Optimizing the Relevance-Redundancy Tradeoff for Efficient Semantic Segmentation

Caner Hazırbaş, Julia Diebold, and Daniel Cremers

Technical University of Munich, Germany
{c.hazirbas,julia.diebold,cremers}@tum.de

Abstract. Semantic segmentation aims at jointly computing a segmentation and a semantic labeling of the image plane. The main ingredient is an efficient feature selection strategy. In this work we perform a systematic information-theoretic evaluation of existing features in order to address the question which and how many features are appropriate for an efficient semantic segmentation. To this end, we discuss the tradeoff between relevance and redundancy and present an information-theoretic feature evaluation strategy. Subsequently, we perform a systematic experimental validation which shows that the proposed feature selection strategy provides state-of-the-art semantic segmentations on five semantic segmentation datasets at significantly reduced runtimes. Moreover, it provides a systematic overview of which features are the most relevant for various benchmarks.

Keywords: Feature analysis, feature selection, image segmentation, semantic scene understanding

1 Introduction

1.1 Semantic Segmentation and Feature Selection

Semantic segmentation – sometimes also referred to as class-specific segmentation – aims at jointly computing a partitioning of the image plane and a semantic labeling of the various regions in terms of previously learned object classes. Numerous works are focused on the development of sophisticated regularizers for this problem: co-occurrence priors [9,18] have been suggested to learn and penalize the joint occurrence of semantic labels within the same image. Proximity priors [1] have been proposed to penalize the co-occurrence of labels within a certain spatial neighborhood. Hierarchical priors [17,20] have been introduced to impose certain label hierarchies – for example that an office is composed of chairs and tables, whereas an outdoor scene is composed of water, grass, cows, etc. Proportion priors [11] have been proposed to learn and impose priors on the relative size of object parts. The quantitative performance in terms of segmentation accuracy of respective methods, however, is generally dominated by respective data terms. In this paper we therefore focus on the data term.

A multitude of data terms have been proposed over the last years to take texture, color, spatial location and even depth into account in the construction

of appropriate observation likelihoods associated with each pixel. Not surprisingly, depending on the object class and image benchmark, some features are more relevant than others. While in principle taking more and more features into account should improve the segmentation accuracy, in the interest of computational efficiency, the redundancy among features should be minimized. How can we quantify relevance and redundancy of features? How can we devise a systematic feature selection strategy to identify a small set of optimal features for semantic image segmentation? And how can we automatically determine the number of features to use?

In this work we make use of information-theoretic quantities in order to characterize and optimize the relevance and redundancy tradeoff of respective features for semantic segmentation. An overview of the studied features is given in Section 2. For two continuous random variables X and Y, the *mutual information*

$$MI(X;Y) = \int_Y \int_X p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy, \tag{1}$$

is a measure of the mutual dependency of X and Y, where p denotes their probability density function. A feature f_i is relevant for the class labeling c if the mutual information $MI(f_i;c)$ of the feature and the class label is large. Moreover, it is redundant with respect to another feature f_j if the mutual information $MI(f_i;f_j)$ is high. In the following we will show that an appropriate information-theoretic feature selection strategy will lead to semantic segmentation methods which provide state-of-the-art performance at substantially reduced computation time. Figure 1 shows the improvement of classification accuracy on different benchmarks with increasing size of the feature set. The features are ordered based on their relevance and redundancy.

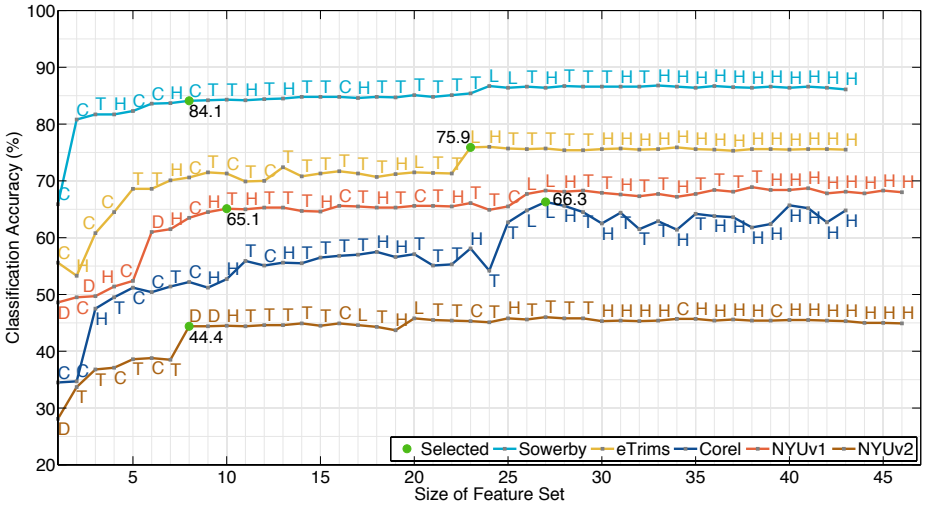


Fig. 1. Impact of features on the classification accuracy. The labels indicate the type of feature added to the feature set: Haar-like (H), color (C), texton (T), location (L) and depth (D). For each benchmark a green dot indicates the feature set which is selected by the proposed approach.

This paper is organized as follows: we introduce the studied features in Section 2. In Section 3 we propose an information-theoretic feature analysis method and in Section 4 we show the list of ranked and selected features for five different benchmarks. Finally, we compare our runtime as well as qualitative and quantitative results to state-of-the-art methods (Section 5).

1.2 Related Work

The literature on object detection can be roughly grouped into two complementary approaches. Conventional object detectors deal with the task of finding bounding boxes around each object [4,10,19]. In contrast, dense object detection approaches [7,13] focus on detecting objects at pixel level. We focus on the choice of the best visual object recognition features for dense object detection.

Shotton *et al.* [13] proposed texture-layout filters based on textons which jointly model patterns of texture and their spatial layout for dense object detection. As they use a large set of features in their computations, their method is not applicable in real-time. On the contrary, our method only chooses the most *significant* features. Thus, we are able to improve the detection performance at a highly reduced runtime.

In 2012, Fröhlich *et al.* [5] proposed an iterative approach for semantic segmentation of a facade dataset. This approach uses millions of features and refines the semantic segmentation by iteratively adding context features derived from coarser levels to a Random Forests classifier. As a result, this approach is fairly slow. In contrast, we determine the optimal set of features and are thus able to receive similar detection accuracies with a significantly smaller set of features at a reduced runtime.

Couprie *et al.* [3] introduced a multiscale convolutional network to segment indoor RGB-D images. They implicitly compute and select features by constructing complex and deep architectures. In contrast, our method is based on a transparent selection criterion.

Recently, Hermans *et al.* [7] discussed 2D semantic segmentation for RGB-D sensor data in order to reconstruct 3D scenes. They use a very basic set of features. However, this basic set of features is determined by experiments and no clear selection criterion is given. In general, none of the above approaches gives justification for their chosen set of features. We specifically address this problem and give detailed explanations on how to choose the best feature set for dense object detection.

1.3 Contributions

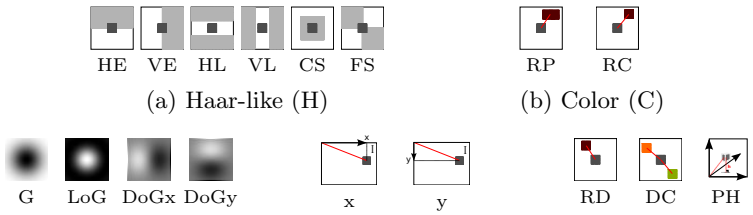
We present an information-theoretic feature analysis method which resolves the following challenges:

+ We answer the questions *which features are the most significant for object recognition* and *how many features are needed for a good tradeoff between accuracy and runtime.*

+ The proposed feature analysis method is easy to use and immediately applicable to different datasets. It runs fast in real-time even on large datasets with high-resolution images. All parameters are determined automatically from the information-theoretic formulation.[1]
+ We evaluate our method on five different datasets and compare our classification and segmentation results with the state-of-the-art methods by Shotton *et al.* [14], Fröhlich *et al.* [5], Couprie *et al.* [3] and Hermans *et al.* [7]. The proposed feature selection strategy provides state-of-the-art semantic classifications and segmentations at significantly reduced runtimes.

2 The Feature Set

We consider 17 shape and texture features composed of 6 Haar-like, 2 color, 4 texton, 2 location and 3 depth features. The features are computed in a patch surrounding the image pixels. Thereby, different patch sizes of the features are used. We convert the images from the RGB(-D) to the CIELab color space and compute the features on the channels: L, a, b (and D). Depth maps are normalized and converted to gray scale.



(a) Haar-like (H) (b) Color (C) (c) Texton (T) (d) Location (L) (e) Depth (D)

Fig. 2. The feature set. Illustration of the 17 shape and texture features which are studied in various patch sizes on different image channels. We analyze the significance of the features and explain which and how many of them to use.

Haar-like features We use six types of Haar-like features [19]: horizontal and vertical edge (HE/VE) and line (HL/VL), center surround (CS) and four square (FS) illustrated in Figure 2a.

Color features We use the average of the relative patch (RP) and the relative color (RC) feature shown in Figure 2b.

Texton features As texton features we use Gaussian filter (G), Laplacian of Gaussian (LoG) and the first order derivatives of Gaussian filter (DoG) in x and y direction with different bandwidths (see Figure 2c).

Location features We use normalized canonical location features (see Figure 2d) computed for each pixel p in the image I.

Depth features We use the relative depth (RD), the relative depth comparison (DC) and the height of a pixel (PH) [7], illustrated in Figure 2e.

[1] Our code is publicly available at `vision.in.tum.de/data/software`

3 Feature Ranking and Selection for Object Recognition

Among the discussed features, *which* are the most significant for object recognition? And *how many* are needed for a good tradeoff between accuracy and runtime? To this end, we first rank the features according to their significance, then we analyze them and propose an automatic selection criterion.

3.1 Feature Ranking

In the first step, a set of training images is used to compute a ranked set of features $\mathcal{F}_R$ of the full feature set $\mathcal{F}$ where the ranking is based on significance. As described in the introduction, features are significant if they are relevant for the classification performance but as little redundant as possible. Ideally, the optimal set of features $\{f_1, .., f_N\}$ is obtained by maximizing the expression

$$\max_{\{f_1,..,f_N\}\in\mathcal{F}} \sum_{f_i\in\mathcal{F}} MI(f_i;c) - \frac{1}{N}\sum_{f_i,f_j\in\mathcal{F}} MI(f_i;f_j), \tag{2}$$

where the first term aims at maximizing the relevance of each feature in terms of the mutual information with respect to the target class c and the second term aims at minimizing the redundancy between pairs of features. We call a feature *significant* if it maximizes the relevance for the classification task while minimizing the redundancy with respect to the other features. First of all, the joint optimization over all features is computationally demanding. Secondly, it does not provide us with a ranking of features by significance.

To address these drawbacks, we revert to a greedy strategy for feature selection introduced by Peng *et al.* [12] in the context of biological feature analysis and handwritten digit recognition.

For a fixed target class c, let $\mathcal{F}_{m-1} = \{f_1, \ldots, f_{m-1}\}$ be the *best* feature set with $m-1$ features. To identify the *best additional* feature $f_m \in \mathcal{F} \setminus \mathcal{F}_{m-1}$, we simply optimize its relevance-redundancy tradeoff with respect to the existing features:

$$f_m = \operatorname*{arg\,max}_{f_i\in\mathcal{F}\setminus\mathcal{F}_{m-1}} \left[MI(f_i;c) - \frac{1}{m-1}\sum_{f_j\in\mathcal{F}_{m-1}} MI(f_i;f_j)\right]. \tag{3}$$

This leads to a set of features $\mathcal{F}_R = \{f_1, \ldots, f_N\}$ which are ranked with respect to their significance for the target class c.

3.2 Automatic Feature Selection

Let the first n features in $\mathcal{F}_R$ be denoted by $\mathcal{F}_R(n) := \{f_1, \ldots, f_n\}$. In the following step, we determine $n^* \in \{1, \ldots, N\}$ such that $\mathcal{F}_R(n^*)$ consists of only the most significant features. Therefore, we initially apply an *incremental feature analysis* returning the classification accuracy $Acc(n)$ for each feature set $\mathcal{F}_R(n)$. Algorithm 1 sketches the steps we carried out to obtain $\big(Acc(1), \ldots, Acc(N)\big)$.

To figure out *how many* features $n^* \in \{1, \ldots, N\}$ to choose, the following conditions have to be met: a) For optimizing the runtime a small n^* is preferred, while b) for the optimization of the accuracy a large n^* is desired. Hence, n^* should be small but still lead to a satisfying accuracy. We therefore propose the following optimization criterion:

$$n^* = \underset{n \in \{1,\ldots,N\}}{\arg\max} \left(Acc\,(n)\right)^{\alpha} (N + 1 - n)^{\frac{1}{\beta}}, \tag{4}$$

where $\alpha, \beta \geq 1$ (we set $\alpha = 5$, $\beta = 2$). This function jointly maximizes the accuracy $Acc(n)$ and minimizes the number of features n. Taking $Acc(n)$ to the power of α emphasizes the jumps in the accuracy in which we are interested. Taking the βth root of $(N - n)$ prevents too strong influence of the size of $\mathcal{F}_R(n)$. By varying the values of α and β, the method can be adapted to the user's interest focusing on optimal runtime and/or accuracy.

Algorithm 1 Incremental Feature Analysis

procedure AnalyzeFeatures($\mathcal{D}, \mathcal{F}_R$) ▷ $\mathcal{D}$: Dataset, $\mathcal{F}_R$: Ranked Features
  $n = 0$, $Acc = \emptyset$ ▷ Acc: Classification Accuracy
  while $n < N$ **do**
    $n \leftarrow n + 1$
    Extract the features $\mathcal{F}_R(n)$ on the training set.
    Train K Random Trees $\{T_1(\cdot), \ldots, T_K(\cdot)\}$ on the training samples.
    For each class $c \in \{1, \ldots, C\}$ estimate the class probabilities $\widetilde{P}$ at each pixel p on the validation set: ▷ C: #Classes

$$\widetilde{P}(c \mid p, \mathcal{F}_R\,(n)) = \frac{\sum_{k=1}^{K} \left[T_k\big(p, \mathcal{F}_R\,(n)\big) == c\right]}{K}. \tag{5}$$

    Predict the class label $c^*(p)$ for each pixel p with:

$$c^*(p) = \underset{c \in \{1,\ldots,C\}}{\arg\max}\ \widetilde{P}(c \mid p, \mathcal{F}_R(n)).$$

    Compute $Acc(n)$ with the predicted class labels c^* on the validation set:

$$Acc(n) = \frac{\text{Number of correctly classified pixels}}{\text{Total number of labeled pixels}}.$$

  end while
  return Acc ▷ List of accuracies $\big(Acc(1), \ldots, Acc(N)\big)$
end procedure

This two-step approach leads to the feature set $\mathcal{F}_R(n^*)$ which consists of only the most significant features for the respective dataset. Compared to other approaches which mostly use arbitrary large feature sets, we are able to obtain competitive classification accuracies at a remarkably reduced runtime.

Related works such as [5,7] mostly tune the parameters used for training the Random Forests. In contrast, we use default settings for all benchmarks. Our

experimental results (Section 5) show that the choice of the right features is more important than the best parameter settings for Random Forests. Reduced redundancy in the feature set keeps the accuracy high while it decreases the runtime significantly.

3.3 Implementation

The algorithm runs fast in real-time even on large datasets with high-resolution images. The whole algorithm runs on a single CPU. We restricted the system to the minimal number of parameters. This makes the application independent from parameter tuning for different benchmarks. Except for the patch size of the features and the grid size Δ_{ss} all other parameters are fixed. Therefore, the proposed method is easy to use and immediately applicable for different datasets.

4 Which and How Many Features?

We apply the proposed feature ranking and selection method using the 17 shape and texture features introduced in Section 2 on five different benchmarks. In the following we discuss the resulting significance of the different features. We made similar observations on all benchmarks.

The following benchmarks are studied: (i) the 8-class facade dataset eTrims [8], (ii) the 7-class Corel and (iii) Sowerby datasets [6] and (iv) the 12-class NYUv1 [15,7] as well as (v) the 13-class NYUv2 [16,3] RGB-D benchmark. For the eTrims dataset we follow Fröhlich *et al.* [5] and split the dataset by a ratio of 60/40 for training and testing. We split the Corel and Sowerby benchmark by a ratio of 60/40, the NYUv1 dataset by a ratio of 50/50 and the NYUv2 by 55/45 for training and testing, similar to [3]. For each benchmark 20% of the training set is used as validation set. On the Corel benchmark we follow Shotton *et al.* [14] and normalize the color and intensity of the images.

For the eTrims, Corel and Sowerby benchmarks we use 50 trees to train the Random Forests, each having at most a depth of 15. For the NYUv1 and NYUv2 benchmark we follow Hermans *et al.* [7] and use 8 trees, each having at most a depth of 10.

To reduce the computational cost during the training process, filter responses are computed on a $\Delta_{ss} \times \Delta_{ss}$ grid on the image [14]. We set $\Delta_{ss} = 3$ for the Corel and Sowerby benchmark and $\Delta_{ss} = 5$ for the eTrims, NYUv1 and NYUv2 benchmark.

4.1 Which Features

The ranked set of features $\mathcal{F}_R$, listed in Table 1, is computed for each dataset with the method proposed in Section 3.1. The following observations can be made on the relevance of the studied features:

Haar-like features *(orange)* In the literature Haar-like features are commonly evaluated on a gray-scale image or on the luminance channel. Table 1, however, shows that for all five benchmarks the top ranked Haar-like features are particularly those ones evaluated on the 'a' and 'b' color channel.

Color features *(turquoise)* Independently of the benchmark, almost all color features appear among the top ranked features. Hence, color features should definitely be used for training object classifiers.

Texton features *(gray)* Several texton features are ranked on a top position. Most of the higher ranked texton features (≤ 20) are computed on the 'L' channel. We conclude that texton features are more distinctive on the luminance channel.

Location features *(blue)* All location features are ranked in the lower half (≥ 17). However, for the eTrims and Corel benchmark, they significantly enhance the classification accuracy (cf. Figure 1).

Depth features *(purple)* are only available for the NYUv1 and NYUv2 benchmark (columns 4,5). All depth features are ranked among the top nine features and strongly boost the accuracy (cf. Figure 1).

We gained a valuable insight into the significance of various features for the task of pixel-wise object recognition. In summary, Haar-like features should particularly be evaluated on the color channels. Color features are important in general. Texton features should be considered especially on the 'L' channel. Location features can be essential and depth features are the most distinctive ones (when available).

4.2 How Many Features

In the following we answer the question on the best size of the feature set. For each benchmark, Figure 1 illustrates the classification accuracies $Acc(n)$ with increasing n. n indicates the size of the feature set $\mathcal{F}_R(n)$ which leads to $Acc(n)$ (cf. Algorithm 1). The green dots indicate the numbers n^* which are chosen by the proposed optimization criterion in Equation (4). The intention is to chose n^* small, but large enough to obtain an optimal tradeoff between accuracy and number of features.

For the eTrims benchmark, *e.g.*, the accuracy has a significant jump from $n = 22$ to $n = 23$. For values of n larger than 23, only very minor improvements can be achieved. Hence, one would prefer n^* to be equal to 23. As marked by the green dot in Figure 1, the proposed optimization criterion (4) selects $n^* = 23$. The accuracy plot computed for the Corel benchmark has a peak at $n = 27$. Thus, the selected $n^* = 27$ gives the best tradeoff between the accuracy and the size of the feature set. The accuracy plot for the NYUv2 benchmark shows a jump from $n = 7$ to $n = 8$. All values $n \in [9, 46]$ only provide an insignificant increase of $Acc(8)$. Thus, $n^* = 8$ is the perfect value for n and selected by Equation (4).

The accuracy plots obtained for the Sowerby and the NYUv1 benchmark show a less significant jump than the plots of the other benchmarks. For the Sowerby benchmark, the proposed method selects $n^* = 8$. Still, this value can be seen as optimal. For smaller values of n, the accuracy is not good enough. For larger values of n, up to $n = 23$, the accuracy improves only very little, whereas the feature set grows much more. The small gain in accuracy would have to be paid for by a much larger runtime. The same holds for the NYUv1 benchmark.

Table 1. Ranked features $\mathcal{F}_R$ for the eTrims, Corel, Sowerby, NYUv1 and NYUv2 benchmark. Different colors are set for Haar-like (H), color (C), texton (T), location (L) and depth (D) features. The features are labeled as follows: {feature type}_{feature name}_{patch size}_{color channel}. For an interpretation see Section 4.1.

Rank	eTrims	Corel	Sowerby	NYUv1	NYUv2
1	C_RP_25_b	C_RP_11_a	C_RC_7_a	D_PH_25_D	D_PH_25_D
2	H_VL_25_a	C_RC_11_b	C_RP_7_L	C_RP_25_a	T_G_3_L
3	C_RC_25_L	H_CS_11_L	T_DoGy_13x5_L	D_DC_25_D	T_LoG_17_L
4	C_RP_25_a	T_DoGy_13x5_L	H_CS_7_a	H_FS_25_b	C_RP_25_a
5	T_LoG_3_L	C_RP_11_L	C_RC_7_b	C_RC_25_b	T_LoG_5_L
6	T_G_3_L	C_RC_11_a	C_RP_7_a	D_RD_25_D	C_RC_25_L
7	H_CS_25_L	T_DoGx_9x25_L	H_VL_7_b	H_FS_25_L	T_G_5_L
8	C_RC_25_b	C_RP_11_b	C_RC_7_L	C_RP_25_L	D_RD_25_D
9	T_DoGy_25x9_L	H_HE_11_a	T_DoGx_9x25_L	H_CS_25_a	D_DC_25_D
10	C_RP_25_L	H_CS_11_a	T_DoGy_25x9_L	T_LoG_17_L	H_FS_25_L
11	T_DoGy_13x5_L	T_G_9_b	H_HL_7_a	H_VE_25_b	T_DoGy_25x9_L
12	C_RC_25_a	C_RC_11_L	T_G_9_b	T_LoG_3_L	T_G_9_L
13	T_G_9_a	H_CS_11_b	H_CS_7_b	T_DoGx_9x25_L	T_LoG_9_L
14	T_G_5_L	T_DoGy_25x9_L	T_LoG_3_L	T_LoG_5_L	T_DoGx_5x13_L
15	T_LoG_5_L	T_LoG_3_L	T_LoG_5_L	H_CS_25_b	T_LoG_3_L
16	T_LoG_9_L	T_LoG_5_L	C_RP_7_b	C_RC_25_a	C_RP_25_b
17	T_DoGx_9x25_L	T_LoG_9_L	H_CS_7_L	T_LoG_9_L	L_y
18	T_G_9_L	H_FS_11_a	T_LoG_9_L	H_HL_25_L	T_DoGx_9x25_L
19	H_VL_25_b	H_VL_11_a	T_DoGx_5x13_L	T_DoGy_25x9_L	H_HE_25_a
20	L_y	T_DoGx_5x13_L	T_G_3_L	C_RP_25_b	L_x
21	T_DoGx_5x13_L	T_G_5_L	T_G_9_a	T_G_9_a	T_G_5_b
22	T_G_9_b	T_G_9_a	T_G_3_a	H_FS_25_a	T_G_3_b
23	L_x	H_VL_11_b	T_G_3_b	T_G_3_b	C_RP_25_L
24	H_HE_25_L	T_G_3_L	L_x	T_G_5_a	T_DoGy_13x5_L
25	T_G_5_b	T_G_3_a	L_y	C_RC_25_L	H_VL_25_L
26	T_G_5_a	L_x	T_G_9_L	L_x	T_G_5_a
27	T_G_3_b	L_y	H_VL_7_L	L_y	T_G_3_a
28	T_G_3_a	H_VE_11_a	T_G_5_b	H_VE_25_a	T_G_9_b
29	T_LoG_17_L	T_G_9_L	T_G_5_L	T_DoGy_13x5_L	T_G_9_a
30	H_FS_25_a	H_HL_11_a	T_G_5_a	T_DoGx_5x13_L	H_FS_25_a
31	H_VL_25_L	T_G_5_b	H_HE_7_a	H_HE_25_L	H_HL_25_L
32	H_FS_25_b	T_G_3_b	T_LoG_17_L	T_G_5_b	H_CS_25_a
33	H_HL_25_a	T_G_5_a	H_VL_7_a	T_G_9_b	H_FS_25_b
34	H_VE_25_a	H_HL_11_b	H_FS_7_b	H_VL_25_b	C_RC_25_a
35	H_HE_25_a	T_LoG_17_L	H_VE_7_a	T_G_3_a	H_VE_25_a
36	H_CS_25_b	H_FS_11_b	H_FS_7_a	T_G_9_L	H_CS_25_b
37	H_CS_25_a	H_HE_11_b	H_HE_7_b	T_G_5_L	H_HE_25_b
38	H_HE_25_b	H_VE_11_b	H_VE_7_b	T_G_3_L	H_VE_25_b
39	H_VE_25_b	H_FS_11_L	H_FS_7_L	H_VL_25_L	C_RC_25_b
40	H_HL_25_b	H_VL_11_L	H_HL_7_b	H_HE_25_a	H_VL_25_a
41	H_FS_25_L	H_VE_11_L	H_VE_7_L	H_HE_25_b	H_HL_25_a
42	H_VE_25_L	H_HE_11_L	H_HE_7_L	H_VL_25_a	H_VL_25_b
43	H_HL_25_L	H_HL_11_L	H_HL_7_L	H_HL_25_a	H_HL_25_b
44				H_HL_25_b	H_HE_25_L
45				H_VE_25_L	H_VE_25_L
46				H_CS_25_L	H_CS_25_L

5 Experimental Results

Our framework chooses the feature set small but still large enough to obtain a satisfying accuracy. The above observations already show an experimental proof of the proposed feature ranking and selection method. In the following, we compare our runtime, classification and segmentation accuracies as well as qualitative results with state-of-the-art methods.

5.1 Significantly Improved Runtime

We ran our experiments on an Intel® Core™ i7-3770 3.40GHz CPU equipped with 32 GB RAM which is similar to the hardware used by competing approaches. Table 2 compares the training and testing runtimes for the classification task. Our framework runs much faster than state-of-the-art methods. In particular for the Sowerby and NYUv2 benchmark, we reduce the training time by a factor of 600 and 900, respectively. Furthermore, our method accelerates the testing runtime on all benchmarks.

Table 2. Comparison of runtimes for object classification in seconds. The training time is given for the whole training set whereas the testing time is averaged over all test images. The proposed method significantly outperforms the other methods in terms of training and testing runtime.

	eTrims		Corel		Sowerby		NYUv1		NYUv2	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
Shotton *et al.* [14]	-	-	1800	1.10	1200	2.50	-	-	-	-
Fröhlich *et al.* [5]	-	17.0	-	-	-	-	-	-	-	-
Hermans *et al.* [7]	-	-	-	-	-	-	-	0.38	-	0.38
Couprie *et al.* [3]	-	-	-	-	-	-	-	-	172800	0.70
Proposed	**143**	**6.6**	**20**	**0.27**	**2**	**0.07**	**133**	**0.32**	**183**	**0.26**

5.2 Competitive Classification and Segmentation Results

In Table 3 we compare the classification and segmentation accuracies to Shotton *et al.* [14], Fröhlich *et al.* [5], Hermans *et al.* [7] and Couprie *et al.* [3]. To obtain a smooth segmentation result we minimize the following energy [2]:

$$E(\Omega_1, .., \Omega_C) = \sum_{c=1}^{C} \Big(\mathrm{Per}\,(\Omega_c) + \lambda \int_{\Omega_c} f_c\,(p)\ \ dp \Big), \tag{6}$$

where $\Omega_1, .., \Omega_C$ denote the partitions of the image plane, $\mathrm{Per}\,(\Omega_c)$ the perimeter of each set Ω_c which is minimized to favor segments of shorter boundary and $f_c\,(p) = -\log \widetilde{P}(c \mid p, \mathcal{F}_R(n^*))$ the data term, where $\widetilde{P}$ are the class probabilities estimated with the proposed method. λ is a weighting parameter and optimized during the computation.

Table 3 indicates that our classification and segmentation accuracies are competitive with the state-of-the-art approaches. For each benchmark, our method achieves the best accuracies at a remarkably speeded up runtime (cf. Table 2).

Table 3. Quantitative results compared in terms of accuracies. The accuracies are computed as the percentage of correctly labeled pixels on the test set. At significantly reduced runtime our method achieves competitive classification and segmentation accuracies with state-of-the-art methods.

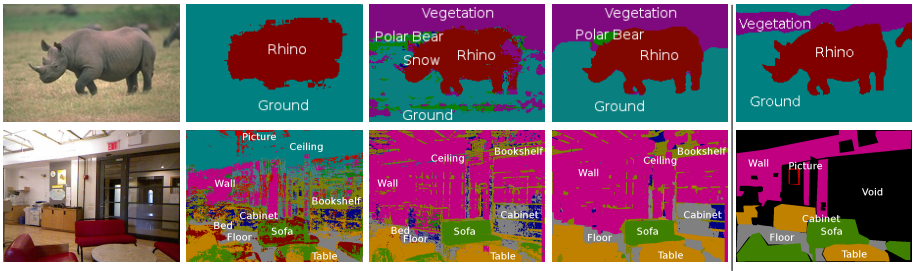
	eTrims		Corel		Sowerby		NYUv1		NYUv2	
	Class.	Segm.	Class.	Segm.	Class.	Segm.	Class.	Segm.	Class.	Segm.
Shotton *et al.* [14]	-	-	68.4	74.6	85.6	88.6	-	-	-	-
Fröhlich *et al.* [5]	-	77.22	-	-	-	-	-	-	-	-
Hermans *et al.* [7]	-	-	-	-	-	-	**65.0**	**71.5**	-	**54.2**
Couprie *et al.* [3]	-	-	-	-	-	-	-	-	-	52.4
Proposed	**77.1**	**77.9**	**74.4**	**78.2**	**87.1**	**88.8**	**65.0**	66.5	**44.0**	45.0

Most importantly our scores are a) obtained at a significantly improved runtime and b) by using an automatically chosen feature set. We neither tuned the parameters nor the feature set manually to obtain better scores on the specific benchmarks. The proposed method is designed to autonomously compute accurate classifications/segmentations at a significantly reduced runtime for all benchmarks.

Figure 3 shows exemplary qualitative classification and segmentation results obtained with the proposed method. In column b), we additionally provide the classification results of the related methods.

6 Conclusion

We introduced a framework for automatic feature selection for semantic image segmentation. Starting from a large set of popular features, we sequentially construct a ranked set of features by maximizing the relevance of each feature for the classification task while minimizing its redundancy with respect to the previously selected features. Subsequently, we define an automatic criterion to choose a small number of the most significant features. Integrated in a variational approach to multi-region segmentation, we obtain a fully automatic algorithm which provides state-of-the-art semantic classifications and segmentations on five popular benchmarks at drastically reduced computation time.



a) Original image | b) Classification of [14]/[7] | c) Proposed Classification | d) Proposed Segmentation | e) Ground Truth

Fig. 3. Accurate qualitative classification and segmentation results are achieved with the proposed framework. We compare our classification result to Shotton *et al.* [14] on the Corel benchmark (first row) and to Hermans *et al.* [7] on the NYUv1 benchmark (second row).

References

1. J. Bergbauer, C. Nieuwenhuis, M. Souiai, and D. Cremers. Proximity priors for variational semantic segmentation and recognition. In *ICCV Workshop*, 2013.
2. A. Chambolle, D. Cremers, and T. Pock. A convex approach to minimal partitions. *SIAM Journal on Imaging Sciences*, 5(4):1113–1158, 2012.
3. C. Couprie, C. Farabet, L. Najman, and Y. LeCun. Indoor semantic segmentation using depth information. In *Int. Conf. on Learning Representations*, 2013.
4. N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Int. Conf. on Comp. Vision and Pattern Recog.*, 2005.
5. B. Fröhlich, E. Rodner, and J. Denzler. Semantic Segmentation with Millions of Features: Integrating Multiple Cues in a Combined Random Forest Approach. In *Asian Conf. on Comp. Vision*, 2012.
6. X. He, R.S. Zemel, and M.A. Carreira-Perpindn. Multiscale conditional random fields for image labeling. In *Int. Conf. on Comp. Vision and Pattern Recog.*, 2004.
7. A. Hermans, G. Floros, and B. Leibe. Dense 3D Semantic Mapping of Indoor Scenes from RGB-D Images. In *Int. Conf. on Robotics and Automation*, 2014.
8. F. Korč and W. Förstner. eTRIMS Image Database for Interpreting Images of Man-Made Scenes. Technical report, Department of Photogrammetry, University of Bonn, 2009.
9. Lubor Ladicky, Chris Russell, Pushmeet Kohli, and Philip H. S. Torr. Graph cut based inference with co-occurrence statistics. In *Europ. Conf. on Comp. Vision*, 2010.
10. D. Lowe. Object recognition from local scale-invariant features. In *Int. Conf. on Comp. Vision*, 1999.
11. C. Nieuwenhuis, E. Strekalovskiy, and D. Cremers. Proportion priors for image sequence segmentation. In *Int. Conf. on Comp. Vision*, 2013.
12. H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *Trans. on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.
13. J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Europ. Conf. on Comp. Vision*. Springer, 2006.
14. J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context. *Int. Journal of Comp. Vision*, 81(1):2–23, 2009.
15. N. Silberman and R. Fergus. Indoor Scene Segmentation using a Structured Light Sensor. In *ICCV Workshop on 3D Representation and Recognition*, 2011.
16. N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *Europ. Conf. on Comp. Vision*. Springer, 2012.
17. M. Souiai, C. Nieuwenhuis, E. Strekalovskiy, and D. Cremers. Convex optimization for scene understanding. In *ICCV Workshop*, 2013.
18. M. Souiai, E. Strekalovskiy, C. Nieuwenhuis, and D. Cremers. A co-occurrence prior for continuous multi-label optimization. In *Int. Conf. on Energy Minimization Methods for Comp. Vision and Pattern Recog.*, 2013.
19. P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Int. Conf. on Comp. Vision and Pattern Recog.*, 2001.
20. Jian Yao, Sanja Fidler, and Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *Int. Conf. on Comp. Vision and Pattern Recog.*, 2012.