr * Re}{Pr + Re} \quad (4)$$

Another common evaluation measurement in semantic segmentation tasks, the Jaccard index, has a monotonically increasing relationship with the Dice score and as a consequence both measures are equivalent for comparison purposes [35]. We report the Dice score, since it is more common in the literature on retinal vessel segmentation. For comparison purposes with methods that do not report Dice scores, we additionally report accuracy and area under the receiver operating characteristic curve (AuC). We adjust the number of true negatives by the number of pixels that were cropped to ensure a fair comparison of accuracy and AuC with previous works.

5. Results

In this section we present the results of M2U-Net on DRIVE, CHASE_DB1 and HRF in comparison to the performance of U-Net, and ERFNet, where possible. All networks were trained with the same training setups as M2U-Net and their inference time evaluated on the same hardware (Rockchip RK3399 SoC) and software (PyTorch [1] with NNPACK [15] backend and TVM [13]). Additionally we discuss the achieved results with regard to previous published literature.

5.1. Experimental measurements

Table 2 shows the model sizes in terms of number of trainable parameters and Multiply-Add operations together with the achieved Dice score. For ERFNet, no ARM TVM inference times are reported since the PyTorch implementation as provided by its authors is incompatible with TVM.

Out of the three evaluated networks, M2U-Net is by far the smallest and computationally lightest network with only 0.55M parameters, a size on disk of just 2.2MB and 1.4B multiply-adds for an input of dimension $3 \times 544 \times 544$, as illustarted in Figure 4. On DRIVE, our PyTorch implementation has an execution time of under 6 seconds on a single Cortex-A72 compared to over a minute for ERFNet and over 2 hours for U-Net. At the same time, the Dice score of M2U-Net on CHASE_DB1 achieves the best Dice score in our experiment. Visualizations of the segmented vessel probability map and thresholded binary map of M2U-Net together with precision vs. recall curves for all three networks are shown in Figure 5.

Model	Params	Size	DRIVE				CHASE_DB1			
			Dice	MAdds	ARM	ARM TVM	Dice	MAdds	ARM	ARM TVM
2nd human annot.	-	-	0.7881	-	-	-	0.7686	-	-	-
U-Net [37]	31.03M	119.0MB	0.7941	246.6B	141min	58.1s	-	-	-	-
ERFNet [36]	2.06M	8.0MB	0.8022	2.8B	69.8s	-	0.7994	51.3B	336s	-
M2U-Net (ours)	**0.55M**	**2.2MB**	**0.8091**	**1.4B**	**5.87s**	**577ms**	**0.8006**	**4.4B**	**23.5s**	1.67s

Table 2: Results on DRIVE with input dimension $3 \times 544 \times 544$ and CHASE_DB1 with input dimension $3 \times 960 \times 960$. **Params**: Number of parameters. **Size**: Size of the weight file on disk. **Dice**: Dice score (= F1-Score). **MAdds**: Number of Multiply-Add operations. **ARM**: Rockchip RK3399; NNPACK single thread Cortex-A72 inference time. **ARM TVM**: Rockchip RK3399; TVM dual Cortex-A72 inference time.

Method	Platform	DRIVE				CHASE_DB1			
		Time	Dice	Acc	AuC	Time	Dice	Acc	AuC
2nd human observer	-	-	0.7881	0.9472	-	-	0.7686	0.9538	-
Unsupervised									
Argüello *et al.* [6]	NVIDIA GTX 680	10ms	-	0.9431	-	-	-	-	-
Azzopardi *et al.* [7]	2GHz CPU	-	-	-	-	25s	-	0.9387	0.9487
Bendaoudi *et al.* [8]	Xilinx Kintex-7 FPGA*	**2ms**	-	0.9218	0.9207	-	-	-	-
Bibiloni *et al.* [9]	Intel Core i5-3340	37ms	0.7521	0.938	-	-	-	-	-
Jiang *et al.* [22]	Intel Core i7 Duo	1.677s	-	0.9588	-	-	-	-	-
Koukounis *et al.* [24]	Spartan 6 FPGA*	52.3ms	-	0.9240	0.9008	-	-	-	-
Xu *et al.* [44]	Samsung Galaxy S5*	118s	-	0.933	0.959	-	-	-	-
Zhang *et al.* [46]	2.7GHz CPU	20s	-	0.9476	0.9636	-	-	0.9452	0.9487
Supervised									
Fraz *et al.* [18]	Intel Core2Duo	100s	0.7929	0.9480	0.9747	-	0.7566	0.9469	0.9712
Fu *et al.* [19]	-	1.3s	-	0.9523	-	-	-	-	-
Li *et al.* [25]	AMD Athlon II X4	70s	-	0.9527	0.9738	70s	-	0.9581	0.9716
Liskowski *et al.* [26]	NVIDIA TITAN	-	-	0.9535	**0.9790**	-	-	-	-
Maninis *et al.* [29]	NVIDIA TITAN-X	85ms	**0.8220**	-	-	-	-	-	-
Marin *et al.* [30]	Intel Core2Duo	90s	-	0.9452	-	-	-	-	-
Orlando *et al.* [33]	Intel Xeon E5-2690	1s	0.7857	-	0.9507	2.7s	0.7332	-	-
Roychowdhury *et al.* [38]	Intel Core i3	3.115s	-	0.9520	0.9620	11.71s	-	0.9530	0.9532
Yan *et al.* [45]	NVIDIA TITAN-Xp	-	0.8183	0.9529	0.9752	-	-	0.9610	**0.9781**
M2U-Net (ours)	NVIDIA GTX 1080 Ti	6ms	0.8091	**0.9630**	0.9714	**7ms**	**0.8006**	**0.9703**	0.9666
M2U-Net (ours)	Rockchip RK3399*	577ms	0.8091	**0.9630**	0.9714	1.67s	**0.8006**	**0.9703**	0.9666

* Embedded/mobile platform

Table 3: Inference time, Dice score, accuracy, AuC and hardware platform of our proposed method and previous works on DRIVE and CHASE_DB1.

5.2. Comparison with previous work

Table 3 lists previous works in retinal vessel segmentation on DRIVE and CHASE_DB1 in terms of reported inference time, hardware platform, Dice score, accuracy, and AuC. On CHASE_DB1, our proposed architecture achieves a new state-of-the-art Dice score of 0.8006 and an inference time of 1.67s on a Rockchip RK3399 SoC and 7ms on a NVIDIA GTX 1080Ti. In contrast, the second fastest method [33] takes 2.7s on a Intel Xeon CPU with a Dice score of 0.7332. On DRIVE our method achieves a Dice score that is 1.3 percentage points lower compared to [29], but has $14\times$ less parameters.

On HRF, M2U-Net achieves new state-of-the-art results in terms of Dice score and inference times as shown in Table 4. Compared to the previous best unsupervised and supervised method we report an improvement in Dice score of 2.36 and 6.02 percentage points respectively. On ARM M2U-Net achieves inference times of 14.7s and on a GPU it achieves inference speed of 19.9ms, making it the first real-
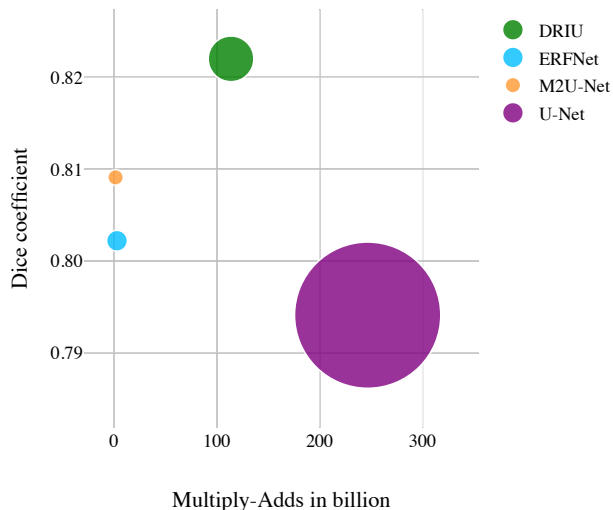
Figure 4: Illustration of model size, performance and computational requirements on DRIVE. The circle sizes are indicative of the number of parameters. DRIU is the model by Maninis *et al.* [29]. Best viewed in color.

Method	Plt.	Time	Dice	Acc
Unsupervised				
Annunziata *et al.* [5]	CPU*	-	0.7578	0.9581
Budai *et al.* [12]	CPU†	26.7s	-	0.9610
Odstrcilik *et al.* [32]	CPU‡	92s	0.7324	0.9494
Zhang *et al.* [46]	-	-	-	0.9556
Supervised				
Orlando *et al.* [33]	CPU¶	5.8s	0.7158	-
Yan *et al.* [45]	GPU§	-	0.7212	0.9437
M2U-Net (ours)	GPU‖	**19.9ms**	**0.7814**	**0.9635**
M2U-Net (ours)	ARM#	14.7s	**0.7814**	**0.9635**

* AMD A4-3300M
† 2.3 GHz
‡ Intel Core i7
¶ Intel Xeon E5-2690
§ NVIDIA TITAN-Xp
‖ NVIDIA GTX 1080 Ti
\# Rockchip RK3399 (TVM)

Table 4: Inference time, Dice score, accuracy and hardware platform of our proposed method and previous works on HRF.

time method on this high-resolution dataset (2336×3504).

The method by [29] might be able to achieve higher quality results on HRF and CHASE_DB1 compared to ours, however due to it's larger size, we expect it to face memory constraints when trained with full resolution images.

The power consumption measured during inference of our method on a Rockchip RK3399 SoC development board is 9.6W with an idle power consumption of 3.6W. This is substantially lower than the power consumption of common server grade GPUs that is often above 200W. The only other work in the reviewed literature that explicitly reports power consumption measurements [24], reports a power consumption of 2.89W and takes 52.3ms on a Spartan 6 FPGA, which is not directly comparable. Optimized FPGA implementations are generally faster and more energy efficient than ARM implementations, and we would expect similar gains for our model on FPGA. The model of [24], however achieves an accuracy of only 0.9240 and an AuC of 0.9008 on DRIVE, compared to an accuracy of 0.9630 and AuC of 0.9714 of our method (no Dice score is reported in [24]).

5.3. Ablation study

In our experiments with various values for t in the decoder block, we found that values smaller than 1 result in similar Dice scores to values greater than 1, while at

the same time substantially reducing the parameter count and memory footprint of the network. We also tried using only depthwise separable convolutions after each upsampling and concatenate operation instead of Stride=1 bottleneck blocks and found slightly worse performance in terms of Dice score while having a higher number of parameters.
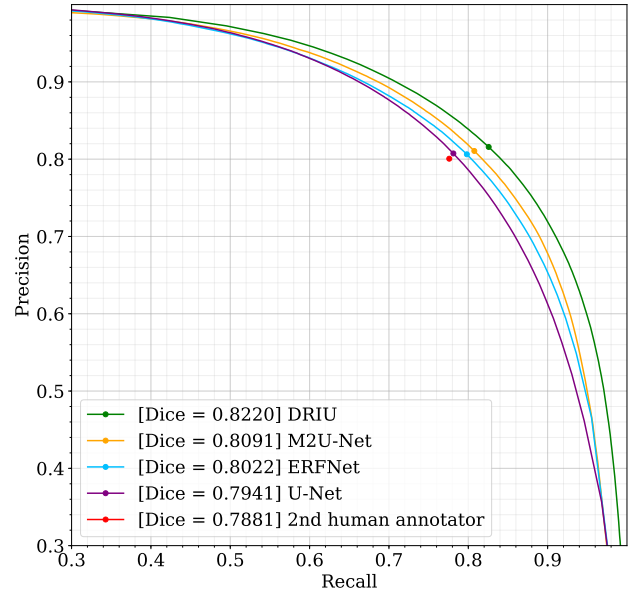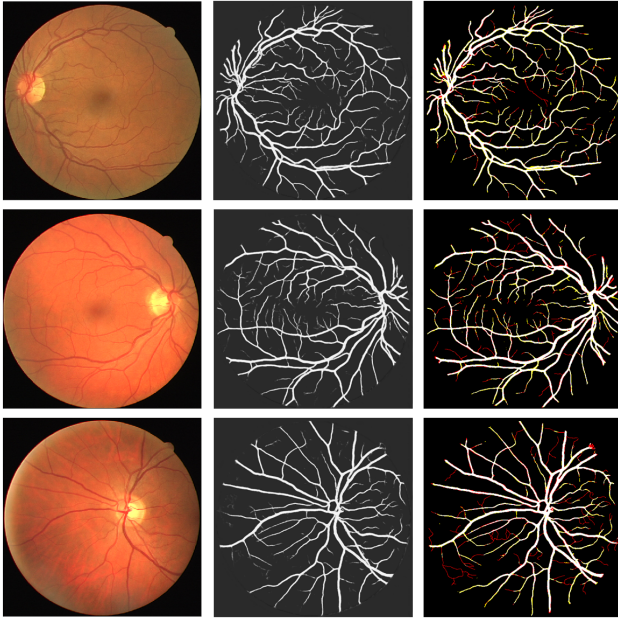
6. Conclusion

We introduced M2U-Net, a new light-weight neural network architecture for the segmentation of retinal vasculature with novel contracting bottleneck blocks in the decoder part and elements of a pretrained MobileNetV2 in the encoder part.

It achieves a significant reduction in the trade-off between hardware requirements, energy demand and execution times for retinal vessel segmentation compared to previous models. Furthermore, it will enable new practical applications, where high quality analysis of high resolution images in real-time is required, e.g. in robotic microsurgery of the eye. We expect that the architecture can also be successfully applied to related tasks, such as road segmentation on satellite images or the semantic segmentation of urban scenes.

Acknowledgements
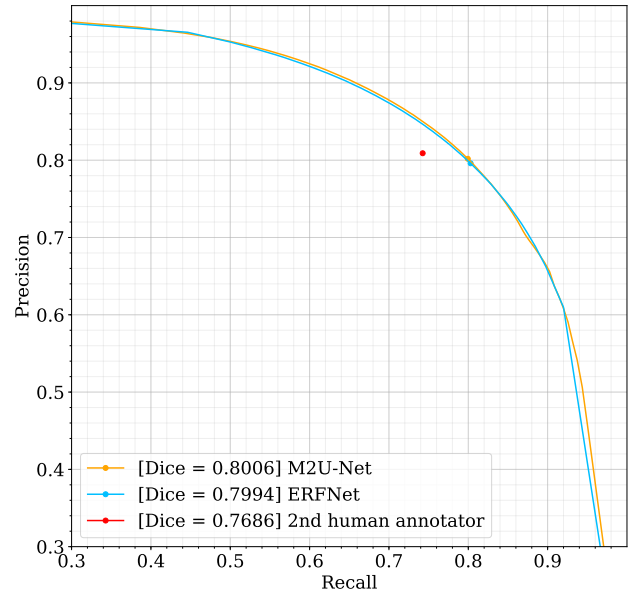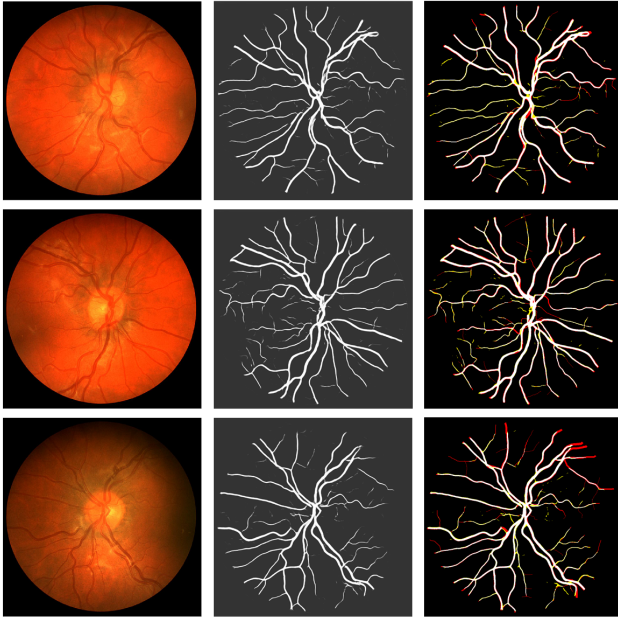
DRIVE

CHASE_DB1

Figure 5: Visualization of results for DRIVE (top) and CHASE_DB1 (bottom). **Left**: first column, fundus test image; second column, vessel probability map output of M2U-Net; third column, thresholded binary map with indication of true positives (white), false positives (yellow) and false negatives (red). **Right**: precision vs. recall curves together with the Dice score at the optimal point along the curve. Best viewed in color.

References

[1] pytorch: Tensors and Dynamic neural networks in Python with strong GPU acceleration, Aug. 2018. original-date: 2016-08-13T05:26:41Z. 5

[2] M. D. Abramoff, M. K. Garvin, and M. Sonka. Retinal Imaging and Image Analysis. *IEEE Reviews in Biomedical Engineering*, 3:169–208, 2010. 1

[3] M. I. Ahmed, M. A. Amin, B. Poon, and H. Yan. Retina based biometric authentication using phase congruency. *International Journal of Machine Learning and Cybernetics*, 5(6):933–945, Dec. 2014. 1

[4] M. U. Akram, A. Tariq, and S. A. Khan. Retinal recognition: Personal identification using blood vessels. In *2011 International Conference for Internet Technology and Secured Transactions*, pages 180–184, Dec. 2011. 1

[5] R. Annunziata, A. Garzelli, L. Ballerini, A. Mecocci, and E. Trucco. Leveraging Multiscale Hessian-Based Enhancement With a Novel Exudate Inpainting Technique for Retinal Vessel Segmentation. *IEEE Journal of Biomedical and Health Informatics*, 20(4):1129–1138, July 2016. 2, 7

[6] F. Argello, D. L. Vilario, D. B. Heras, and A. Nieto. GPU-based segmentation of retinal blood vessels. *Journal of Real-Time Image Processing*, 14(4):773–782, Nov. 2014. 2, 6

[7] G. Azzopardi, N. Strisciuglio, M. Vento, and N. Petkov. Trainable COSFIRE filters for vessel delineation with application to retinal images. *Medical Image Analysis*, 19(1):46–57, Jan. 2015. 6

[8] H. Bendaoudi, F. Cheriet, A. Manraj, H. Ben Tahar, and J. M. P. Langlois. Flexible architectures for retinal blood vessel segmentation in high-resolution fundus images. *Journal of Real-Time Image Processing*, 15(1):31–42, Dec. 2016. 2, 6

[9] P. Bibiloni, M. Gonzlez-Hidalgo, and S. Massanet. A real-time fuzzy morphological algorithm for retinal vessel segmentation. *Journal of Real-Time Image Processing*, Jan. 2018. 2, 6

[10] M. D. Bloice, C. Stocker, and A. Holzinger. Augmentor: An Image Augmentation Library for Machine Learning. *ArXiv e-prints*, Aug. 2017. 4

[11] D. Braun, S. Yang, J. N. Martel, C. N. Riviere, and B. C. Becker. EyeSLAM: Real-time simultaneous localization and mapping of retinal vessels during intraocular microsurgery. *The international journal of medical robotics + computer assisted surgery : MRCAS*, 14(1), Feb. 2018. 1

[12] A. Budai, R. Bock, A. Maier, J. Hornegger, and G. Michelson. Robust Vessel Segmentation in Fundus Images, 2013. 7

[13] T. Chen, T. Moreau, Z. Jiang, L. Zheng, E. Yan, M. Cowan, H. Shen, L. Wang, Y. Hu, L. Ceze, C. Guestrin, and A. Krishnamurthy. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning. In *SysML 2018*, Feb. 2018. 5

[14] T. Ching, D. S. Himmelstein, B. K. Beaulieu-Jones, A. A. Kalinin, B. T. Do, G. P. Way, E. Ferrero, P.-M. Agapow, M. Zietz, M. M. Hoffman, W. Xie, G. L. Rosen, B. J. Lengerich, J. Israeli, J. Lanchantin, S. Woloszynek, A. E. Carpenter, A. Shrikumar, J. Xu, E. M. Cofer, C. A. Lavender, S. C. Turaga, A. M. Alexandari, Z. Lu, D. J. Harris, D. De-Caprio, Y. Qi, A. Kundaje, Y. Peng, L. K. Wiley, M. H. S. Segler, S. M. Boca, S. J. Swamidass, A. Huang, A. Gitter, and C. S. Greene. Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society Interface*, 15(141), Apr. 2018. 4

[15] M. Dukhan. Acceleration package for neural networks on multi-core CPUs: Maratyszcza/NNPACK, Oct. 2018. original-date: 2016-03-21T17:47:34Z. 5

[16] J. D. Fauw, J. R. Ledsam, B. Romera-Paredes, S. Nikolov, N. Tomasev, S. Blackwell, H. Askham, X. Glorot, B. ODonoghue, D. Visentin, G. v. d. Driessche, B. Lakshminarayanan, C. Meyer, F. Mackinder, S. Bouton, K. Ayoub, R. Chopra, D. King, A. Karthikesalingam, C. O. Hughes, R. Raine, J. Hughes, D. A. Sim, C. Egan, A. Tufail, H. Montgomery, D. Hassabis, G. Rees, T. Back, P. T. Khaw, M. Suleyman, J. Cornebise, P. A. Keane, and O. Ronneberger. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature Medicine*, page 1, Aug. 2018. 3

[17] I. N. Figueiredo, S. Moura, J. S. Neves, L. Pinto, S. Kumar, C. M. Oliveira, and J. D. Ramos. Automated retina identification based on multiscale elastic registration. *Computers in Biology and Medicine*, 79:130–143, Dec. 2016. 1

[18] M. M. Fraz, P. Remagnino, A. Hoppe, B. Uyyanonvara, A. R. Rudnicka, C. G. Owen, and S. A. Barman. An Ensemble Classification-Based Approach Applied to Retinal Blood Vessel Segmentation. *IEEE Transactions on Biomedical Engineering*, 59(9):2538–2548, Sept. 2012. 2, 3, 6

[19] H. Fu, Y. Xu, S. Lin, D. W. Kee Wong, and J. Liu. DeepVessel: Retinal Vessel Segmentation via Deep Learning and Conditional Random Field. In S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, editors, *Medical Image Computing and Computer-Assisted Intervention MICCAI 2016*, pages 132–139, Cham, 2016. Springer International Publishing. 6

[20] L. D. Hubbard, R. J. Brothers, W. N. King, L. X. Clegg, R. Klein, L. S. Cooper, A. R. Sharrett, M. D. Davis, and J. Cai. Methods for evaluation of retinal microvascular abnormalities associated with hypertension/sclerosis in the atherosclerosis risk in communities study11the authors have no proprietary interest in the equipment and techniques described in this article. *Ophthalmology*, 106(12):2269 – 2280, 1999. 1

[21] V. Iglovikov, S. Seferbekov, A. Buslaev, and A. Shvets. TernausNetV2: Fully Convolutional Network for Instance Segmentation. pages 233–237, 2018. 3

[22] Z. Jiang, J. Yepez, S. An, and S. Ko. Fast accurate and robust retinal vessel segmentation system. *Biocybernetics and Biomedical Engineering*, 37(3):412–421, Jan. 2017. 6

[23] T. Kohler, A. Budai, M. F. Kraus, J. Odstrcilik, G. Michelson, and J. Hornegger. Automatic no-reference quality

assessment for retinal fundus images using vessel segmentation. In *2013 IEEE 26th International Symposium on Computer-Based Medical Systems (CBMS)*, volume 00, pages 95–100, June 2013. 2

[24] D. Koukounis, C. Ttofis, A. Papadopoulos, and T. Theocharides. A high performance hardware architecture for portable, low-power retinal vessel segmentation. *Integration, the VLSI Journal*, 47(3):377–386, June 2014. 2, 6, 7

[25] Q. Li, B. Feng, L. Xie, P. Liang, H. Zhang, and T. Wang. A Cross-Modality Learning Approach for Vessel Segmentation in Retinal Images. *IEEE transactions on medical imaging*, 35(1):109–118, Jan. 2016. 6

[26] P. Liskowski and K. Krawiec. Segmenting Retinal Blood Vessels With Deep Neural Networks. *IEEE Transactions on Medical Imaging*, 35(11):2369–2380, Nov. 2016. 2, 6

[27] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, June 2015. 2

[28] I. Loshchilov and F. Hutter. *Fixing Weight Decay Regularization in Adam*. 2018. 3

[29] K. K. Maninis, J. Pont-Tuset, P. Arbelez, and L. V. Gool. Deep Retinal Image Understanding. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2016. 2, 5, 6, 7

[30] D. Marin, A. Aquino, M. E. Gegundez-Arias, and J. M. Bravo. 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, Jan. 2011. 6

[31] C. Mario, M. G. Penedo, M. Penas, M. J. Carreira, and F. Gonzalez. Personal authentication using digital retinal images. *Pattern Analysis and Applications*, 9(1):21–33, May 2006. 1

[32] J. Odstrcilik, R. Kolar, A. Budai, J. Hornegger, J. Jan, J. Gazarek, T. Kubena, P. Cernosek, O. Svoboda, and E. Angelopoulou. Retinal vessel segmentation by improved matched filtering: evaluation on a new high-resolution fundus image database. *IET Image Processing; Stevenage*, 7(4):373–383, June 2013. 7

[33] J. I. Orlando, E. Prokofyeva, and M. B. Blaschko. 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, Jan. 2017. 3, 6, 7

[34] M. Ortega, M. G. Penedo, J. Rouco, N. Barreira, and M. J. Carreira. Personal verification based on extraction and characterisation of retinal feature points. *Journal of Visual Languages & Computing*, 20(2):80–90, Apr. 2009. 1

[35] J. Pont-Tuset and F. Marques. Supervised Evaluation of Image Segmentation and Object Proposal Techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7):1465–1478, July 2016. 5

[36] E. Romera, J. M. lvarez, L. M. Bergasa, and R. Arroyo. ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation. *IEEE Transactions on Intel-*

ligent Transportation Systems, 19(1):263–272, Jan. 2018. 2, 4, 6

[37] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. 1, 2, 4, 6

[38] S. Roychowdhury, D. D. Koozekanani, and K. K. Parhi. Blood Vessel Segmentation of Fundus Images by Major Vessel Extraction and Subimage Classification. *IEEE Journal of Biomedical and Health Informatics*, 19(3):1118–1128, May 2015. 6

[39] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation. *CoRR*, abs/1801.04381, 2018. 1, 2

[40] L. Sifre. *Rigid-motion scattering for image classification*. Ph. D. thesis, 2014. 2

[41] C. Simon and I. Goldstein. A new scientific method of identification. *New York State Journal of Medicine*, 35(18):901–906, Sept. 1935. 1

[42] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556, Sept. 2014. 2

[43] J. Staal, M. D. Abramoff, M. Niemeijer, M. A. Viergever, and B. v. Ginneken. Ridge-based vessel segmentation in color images of the retina. *IEEE Transactions on Medical Imaging*, 23(4):501–509, Apr. 2004. 2, 3

[44] X. Xu, W. Ding, X. Wang, R. Cao, M. Zhang, P. Lv, and F. Xu. Smartphone-Based Accurate Analysis of Retinal Vasculature towards Point-of-Care Diagnostics. *Scientific Reports*, 6:34603, Oct. 2016. 2, 6

[45] Z. Yan, X. Yang, and K. Cheng. Joint Segment-Level and Pixel-Wise Losses for Deep Learning Based Retinal Vessel Segmentation. *IEEE Transactions on Biomedical Engineering*, 65(9):1912–1923, Sept. 2018. 2, 3, 6, 7

[46] J. Zhang, B. Dashtbozorg, E. Bekkers, J. P. W. Pluim, R. Duits, and B. M. t. H. Romeny. Robust Retinal Vessel Segmentation via Locally Adaptive Derivative Frames in Orientation Scores. *IEEE Transactions on Medical Imaging*, 35(12):2631–2644, Dec. 2016. 6, 7