Appendices of the paper *"On the Relevance of Sparsity for Image Classification"*

Roberto Rigamonti[a], Vincent Lepetit[a], Germán González[b], Engin Türetken[a], Fethallah Benmansour[a], Matthew Brown[c], Pascal Fua[a]

[a]*Computer Vision Laboratory, École Polytechnique Fédérale de Lausanne (EPFL), EPFL/IC/ISIM/CVLab, Station 14, CH-1015 Lausanne, Switzerland.*
[b]*Massachusetts Institute of Technology, 77 Massachusetts Ave, Cambridge, MA 02139-4307, USA*
[c]*Department of Computer Science, University of Bath, BA2 7AY, UK.*

1. Image categorization

The results presented in this section are encoded according to the naming convention presented in the paper, that is, by using capital letters to represent the component names. For example, *OLS-SPARSEIT-ABS-MAX-PCA-SVM* represents the pipeline where we first extract sparse features computed by Iterative Thresholding using filters obtained with the Olshausen and Field's algorithm, then we use the absolute value function for rectification, use a max-pooling operation, project the result into an eigenspace, and finally use a Support Vector Machine for classification. We sometimes use a star (*) as the name of one component that is varied for an evaluation.

1.1. Filter learning

The filter learning procedure is based on Stochastic Gradient Descent with Iterative Thresholding [1]. The objective function of Eq. (3) in the

Email addresses: `roberto.rigamonti@epfl.ch` (Roberto Rigamonti), `vincent.lepetit@epfl.ch` (Vincent Lepetit), `ggonzale@mit.edu` (Germán González), `engin.turetken@epfl.ch` (Engin Türetken), `fethallah.benmansour@epfl.ch` (Fethallah Benmansour), `m.brown@bath.ac.uk` (Matthew Brown), `pascal.fua@epfl.ch` (Pascal Fua)

paper is minimized by alternating optimizations over the filters $\mathbf{f}^j$ and the coefficients $\mathbf{t}_i^j$. Table 1 reports the recognition rates when the filters learned with two extreme values for λ_{learn}, $\lambda_{learn} = 0$ and $\lambda_{learn} = 7$, are used. For both the CIFAR-10 and the Caltech-101 datasets we have found that the most effective gradient step for the feature maps is $\eta_{coeffs} = 10^{-1}$, and the one on the filters is $\eta_{filters} = 10^{-5}$. Unless otherwise stated, the results for the CIFAR-10 dataset refer to filter banks composed by 49 11×11 filters learned with $\lambda_{learn} = 2$. The filter bank used for the comparisons with [2], on the other hand, is composed by 25 11×11 filters learned with $\lambda_{learn} = 0.02$, the same value used to learn the 64 9×9 filters used in the Caltech-101 experiment.

In the tables below we present the ℓ_0 norm of the feature maps both after the feature extraction ($\|\mathbf{t}\|_0$) and after the pooling ($\|\mathbf{v}\|_0$) steps (when relevant). Also, throughout the text, the best-scoring configurations are marked in bold.

1.2. Sparsification

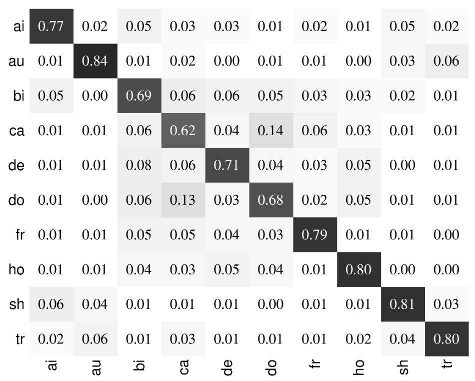
When requested, we have sparsified the output of the convolutions by using Iterative Thresholding. We have chosen to adopt fixed gradient descent step, and we empirically found $\eta_{coeffs} = 210^{-3}$ was giving good results. We analyzed the monotonic convergence of the optimization scheme by plotting the reconstruction error, the ℓ_0 and ℓ_1 norms, and the corresponding functional value for each $\lambda_{extract}$ choice. We then opted for a termination criterion based on the difference between the functional value in two subsequent iterations, and we have set this threshold to 10^{-5}.

1.3. Pooling

For the three different pooling strategies we have considered (*GAUSS*, *BOXCAR*, and *MAX*), we have performed different experiment to devise the best parametrizations. For the CIFAR-10 dataset resized to 16×16 pixels the best results were achieved with a $4\times$ downscaling, and setting the variance of the 9×9 Gaussian filter in the *GAUSS* case to either $\sigma = 2$ or $\sigma = 3$. The experiments on the Caltech-101 dataset used instead a boxcar filter with size 10×10, followed with a $5\times$ downscaling to match the procedure adopted in [3].

Table 1: Comparison of the recognition rates for the filters learned with different λ_{learn} values. These results are obtained on the CIFAR-10 dataset with *OLS-*-POSNEG-GAUSS-PCA-SVM*. For a large range of $\lambda_{extract}$ and for a reasonable value for λ_{learn} ($\lambda_{learn} = 2$), the recognition rate is high. For more extreme values for λ_{learn} the performances tend to be worse than the ones obtained with random filters, as shown in Tab. 4

Method	$\lambda_{extract}$	$\lambda_{learn} = \mathbf{0}$			$\lambda_{learn} = \mathbf{2}$			$\lambda_{learn} = \mathbf{7}$		
		$\Vert\mathbf{t}\Vert_0$	$\Vert\mathbf{v}\Vert_0$	**R. Rate [%]**	$\Vert\mathbf{t}\Vert_0$	$\Vert\mathbf{v}\Vert_0$	**R. Rate [%]**	$\Vert\mathbf{t}\Vert_0$	$\Vert\mathbf{v}\Vert_0$	**R. Rate [%]**
CONV		1.000	1.000	55.08	1.000	1.000	67.16	1.000	0.999	54.59
SPARSEIT	1e-4	0.869	1.000	55.91	0.838	1.000	67.28	0.965	0.999	54.11
SPARSEIT	2e-4	0.850	1.000	54.62	0.787	1.000	67.34	0.956	0.999	54.29
SPARSEIT	3e-4	0.829	1.000	56.62	**0.741**	**1.000**	**67.39**	0.945	0.999	54.39
SPARSEIT	4e-4	0.806	1.000	56.55	0.701	1.000	67.30	0.934	0.999	54.28
SPARSEIT	5e-4	0.782	1.000	56.64	0.665	1.000	67.24	0.922	0.999	54.36
SPARSEIT	6e-4	0.757	1.000	56.78	0.633	1.000	67.26	0.911	0.999	54.32
SPARSEIT	7e-4	**0.731**	**1.000**	**56.83**	0.604	1.000	67.26	0.899	0.999	54.20
SPARSEIT	8e-4	0.705	1.000	56.77	0.578	1.000	67.18	0.887	0.999	54.46
SPARSEIT	9e-4	**0.679**	**1.000**	**56.83**	0.555	1.000	67.24	0.876	0.999	54.37
SPARSEIT	1e-3	0.652	1.000	56.82	0.534	1.000	67.11	0.865	0.999	54.36
SPARSEIT	2e-3	0.373	0.996	55.13	0.387	0.998	66.52	0.759	0.998	54.08
SPARSEIT	3e-3	0.182	0.973	54.89	0.287	0.993	65.70	0.642	0.997	53.88
SPARSEIT	4e-3	0.123	0.940	54.48	0.173	0.965	61.69	0.293	0.987	54.11
SPARSEIT	5e-3	0.102	0.912	53.81	0.075	0.883	54.81	**0.173**	**0.978**	**55.21**
SPARSEIT	6e-3	0.088	0.883	53.49	0.060	0.850	53.30	0.150	0.972	55.20
SPARSEIT	7e-3	0.078	0.855	52.38	0.055	0.825	52.55	0.137	0.967	55.00
SPARSEIT	8e-3	0.069	0.827	49.74	0.050	0.802	51.78	0.126	0.963	54.61
SPARSEIT	9e-3	0.062	0.800	51.84	0.046	0.780	51.29	0.117	0.957	54.90
SPARSEIT	1e-2	0.057	0.773	49.12	0.042	0.759	50.28	0.109	0.952	53.55



	ai	au	bi	ca	de	do	fr	ho	sh	tr
ai	0.77	0.02	0.05	0.03	0.03	0.01	0.02	0.01	0.05	0.02
au	0.01	0.84	0.01	0.02	0.00	0.01	0.01	0.00	0.03	0.06
bi	0.05	0.00	0.69	0.06	0.06	0.05	0.03	0.03	0.02	0.01
ca	0.01	0.01	0.06	0.62	0.04	0.14	0.06	0.03	0.01	0.01
de	0.01	0.01	0.08	0.06	0.71	0.04	0.03	0.05	0.00	0.01
do	0.01	0.00	0.06	0.13	0.03	0.68	0.02	0.05	0.01	0.01
fr	0.01	0.01	0.05	0.05	0.04	0.03	0.79	0.01	0.01	0.00
ho	0.01	0.01	0.04	0.03	0.05	0.04	0.01	0.80	0.00	0.00
sh	0.06	0.04	0.01	0.01	0.01	0.00	0.01	0.01	0.81	0.03
tr	0.02	0.06	0.01	0.03	0.01	0.01	0.01	0.02	0.04	0.80

Figure 1: Confusion matrix for our best scoring configuration, *OLS-CONV-POSNEG-GAUSS-LDE-SVM*, when applied on 32×32 pixels grayscale images from the CIFAR-10 dataset. The abbreviations used for the class names are explained in Tab. 2. The final recognition rate in this experiment is 75.28%.

1.4. Classification step setup

1.4.1. CIFAR-10

We identify two configurations for the *SVM* that prove to suit our needs:

- **Fast classification setup.** We use the LIBSVM library [1], and we perform a C-Support Vector Classification (C-SVC) by using Radial Basis Functions as kernels, and setting the γ parameter to 10. This is a useful configuration to explore a large parameter space, since classification requires between 1 and 2 hours on a modern laptop.

- **Best performing setup.** We achieve our best results on the CIFAR-10 dataset by solving an expensive multi-class bound-constrained SVC problem with $\gamma = 8$ using the BSVM implementation bundled within the libHIK library [2]. We are unable to systematically adopt this classification scheme since each experiment on our computers requires up to a day. The confusion matrix corresponding to the best result is reported in Fig. 1. The adopted abbreviations for the class names in the CIFAR-10 dataset are reported in Tab. 2.

[1]`http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[2]`http://www.cc.gatech.edu/cpl/projects/libHIK`.

Table 2: Class abbreviations adopted for the CIFAR-10 dataet.

Class	airplane	automobile	bird	cat	deer
Abbreviation	ai	au	bi	ca	de
Class	dog	frog	horse	ship	truck
Abbreviation	do	fr	ho	sh	tr

For the comparison with the approach presented in [2] we have used the classifier provided with their code, that is, a linear SVM classifier, and with approximate Nearest-Neighbor classification.

1.4.2. Caltech-101

We have used the logistic regression classifier accompanying [3]. In particular, we have set the learning rate to $\eta = 0.25$, the decay rate to 10^{-4}, and we have iterated for 200 epochs.

2. Extensive evaluation of pipeline components

We perform a thorough evaluation aimed at establishing the relative importance of pipeline's components, in order to properly tune our image categorization system.

Since extensive experimentations on the Caltech-101 dataset are prohibitively expensive owing to the resolution of the images, the tests reported below are performed on the CIFAR-10 dataset where the images are resized to 16×16 pixels.

Table 3 compares the recognition rates for different non-linearities. The presence of a non-linearity step is important for getting good results [3]. *POSNEG* outperforms *ABS* with both learned and handcrafted filter banks, even though the latter is the one traditionally used in state-of-the-art systems. Despite this result, in the experiments with the Caltech-101 dataset we adopt the *ABS* nonlinearity, both to be consistent with the architecture proposed in [3] and because *POSNEG* has the drawback of doubling the descriptor's size.

To assess the importance of a learned filter bank compared to using a random or an handcrafted filter bank, we perform thorough experimentations adopting different filter banks in the feature extraction step of our pipeline. Table 4 shows the recognition rate when 49 randomly generated filters are

Table 3: Comparison between non-linearities for both learned and handcrafted filter banks. The pipeline configuration is **-CONV-*-GAUSS-PCA-SVM*. *POSNEG* outperforms *ABS* with both learned and handcrafted filter banks

Method	**Rec. Rate [%]**	
	OLS	*LM*
POSNEG	**67.16**	**66.18**
ABS	63.17	62.83
NONE	35.61	37.01

employed, while Tab. 5 compares in the same way learned and handcrafted filter banks, both using SVMs and approximate-NN classification schemes.

Even though the results presented in Tab. 5 seem to indicate that only little advantage is gained by learning the filter bank compared to using an handcrafted one, the results shown in Table 6 demonstrate that if one takes the “off-the-shelf” Leung-Malik filter bank [4] (that is, it does not alter it by a whitening step), the recognition rate drops below the one achieved by the randomly generated filter bank.

3. Pixel classification

Figure 2 presents the ROC curves corresponding to the Precision/Recall curves presented in the paper. In particular, Figs. 2(a-c) show the superiority of learned filters with competing handcrafted approaches, while Fig. 2(d) compares the sparsified descriptors obtained with different values of the λ_{segm} parameter with those resulting from plain convolution.

Figure 3 depicts the Precision/Recall and the ROC curves for two other randomly chosen images from the DRIVE dataset, namely image 12 and image 13. Again, learned filters outperform the other approaches and they get very close to the human performance. Analytic measures of the performance of the filter banks for the given experiments are reported in Tab. 7.

In Fig. 4 and Fig. 5 visual results on the DRIVE dataset are shown, using the convention that True Positive pixels are colored in red, False Positives in green, and False Negatives in blue. The results in the second column are computed with a False Positive Rate fixed at 0.05, while those in the third column with a True Positive Rate of 0.9. Even from a visual stance, the results obtained with learned filters (Fig. 5(g-i)) are markedly more appealing

Table 4: Comparison of the recognition rates achieved with different $\lambda_{extract}$ values when random filters are used in the extraction stage *RND-*-POSNEG-GAUSS-PCA-SVM*. Random filters work surprisingly well, but not as well as learned ones

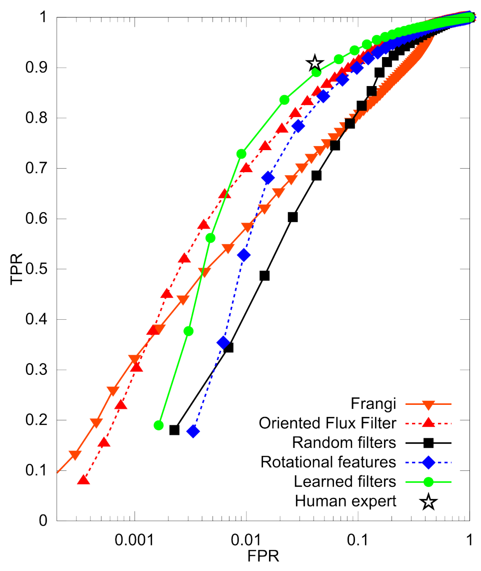
Method	$\lambda_{extract}$	$\Vert\mathbf{t}\Vert_0$	$\Vert\mathbf{v}\Vert_0$	**Rec. Rate** [%]
CONV		1.000	0.999	58.13
SPARSEIT	1e-4	0.811	0.999	59.40
SPARSEIT	2e-4	0.802	0.999	59.39
SPARSEIT	3e-4	0.794	0.999	59.39
SPARSEIT	4e-4	0.786	0.999	58.97
SPARSEIT	5e-4	0.778	0.999	59.45
SPARSEIT	6e-4	0.769	0.999	57.28
SPARSEIT	7e-4	0.761	0.999	58.95
SPARSEIT	8e-4	0.752	0.999	58.92
SPARSEIT	9e-4	0.744	0.999	59.55
SPARSEIT	1e-3	0.736	0.999	59.71
SPARSEIT	2e-3	0.654	0.999	58.88
SPARSEIT	3e-3	0.579	0.999	58.26
SPARSEIT	**4e-3**	**0.507**	**0.999**	**60.17**
SPARSEIT	5e-3	0.434	0.998	57.18
SPARSEIT	6e-3	0.353	0.993	59.65
SPARSEIT	7e-3	0.251	0.970	57.70
SPARSEIT	8e-3	0.130	0.887	52.61
SPARSEIT	9e-3	0.058	0.787	53.02
SPARSEIT	1e-2	0.044	0.747	51.98

Table 5: Comparison of the recognition rates between learned (*OLS*) and Leung-Malik (*LM*) filters [4] with whitening, for convolution and different values of $\lambda_{extract}$ and for different classification methods (*SVM* and *NN*) with **-*-POSNEG-GAUSS-PCA-**. Using sparsity, we can automatically learn filters that perform at least as good as carefully designed ones

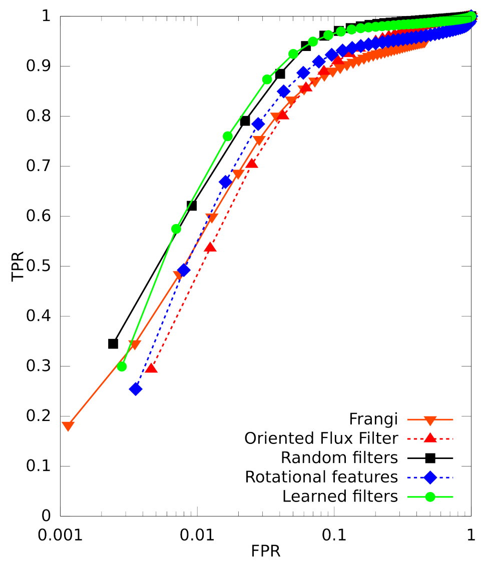
Method	$\lambda_{extract}$	*OLS*				*LM*			
				Rec. Rate [%]				**Rec. Rate [%]**	
		$\\|\mathbf{t}\\|_0$	$\\|\mathbf{v}\\|_0$	*SVM*	*NN*	$\\|\mathbf{t}\\|_0$	$\\|\mathbf{v}\\|_0$	*SVM*	*NN*
CONV		1.000	1.000	67.16	44.07	1.000	1.000	66.18	42.69
SPARSEIT	1e-4	0.838	1.000	67.28	45.48	0.569	0.917	**66.45**	42.99
SPARSEIT	2e-4	0.787	1.000	67.34	45.86	0.505	0.907	66.37	43.71
SPARSEIT	3e-4	0.741	1.000	**67.39**	45.53	0.458	0.900	66.31	43.46
SPARSEIT	4e-4	0.701	1.000	67.30	45.15	0.418	0.893	66.37	43.65
SPARSEIT	5e-4	0.665	1.000	67.24	45.53	0.382	0.885	65.99	44.21
SPARSEIT	6e-4	0.633	1.000	67.26	45.75	0.349	0.878	66.18	44.13
SPARSEIT	7e-4	0.604	1.000	67.26	45.21	0.320	0.870	65.90	43.66
SPARSEIT	8e-4	0.578	1.000	67.18	45.98	0.294	0.862	65.89	43.82
SPARSEIT	9e-4	0.555	1.000	67.24	45.95	0.275	0.855	66.08	44.58
SPARSEIT	1e-3	0.534	1.000	67.11	45.25	0.259	0.849	65.98	**44.68**
SPARSEIT	2e-3	0.387	0.998	66.52	45.59	0.169	0.796	64.93	44.66
SPARSEIT	3e-3	0.287	0.993	65.70	**45.99**	0.124	0.745	63.83	43.89
SPARSEIT	4e-3	0.173	0.965	61.69	44.97	0.097	0.698	62.56	43.65
SPARSEIT	5e-3	0.075	0.883	54.81	44.56	0.078	0.654	61.06	43.99
SPARSEIT	6e-3	0.060	0.850	53.30	44.15	0.064	0.613	59.42	43.97
SPARSEIT	7e-3	0.055	0.825	52.55	44.02	0.054	0.574	58.06	42.73
SPARSEIT	8e-3	0.050	0.802	51.78	43.89	0.046	0.538	56.40	42.35
SPARSEIT	9e-3	0.046	0.780	51.29	43.13	0.040	0.504	54.66	42.87
SPARSEIT	1e-2	0.042	0.759	50.28	43.13	0.034	0.472	53.34	41.55

Table 6: Comparison of the recognition rates achieved with different $\lambda_{extract}$ values when non-whitened Leung-Malik filters [4] are used (*LM-*-POSNEG-GAUSS-PCA-**). The performances are much worse than the ones obtained with the whitened filters

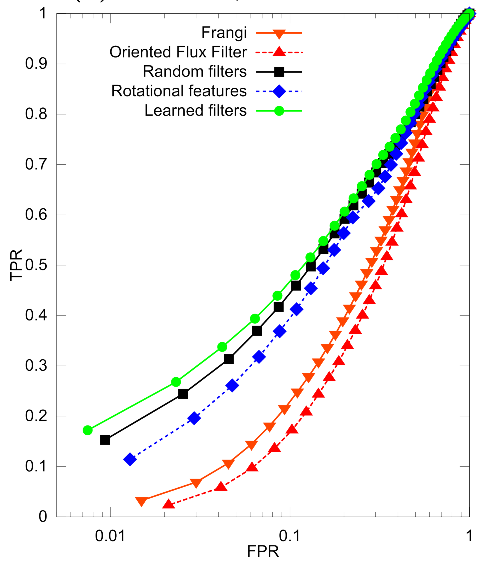
Method	$\lambda_{extract}$	$\|\mathbf{t}\|_0$	$\|\mathbf{v}\|_0$	**Rec. Rate [%]**	
				SVM	*NN*
CONV		0.999	0.997	**55.52**	**37.03**
SPARSEIT	1e-4	0.610	0.840	50.54	30.85
SPARSEIT	2e-4	0.595	0.838	51.61	30.93
SPARSEIT	3e-4	0.583	0.837	51.33	30.78
SPARSEIT	4e-4	0.572	0.836	51.43	31.01
SPARSEIT	5e-4	0.562	0.834	52.12	30.79
SPARSEIT	6e-4	0.554	0.833	52.17	30.48
SPARSEIT	7e-4	0.545	0.832	51.85	30.87
SPARSEIT	8e-4	0.538	0.831	52.26	30.72
SPARSEIT	9e-4	0.531	0.831	52.16	30.99
SPARSEIT	1e-3	0.524	0.830	52.37	30.81
SPARSEIT	2e-3	0.474	0.823	52.02	30.85
SPARSEIT	3e-3	0.438	0.816	52.92	30.87
SPARSEIT	4e-3	0.410	0.809	52.88	31.36
SPARSEIT	5e-3	0.386	0.801	53.79	31.33
SPARSEIT	6e-3	0.365	0.792	53.73	30.99
SPARSEIT	7e-3	0.347	0.783	53.63	31.12
SPARSEIT	8e-3	0.331	0.774	53.58	30.93
SPARSEIT	9e-3	0.316	0.763	53.75	31.50
SPARSEIT	1e-2	0.302	0.753	53.39	31.92



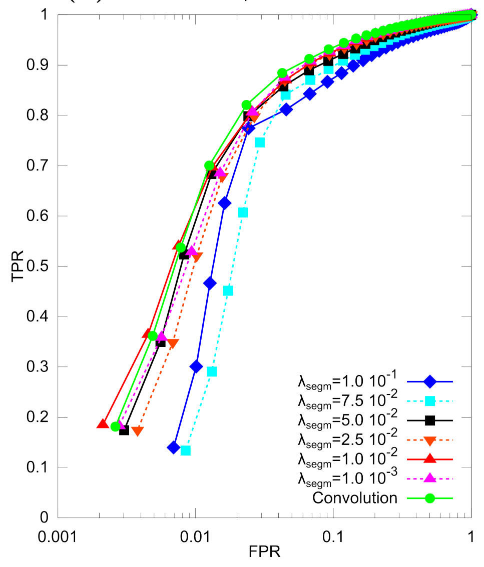
(a) DRIVE, ROC curve



(b) Neurons, ROC curve



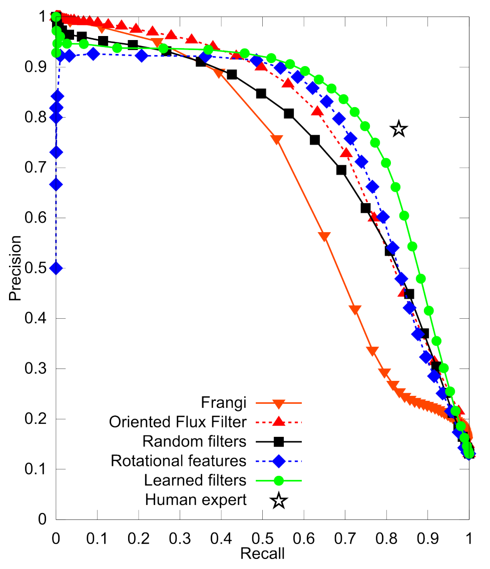
(c) Roads, ROC curve



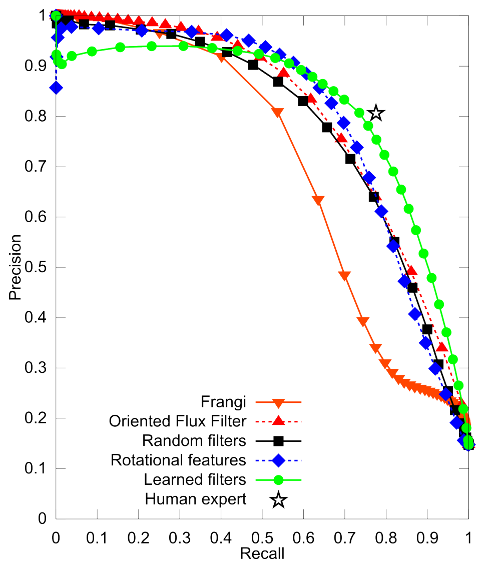
(a) DRIVE, different refinements, ROC curve

Figure 2: ROC curves (with the False Positive Rate expressed in logarithmic scale to better distinguish the curves) corresponding to the P/R curves presented in the paper.

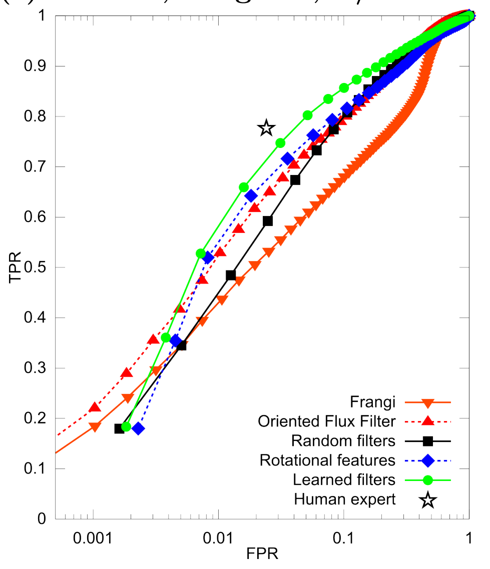
than those achieved with competing approaches, and they get very close to the results obtained by the second human expert (Fig. 4(c)). A detailed view on a randomly chosen region is presented in Fig. 6. An analogous



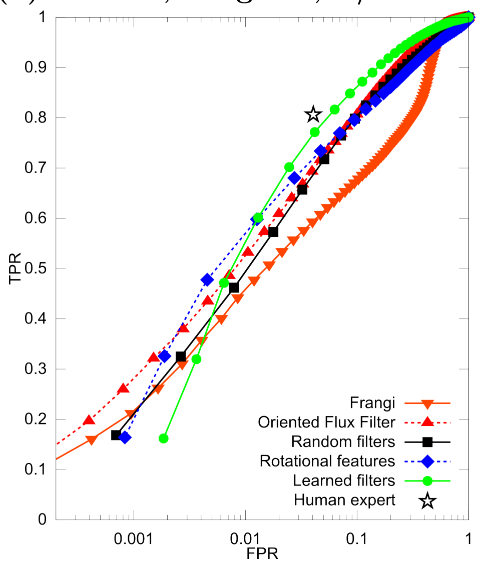
(c) DRIVE, image 12, P/R curve



(d) DRIVE, image 13, P/R curve



(a) DRIVE, image 12, ROC curve



(b) DRIVE, image 13, ROC curve

Figure 3: Precision/Recall and ROC curves (with the False Positive Rate expressed in logarithmic scale to better distinguish the curves) curves for two randomly chosen images from the DRIVE dataset.

Table 7: Analytic measures of the quality of the pixel classification for the experiments presented in Fig. 2(d) and Fig. 3. Both the VI and the RI are computed on the images thresholded at the value found using the F-measure. Please note that VI assumes values in $[\,0, \infty)$, the lower the better, and RI assumes values in $[0, 1]$, the higher the better.

Method	**AUC**	**F-measure**	**VI**	**RI**
DRIVE image 19, refinements				
$\lambda_{segm} = 1.0 \cdot 10^{-1}$	0.9402	0.7906	0.5263	0.9004
$\lambda_{segm} = 7.5 \cdot 10^{-2}$	0.9467	0.7955	0.5380	0.8977
$\lambda_{segm} = 5.0 \cdot 10^{-2}$	0.9612	0.8134	0.4910	0.9096
$\lambda_{segm} = 2.5 \cdot 10^{-2}$	0.9628	0.8068	0.5112	0.9046
$\lambda_{segm} = 1.0 \cdot 10^{-2}$	0.9659	0.8139	0.4813	0.9115
$\lambda_{segm} = 1.0 \cdot 10^{-3}$	0.9660	0.8147	0.4986	0.9077
Convolution	**0.9690**	**0.8299**	**0.4626**	**0.9166**
DRIVE image 12				
Ground truth		*0.8031*	*0.5078*	*0.9049*
Frangi	0.8696	0.6294	0.7202	0.8400
Random filters - SVM	0.9181	0.6931	0.6999	0.8528
Rotational features - SVM	0.9170	0.7372	0.6038	0.8782
Oriented Flux Filter	0.9258	0.7164	0.6318	0.8698
Learned Filters - SVM	**0.9377**	**0.7660**	**0.5737**	**0.8879**
DRIVE image 13				
Ground truth		*0.7913*	*0.6101*	*0.8821*
Frangi	0.8697	0.6510	0.7562	0.8323
Random filters - SVM	0.9185	0.7158	0.7154	0.8510
Rotational features - SVM	0.9131	0.7402	0.6513	0.8670
Oriented Flux Filter	0.9326	0.7217	0.6995	0.8550
Learned Filters - SVM	**0.9465**	**0.7698**	**0.6110**	**0.8792**

comparison for the neurons dataset, with TPR fixed to 0.9, is reported in Figs. 7-12, while a detailed view with FPR 0.05 is given in Fig. 13. Finally, colorized classifications for the roads dataset are shown in Fig. 14 and Fig. 15 for a FPR of 0.05.

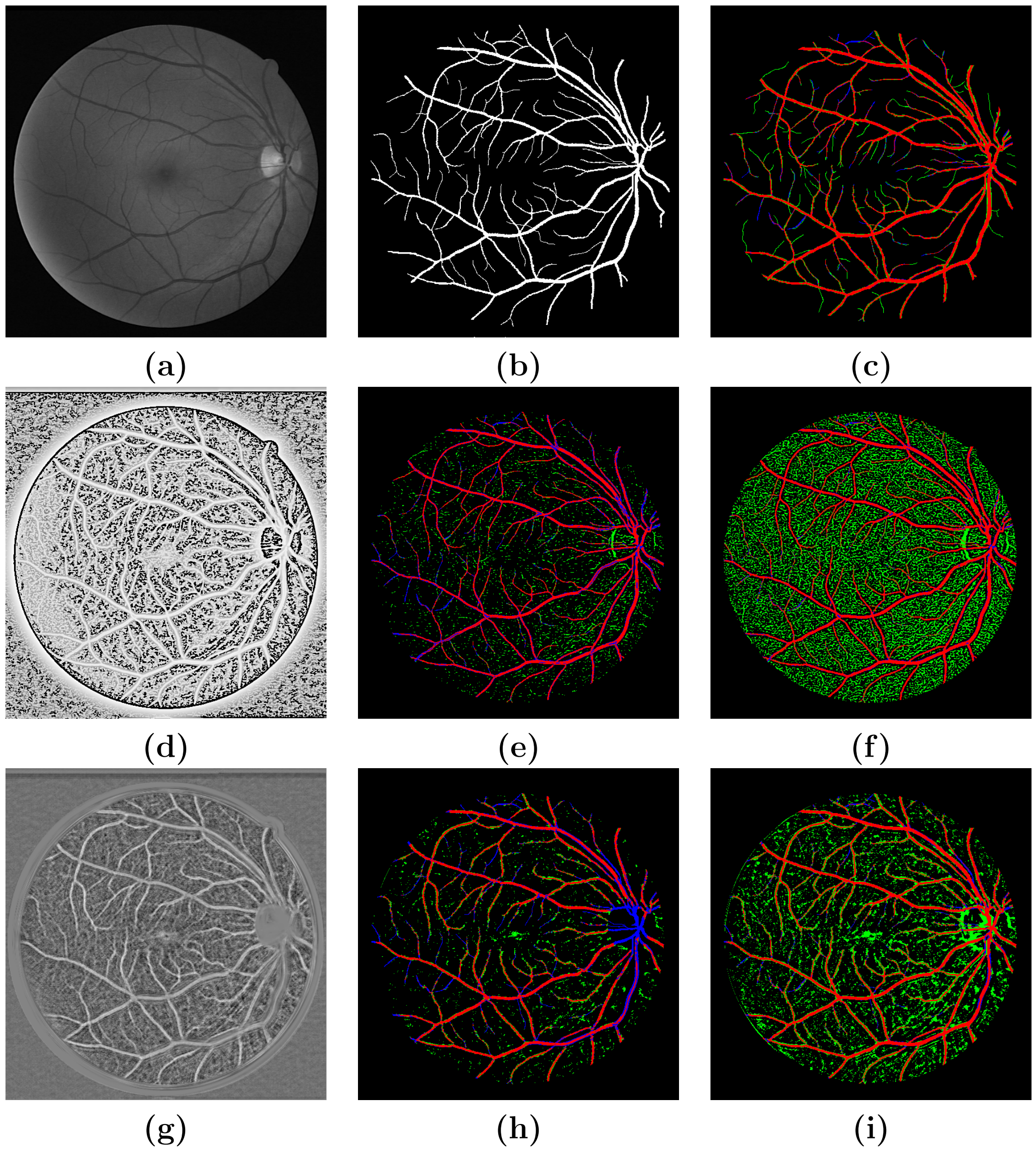
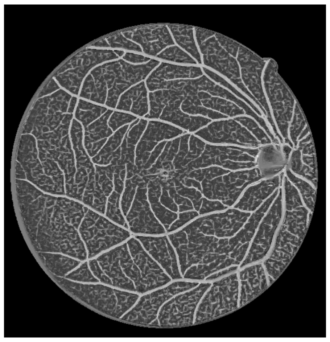
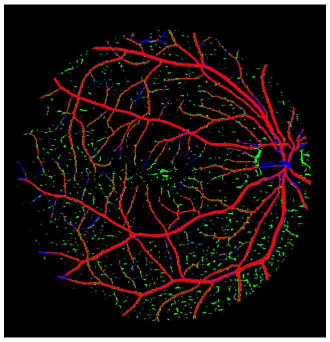
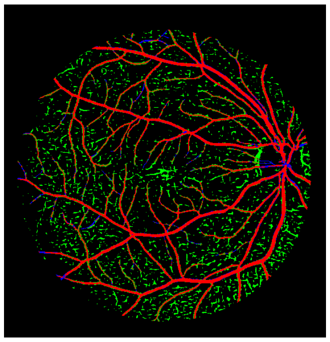
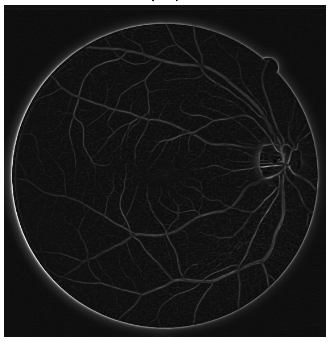


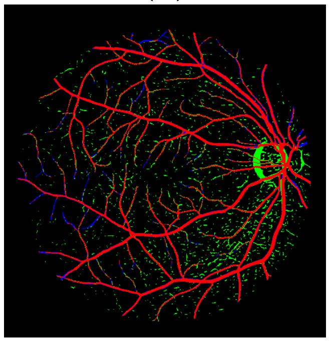
Figure 4: Visual results of pixel classification for the image 19 of the DRIVE dataset **(part I)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Original image. **(b)** Ground truth created by the first human expert. **(c)** Comparison between the ground truth provided by the two human experts. The presence of faint vessels accounts for most of the differences between the two. **(d)** Probabilistic output obtained using the method presented by Frangi *et al.* [5]. **(e)** Colorized classification for a FPR of 0.05 (method: Frangi *et al.* [5]). **(f)** Colorized classification for a TPR of 0.9 (method: Frangi *et al.* [5]). **(g)** Probabilistic output obtained using Random Filters. **(h)** Colorized classification for a FPR of 0.05 (method: Random Filters). **(i)** Colorized classification for a TPR of 0.9 (method: Random Filters).

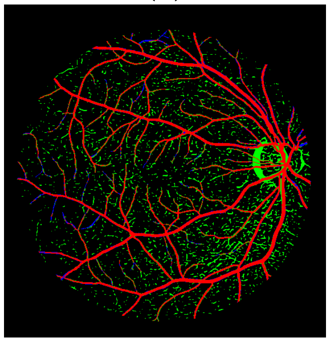

(a)

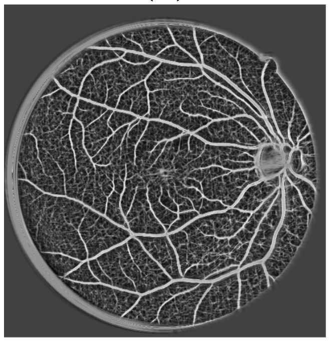

(b)

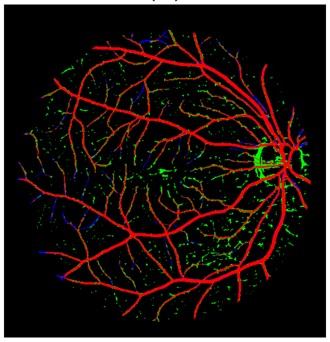

(c)


(d)


(e)


(f)


(g)


(h)

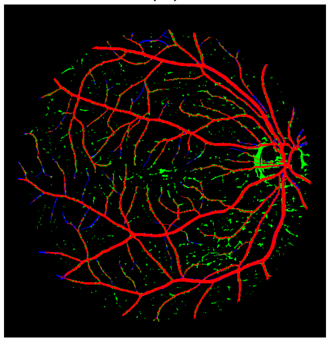

(i)

Figure 5: Visual results of pixel classification for the image 19 of the DRIVE dataset **(part II)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Probabilistic output obtained using Rotational Features [6]. **(b)** Colorized classification for a FPR of 0.05 (method: Rotational Features [6]). **(c)** Colorized classification for a TPR of 0.9 (method: Rotational Features [6]). **(d)** Probabilistic output obtained using the Oriented Flux Filter [7]. **(e)** Colorized classification for a FPR of 0.05 (method: Oriented Flux Filter [7]). **(f)** Colorized classification for a TPR of 0.9 (method: Oriented Flux Filter [7]). **(g)** Probabilistic output obtained using Learned Filters. **(h)** Colorized classification for a FPR of 0.05 (method: Learned Filters). **(i)** Colorized classification for a TPR of 0.9 (method: Learned Filters)

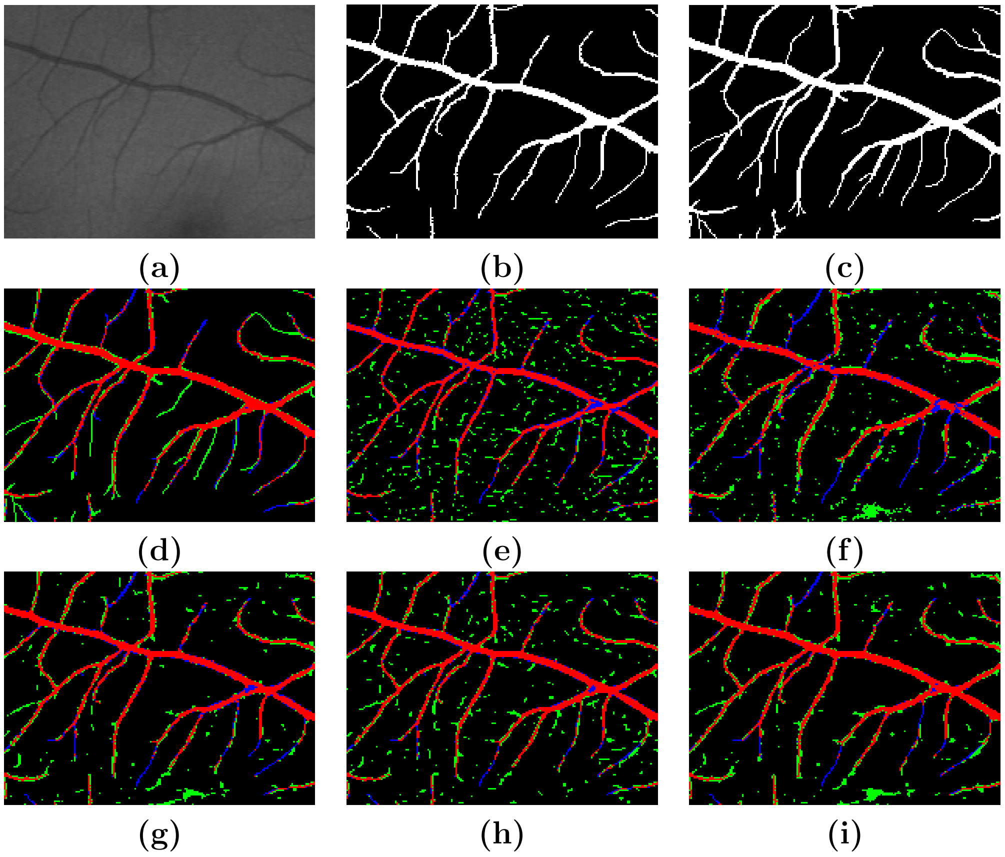


Figure 6: Detailed view of the colorized classification results for the image 19 of the DRIVE dataset at a FPR of 0.05. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Cropped segment of the original image. **(b)** Cropped segment of the segmentation created by the first human expert. **(c)** Cropped segment of the segmentation created by the second human expert. The substantial differences in the segmentations provided by the two experts outlines the difficulty of the task. **(d)** Comparison of the results created by the two human experts. **(e)** Colorized classification obtained using the method proposed by Frangi *et al.* [5]. **(f)** Colorized classification obtained using Random Filters. **(g)** Colorized classification obtained using Rotational Features [6]. **(h)** Colorized classification obtained using the Oriented Flux Filter [7]. **(i)** Colorized classification obtained using Learned Filters.

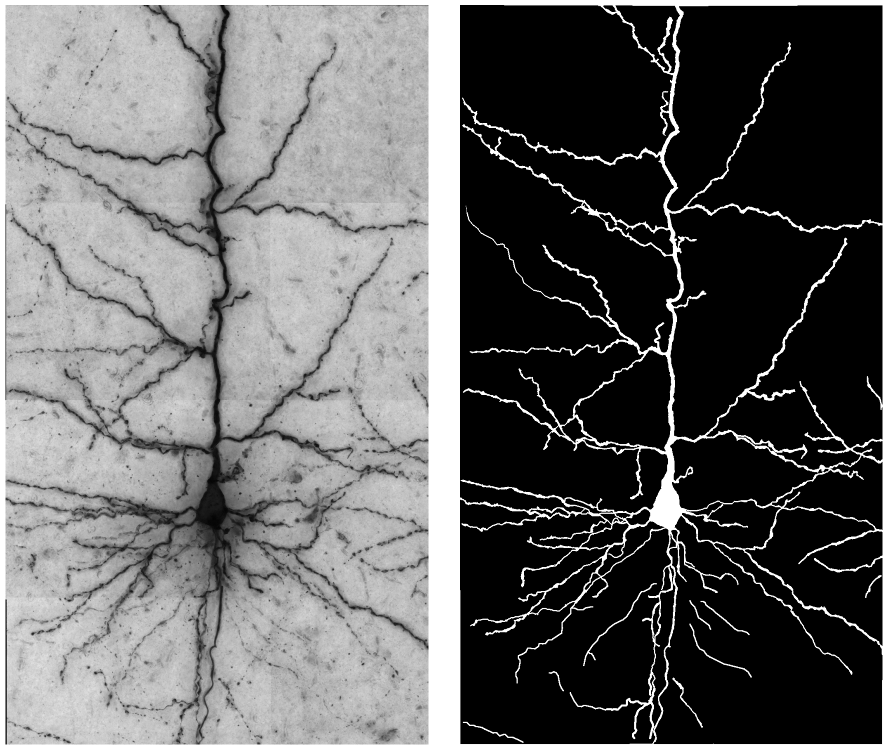
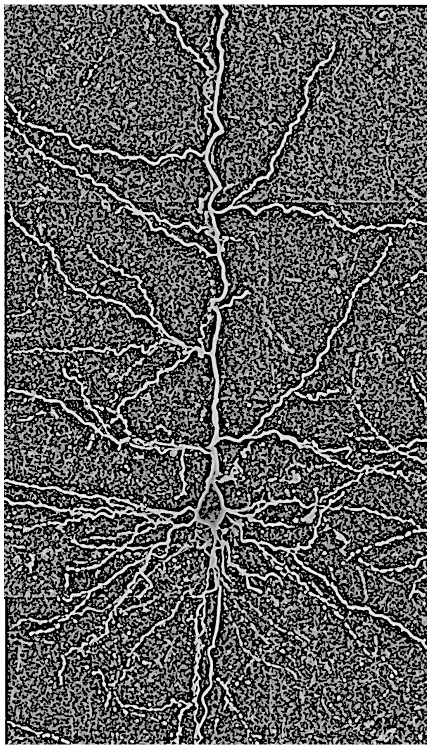

(a) (b)

Figure 7: Visual results of pixel classification for the neurons dataset for a TPR of 0.9 **(part I)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Original image. **(b)** Ground truth created by a human expert.


(a)

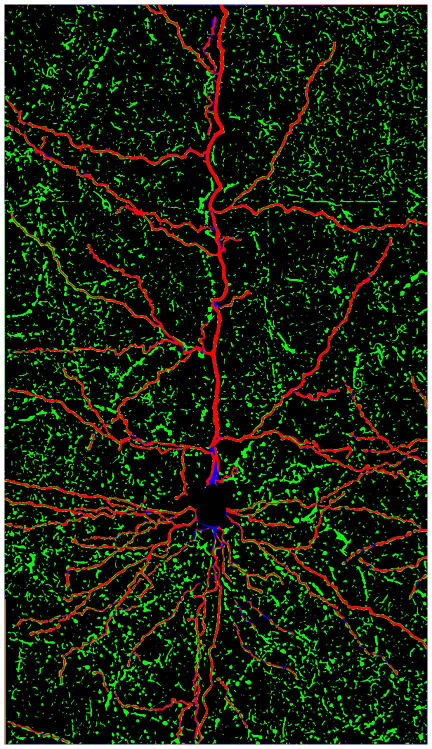

(b)

Figure 8: Visual results of pixel classification for the neurons dataset for a TPR of 0.9 **(part II)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Probabilistic output obtained using the method presented by Frangi *et al.* [5]. **(b)** Colorized classification derived from the output of the method of Frangi *et al.* [5].

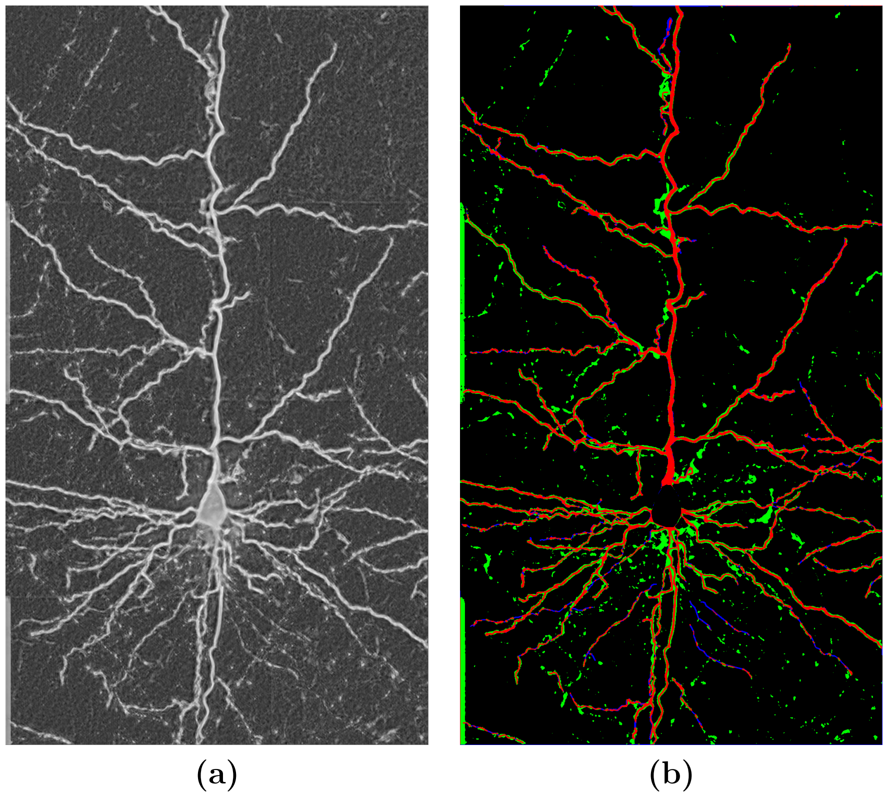


Figure 9: Visual results of pixel classification for the neurons dataset for a TPR of 0.9 **(part III)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Probabilistic output obtained using Random Filters. **(b)** Colorized classification derived from the output obtained using Random Filters.

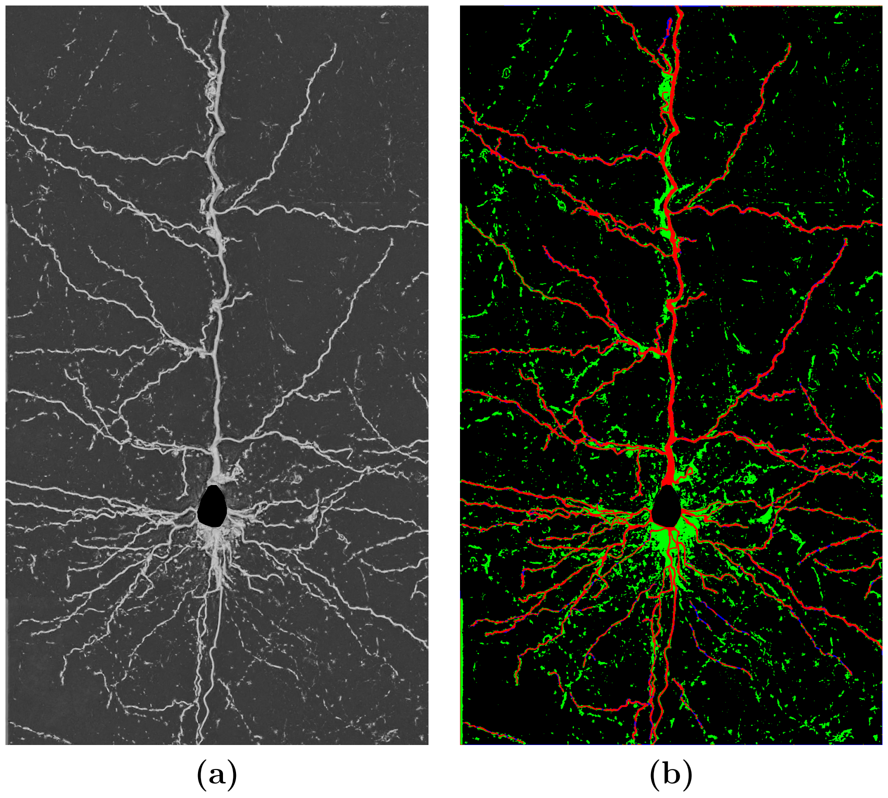


Figure 10: Visual results of pixel classification for the neurons dataset for a TPR of 0.9 **(part IV)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Probabilistic output obtained using Rotational Features [6]. **(b)** Colorized classification derived from the output obtained using Rotational Features [6].

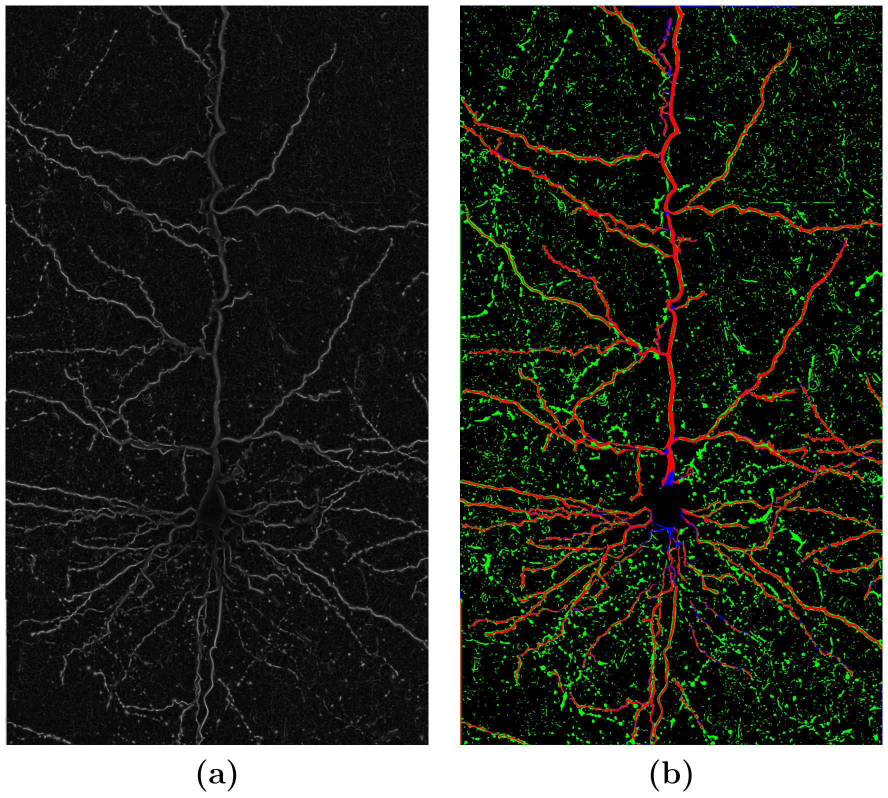


Figure 11: Visual results of pixel classification for the neurons dataset for a TPR of 0.9 **(part V)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Probabilistic output obtained using the Oriented Flux Filter [7]. **(b)** Colorized classification derived from the output obtained using the Oriented Flux Filter [7].

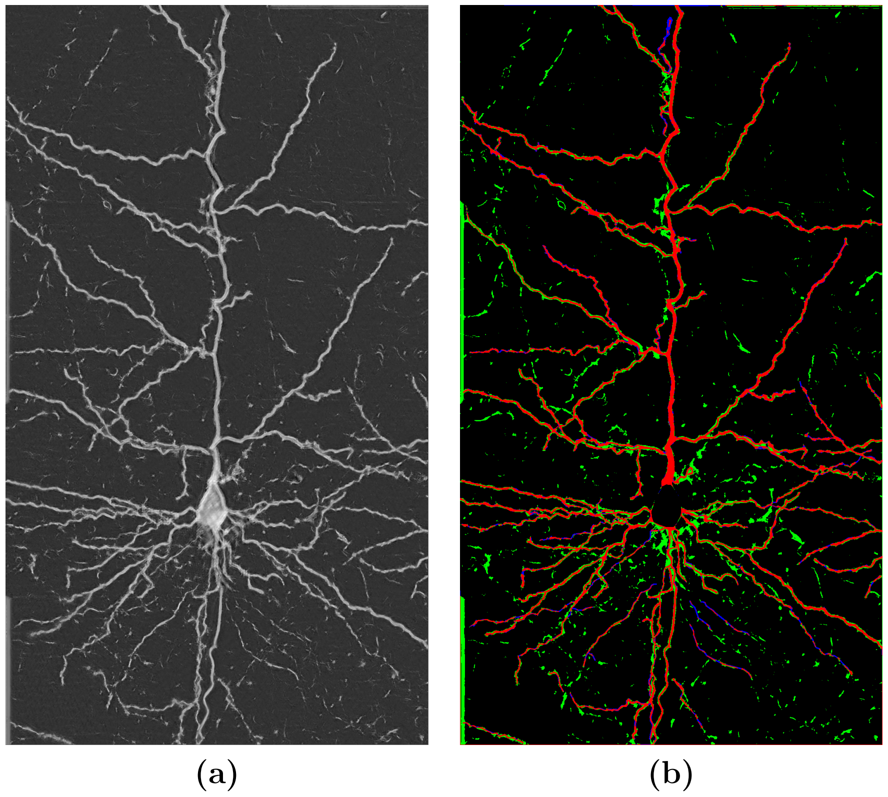


Figure 12: Visual results of pixel classification for the neurons dataset for a TPR of 0.9 **(part VI)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Probabilistic output obtained using Learned Filters. **(b)** Colorized classification derived from the output obtained using Learned Filters.

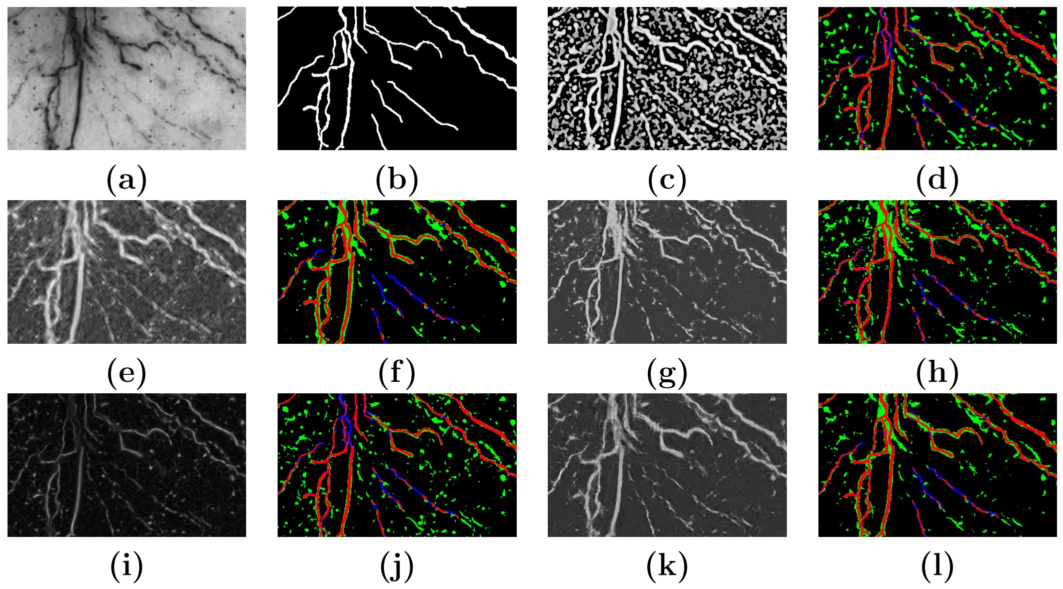
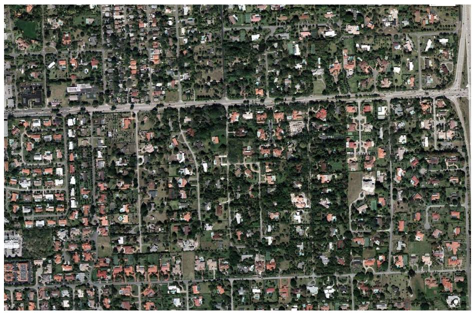
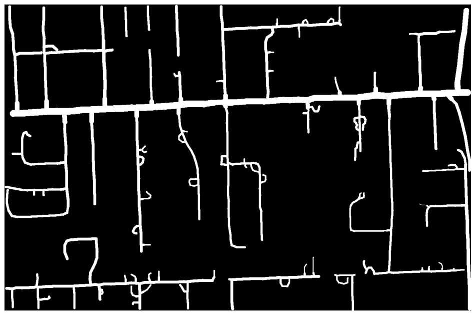


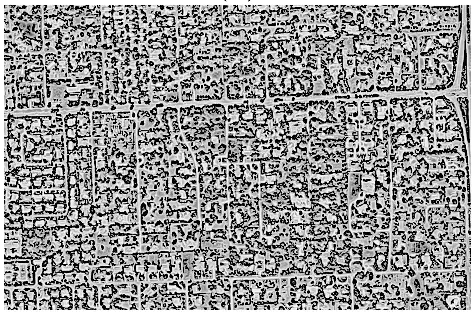
Figure 13: Detailed view of the colorized classification results for the neurons dataset at a FPR of 0.05. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Cropped segment of the original image. **(b)** Cropped segment of the classification created by the human expert. **(c)** Cropped segment of the classification obtained using the method proposed by Frangi *et al.* [5]. **(d)** Colorized classification obtained using the method proposed by Frangi *et al.* [5]. **(e)** Cropped segment of the classification obtained using Random Filters. **(f)** Colorized classification derived from the output obtained using Random Filters. **(g)** Cropped segment of the classification obtained using Rotational Features [6]. **(h)** Colorized classification derived from the output obtained using Rotational Features [6]. **(i)** Cropped segment of the classification obtained using the Oriented Flux Filter [7]. **(j)** Colorized classification derived from the output obtained using the Oriented Flux Filter [7]. **(k)** Cropped segment of the classification obtained using Learned Filters. **(l)** Colorized classification derived from the output obtained using Learned Filters.



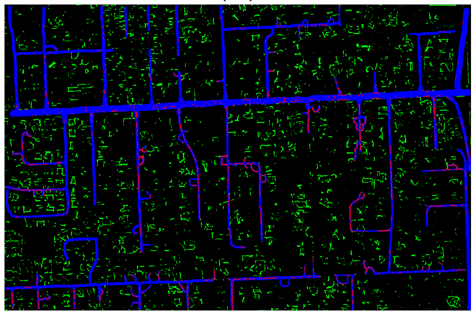
(a)



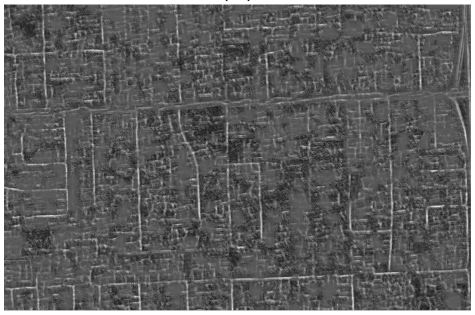
(b)



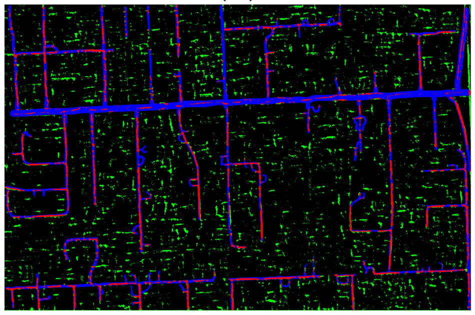
(c)



(d)



(e)



(f)

Figure 14: Visual results of pixel classification for the roads dataset for a FPR of 0.050 **(part I)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Original image. **(b)** Ground truth created by a human expert. **(c)** Probabilistic output obtained using the method presented by Frangi *et al.* [5]. **(d)** Colorized classification derived from the output of the method of Frangi *et al.* [5]. **(e)** Probabilistic output obtained using Random Filters. **(f)** Colorized classification derived from the output obtained by using Random Filters.

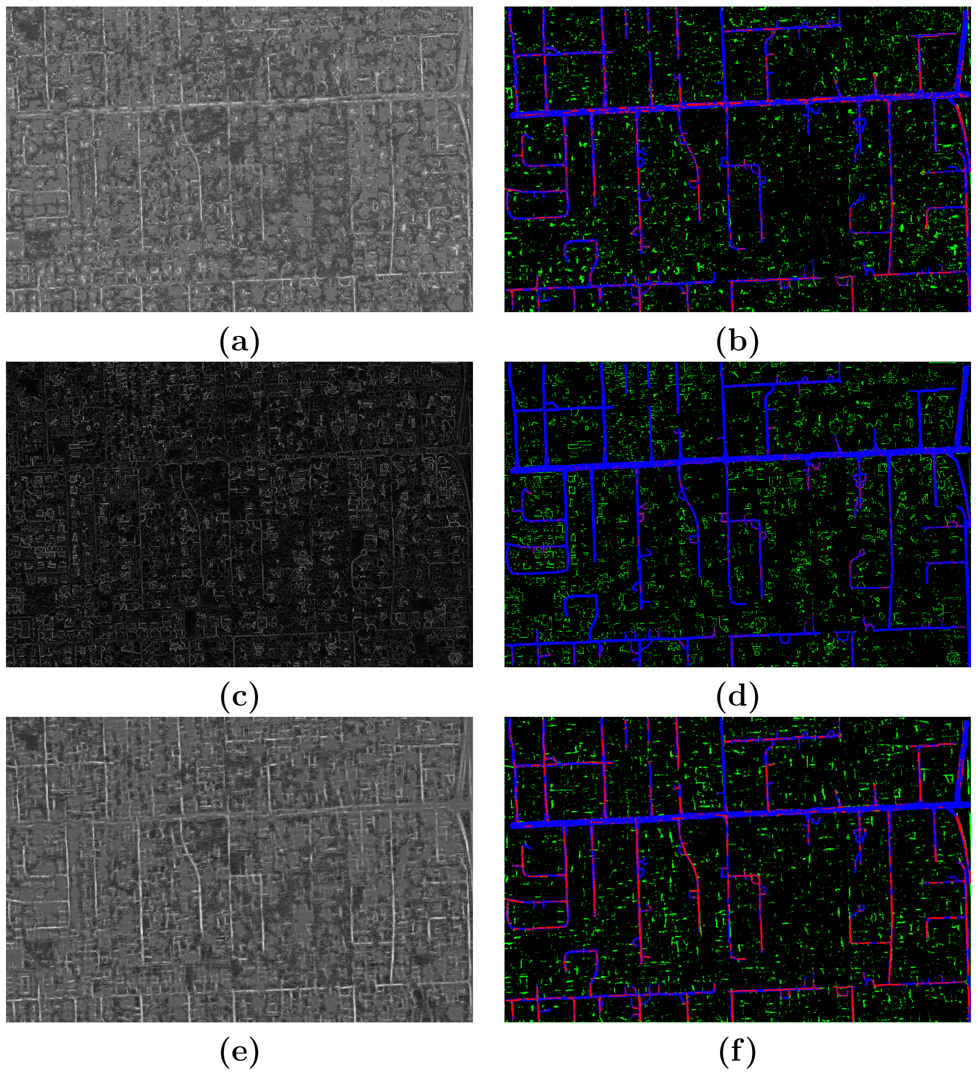


Figure 15: Visual results of pixel classification for the roads dataset for a FPR of 0.050 **(part II)**. True positive pixels are shown in red, false positives in green, and false negatives in blue. **(a)** Probabilistic output obtained using Rotational Features [6]. **(b)** Colorized classification derived from the output obtained by using Rotational Features [6]. **(c)** Probabilistic output obtained using the Oriented Flux Filter [7]. **(d)** Colorized classification derived from the output obtained by using the Oriented Flux Filter [7]. **(e)** Probabilistic output obtained using Learned Filters. **(f)** Colorized classification derived from the output obtained by using Learned Filters.

References

[1] I. Daubechies, M. Defrise, C. D. Mol, An iterative thresholding algorithm for linear inverse problems with a sparsity constraint, Comm. Pure Appl. Math. (2004).

[2] A. Coates, A. Y. Ng, The Importance of Encoding Versus Training with Sparse Coding and Vector Quantization, in: Int. Conf. on Mach. Learn.

[3] K. Jarrett, K. Kavukcuoglu, M. A. Ranzato, Y. LeCun, What is the Best Multi-Stage Architecture for Object Recognition?, in: Int. Conf. on Comput. Vis.

[4] T. Leung, J. Malik, Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons, Int. J. Comput. Vision (2001).

[5] A. F. Frangi, W. J. Niessen, K. L. Vincken, M. A. Viergever, Multiscale vessel enhancement filtering, in: Med. Image Comput. Comput. Assist. Interv.

[6] G. González, F. Fleuret, P. Fua, Learning Rotational Features for Filament Detection, in: IEEE Conf. on Comput. Vis. and Pattern Recogn.

[7] M. Law, A. Chung, Three Dimensional Curvilinear Structure Detection Using Optimally Oriented Flux, in: Europ. Conf. on Comput. Vis.