Linear Span Network for Object Skeleton Detection

Chang Liu, Wei Ke, Fei Qin, and Qixiang Ye(✉)

University of Chinese Academy of Sciences, Beijing, China
{liuchang615,kewei11}@mails.ucas.ac.cn, {fqin1982,qxye}@ucas.ac.cn

Abstract. Robust object skeleton detection requires to explore rich representative visual features and effective feature fusion strategies. In this paper, we first re-visit the implementation of HED, the essential principle of which can be ideally described with a linear reconstruction model. Hinted by this, we formalize a Linear Span framework, and propose Linear Span Network (LSN) which introduces Linear Span Units (LSUs) to minimizes the reconstruction error. LSN further utilizes subspace linear span besides the feature linear span to increase the independence of convolutional features and the efficiency of feature integration, which enhances the capability of fitting complex ground-truth. As a result, LSN can effectively suppress the cluttered backgrounds and reconstruct object skeletons. Experimental results validate the state-of-the-art performance of the proposed LSN.

Keywords: Linear span framework · Linear span unit
Linear span network · Skeleton detection

1 Introduction

Skeleton is one of the most representative visual properties, which describes objects with compact but informative curves. Such curves constitute a continuous decomposition of object shapes [13], providing valuable cues for both object representation and recognition. Object skeletons can be converted into descriptive features and spatial constraints, which enforce human pose estimation [22], semantic segmentation [20], and object localization [8].

Researchers have been exploiting the representative CNNs for skeleton detection and extraction [5,17,18,24] for years. State-of-the-art approaches root in effective multi-layer feature fusion, with the motivation that low-level features focus on detailed structures while high-level features are rich in semantics [5]. As a pioneer work, the holistically-nested edge detection (HED) [24] is computed as a pixel-wise classification problem, without considering the complementary

C. Liu and W. Ke—Equal contributions.

The source code is publicly available at https://github.com/LinearSpanNetwork.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11206, pp. 136–151, 2018.
https://doi.org/10.1007/978-3-030-01216-8_9

among multi-layer features. Other state-of-the-art approaches, *e.g.*, fusing scale-associated deep side-outputs (FSDS) [17,18] and side-output residual network (SRN) [5] investigates the multi-layer association problem. FSDS requires intensive annotations of the scales for each skeleton point, while SRN struggles to pursuits the complementary between adjacent layers without complete mathematical explanation. The problem of how to principally explore and fuse more representative features remains to be further elaborated.

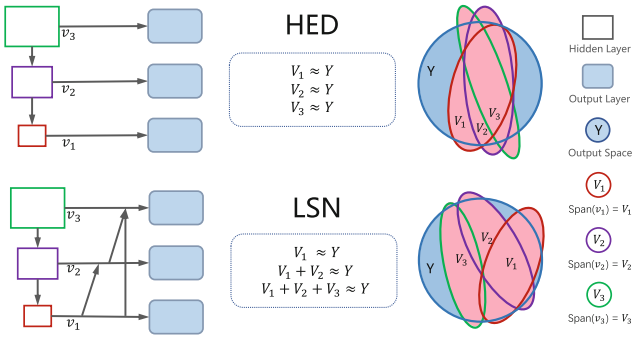


Fig. 1. A comparison of holistically-nested edge detection (HED) network [24] and linear span network (LSN). HED uses convolutional features without considering their complementary. The union of the output spaces of HED is small, denoted as the pink area. As an improved solution, LSN spans a large output space.

Through the analysis, it is revealed that HED treats the skeleton detection as a pixel-wise classification problem with the side-output from convolutional network. Mathematically, this architecture can be equalized with a linear reconstruction model, by treating the convolutional feature maps as linear bases and the 1×1 convolutional kernel values as weights. Under the guidance of the linear span theory [6], we formalize a linear span framework for object skeleton detection. With this framework, the output spaces of HED could have intersections since it fails to optimize the subspace constrained by each other, Fig. 1. To ease this problem, we design Linear Span Unit (LSU) according to this framework, which will be utilized to modify convolutional network. The obtained network is named as Linear Span Network (LSN), which consists feature linear span, resolution alignment, and subspace linear span. This architecture will increase the independence of convolutional features and the efficiency of feature integration, which is shown as the smaller intersections and the larger union set, Fig. 1. Consequently, the capability of fitting complex ground-truth could be enhanced. By stacking multiple LSUs in a deep-to-shallow manner, LSN captures both rich object context and high-resolution details to suppress the cluttered backgrounds and reconstruct object skeletons. The contributions of the paper include:

– A linear span framework that reveals the essential nature of object skeleton detection problem, and proposes that the potential performance gain could

be achieved with both the increased independence of spanning sets and the enlarged spanned output space.
– A Linear Span Network (LSN) can evolve toward the optimized architecture for object skeleton detection under the guidance of linear span framework.

2 Related Work

Early skeleton extraction methods treat skeleton detection as morphological operations [7,9,11,12,14,23,25]. One hypothesis is that object skeletons are the subsets of lines connecting center points of super-pixels [9]. Such line subsets could be explored from super-pixels using a sequence of deformable discs to extract the skeleton path [7]. In [23], The consistence and smoothness of skeleton are modeled with spatial filters, *e.g.*, a particle filter, which links local skeleton segments into continuous curves. Recently, learning based methods are utilized for skeleton detection. It is solved with a multiple instance learning approach [21], which picks up a true skeleton pixel from a bag of pixels. The structured random forest is employed to capture diversity of skeleton patterns [20], which can be also modeled with a subspace multiple instance learning method [15].

With the rise of deep learning, researchers have recently formulated skeleton detection as image-to-mask classification problem by using learned weights to fuse the multi-layer convolutional features in an end-to-end manner. HED [24] learns a pixel-wise classifier to produce edges, which can be also used for skeleton detection. Fusing scale-associated deep side-outputs (FSDS) [18] learns multi-scale skeleton representation given scale-associated ground-truth. Side-output residual network (SRN) [5] leverages the output residual units to fit the errors between the object symmetry/skeleton ground-truth and the side-outputs of multiple convolutional layers.

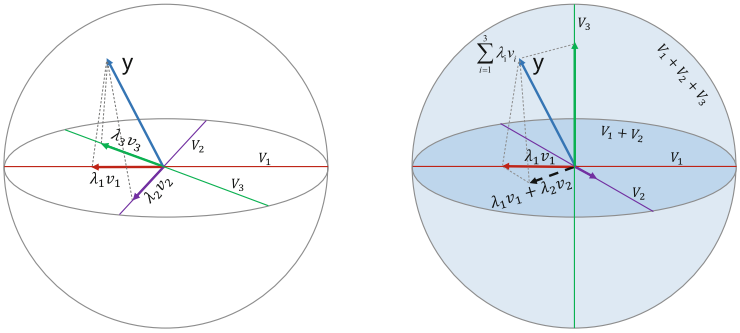
The problem about how to fuse multi-layer convolutional features to generate an output mask, *e.g.*, object skeleton, has been extensively explored. Nevertheless, existing approaches barely investigate the problem about the linear independence of multi-layer features, which limits their representative capacity. Our approach targets at exploring this problem from the perspective of linear span theory by feature linear span of multi-layer features and subspace linear span of the spanned subspaces.

3 Problem Formulation

3.1 Re-thinking HED

In this paper, we re-visit the implementation of HED, and reveal that HED as well as its variations can be all formulated by the linear span theory [6].

HED utilizes fully convolutional network with deep supervision for edge detection, which is one of the typical low-level image-to-mask task. Denoting



(a) dependent linear span (b) independent linear span

Fig. 2. Schematic of linear span with a set of dependent vectors (a) and independent vectors (b).

the convolutional feature as C with m maps and the classifier as w, HED is computed as a pixel-wise classification problem, as

$$\hat{y_j} = \sum_{k=1}^{m} w_k \cdot c_{k,j}, j = 1, 2, \cdots, |\hat{Y}|, \tag{1}$$

where $c_{k,j}$ is the feature value of the j-th pixel on the k-th convolutional map and $\hat{y}_j$ is the classified label of the j-th pixel in the output image $\hat{Y}$.

Not surprisingly, this can be equalized as a linear reconstruction problem, as

$$Y = \sum_{k=1}^{m} \lambda_k v_k, \tag{2}$$

where λ_k is linear reconstruction weight and v_k is the k-th feature map in C.

We treat each side-output of HED as a feature vector in the linear spanned subspace $V_i = span(v_1^i, v_2^i, \cdots, v_m^i)$, in which i is the index of convolutional stages. Then HED forces each subspace V_i to approximate the ground-truth space $\mathcal{Y}$. We use three convolutional layers as an example, which generate subspaces V_1, V_2, and V_3. Then the relationship between the subspaces and the ground-truth space can be illustrated as lines in a 3-dimension space in Fig. 2(a).

As HED does not optimize the subspaces constrained by each other, it fails to explore the complementary of each subspace to make them decorrelated. The reconstructions can be formulated as

$$\begin{cases} V_1 \approx \mathcal{Y} \\ V_2 \approx \mathcal{Y} \\ V_3 \approx \mathcal{Y} \end{cases}. \tag{3}$$

When v_1, v_2, and v_3 are linearly dependent, they only have the capability to reconstruct vectors in a plane. That is to say, when the point Y is out of the plane, the reconstruction error is hardly eliminated, Fig. 2(a).

Obviously, if v_1, v_2, and v_3 are linearly independent, *i.e.*, not in the same plane, Fig. 2(b), the reconstruction could be significantly eased. To achieve this target, we can iteratively formulate the reconstruction as

$$\begin{cases} V_1 \approx \mathcal{Y} \\ V_1 + V_2 \approx \mathcal{Y} \\ V_1 + V_2 + V_3 \approx \mathcal{Y} \end{cases}. \tag{4}$$

It s observed that V_2 is refined with the constraint of V_1. And V_3 is optimized in the similar way, which aims for vector decorrelation. The sum of subspaces, *i.e.*, $V_1 + V_2$ is denoted with the dark blue plane, and $V_1 + V_2 + V_3$ is denoted with the light blue sphere, Fig. 2(b).

Now, it is very straightforward to generalize Eq. (4) to

$$\sum_{k=1}^{l} V_k \approx \mathcal{Y}, l = 1, 2, \cdots, n. \tag{5}$$

One of the variations of HED, *i.e.*, SRN, which can be understand as a special case of Eq. (5) with $\sum_{k=l-1}^{l} V_k \approx \mathcal{Y}$, has already shown the effectiveness.

3.2 Linear Span View

Based on the discussion of last section, we can now strictly formulate a mathematical framework based on linear span theory [6], which can be utilized to guide the design of Linear Span Network (LSN) toward the optimized architecture.

In linear algebra, linear span is defined as a procedure to construct a linear space by a set of vectors or a set of subspaces.

Definition 1. $\mathcal{Y}$ is a linear space over $\mathbb{R}$. The set $\{v_1, v_2, \ldots, v_m\} \subset \mathcal{Y}$ is a spanning set for $\mathcal{Y}$ if every y in $\mathcal{Y}$ can be expressed as a linear combination of $v_1, v_2, \ldots, v_m$, as

$$y = \sum_{k=1}^{m} \lambda_k v_k, \ \lambda_1, \ldots, \lambda_m \in \mathbb{R}, \tag{6}$$

and $\mathcal{Y} = span(\{v_1, v_2, \ldots, v_m\})$.

Theorem 1. Let $v_1, v_2, \ldots, v_m$ be vectors in $\mathcal{Y}$. Then $\{v_1, v_2, \ldots, v_m\}$ spans $\mathcal{Y}$ if and only if, for the matrix $F = [v_1, v_2, \ldots, v_m]$, the linear system $F\lambda = y$ is consistent for every y in $\mathcal{Y}$.

Remark 1. According to *Theorem* 1, if the linear system is consistent for almost every vector in a linear space, the space can be approximated by the linear spanned space. This theorem uncovers the principle of LSN, which pursues a linear system as mentioned above setting up for as many as ground-truth.

Definition 2. A finite set of vectors, which span $\mathcal{Y}$ and are linearly independent, is called a basis for $\mathcal{Y}$.

Theorem 2. Every linearly independent set of vectors $\{v_1, v_2, \ldots, v_m\}$ in a finite dimensional linear space $\mathcal{Y}$ can be completed to a basis of $\mathcal{Y}$.

Theorem 3. Every subspace U has a complement in $\mathcal{Y}$, that is, another subspace V such that vector y in $\mathcal{Y}$ can be decomposed uniquely as

$$y = u + v, u \ in \ U, v \ in \ V. \tag{7}$$

Definition 3. $\mathcal{Y}$ is said to be the sum of its subspaces $V_1, \ldots, V_m$ if every y in Y can be expressed as

$$y = v_1 + \ldots + v_m, v_j \ in \ V_j. \tag{8}$$

Remark 2. We call the spanning of feature maps to a subspace as feature linear span, and the sum of subspaces as subspace linear span. From *Theorems* 2 and 3, it is declared that the union of the spanning sets of subspaces is the spanning set of the sum of the subspaces. That is to say, in the subspace linear span we can merge the spanning sets of subspaces step by step to construct a larger space.

Theorem 4. Supposing $\mathcal{Y}$ is a finite dimensional linear space, U and V are two subspaces of $\mathcal{Y}$ such that $\mathcal{Y} = U + V$, and W is the intersection of U and V, *i.e.*, $W = U \cap V$. Then

$$\dim \mathcal{Y} = \dim U + \dim V - \dim W. \tag{9}$$

Remark 3. From *Theorem* 4, the smaller the dimension of the intersection of two subspaces is, the bigger the dimension of the sum of two subspaces is. Then, successively spanning the subspaces from deep to shallow with supervision increases independence of spanning sets and enlarges the sum of subspaces. It enfores the representative capacity of convolutional features and integrates them in a more effective way.

4 Linear Span Network

With the help of the proposed framework, the Linear Span Network (LSN) is designed for the same targets with HED and SRN, *i.e.*, the object skeleton detection problem. Following the linear reconstruction theory, a novel architecture named Linear Span Unit (LSU) has been defined first. Then, LSN is updated from VGG-16 [17] with LSU and hints from *Remarks* 1–3. VGG-16 has been chosen for the purpose of fair comparison with HED and SRN. In what follows, the implementation of LSU and LSN are introduced.

4.1 Linear Span Unit

The architecture of Linear Span Unit (LSU) is shown in Fig. 3, where each feature map is regarded as a feature vector. The input feature vectors are unified with a concatenation (concat for short) operation, as

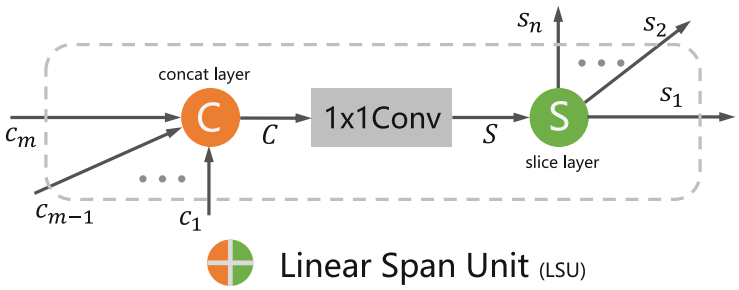


Fig. 3. Linear Span Unit, which is used in both feature linear span and subspace linear span. In LSU, the operation of linear reconstruction is implemented by a concatenation layer and a 1×1 convolutional layer.

$$C = \underset{k=1}{\overset{m}{concat}}(c_k), \tag{10}$$

where c_k is the k-th feature vector. In order to compute the linear combination of the feature vectors, a convolution operation with $1 \times 1 \times m$ convolutional kernels is employed:

$$s_i = \sum_{k=1}^{m} \lambda_{k,i} \cdot c_k, i = 1, 2, \cdots, n, \tag{11}$$

where $\lambda_{k,i}$ is the convolutional parameter with k elements for the i-th reconstruction output. The LSU will generate n feature vectors in the subspace spanned by the input feature vectors. A slice layer is further utilized to separate them for different connections, which is denoted as

$$\bigcup_{i=1}^{n} s_i = slice(\mathrm{S}). \tag{12}$$

4.2 Linear Span Network Architecture

The architecture of LSN is shown in Fig. 4, which is consisted of three components, *i.e.*, feature linear span, resolution alignment, and subspace linear span are illustrated. The VGG-16 network with 5 convolutional stages [19] is used as the backbone network.

In feature linear span, LSU is used to span the convolutional feature of the last layer of each stage according to Eq. 11. The supervision is added to the output of LSU so that the spanned subspace approximates the ground-truth space, following *Remark* 1. If only feature linear span is utilized, the LSN is degraded to HED [24]. Nevertheless, the subspaces in HED separately fit the ground-truth space, and thus fail to decorrelate spanning sets among subspaces. According to *Remarks* 2 and 3, we propose to further employ subspace linear span to enlarge the sum of subspaces and deal with the decorrelation problem.

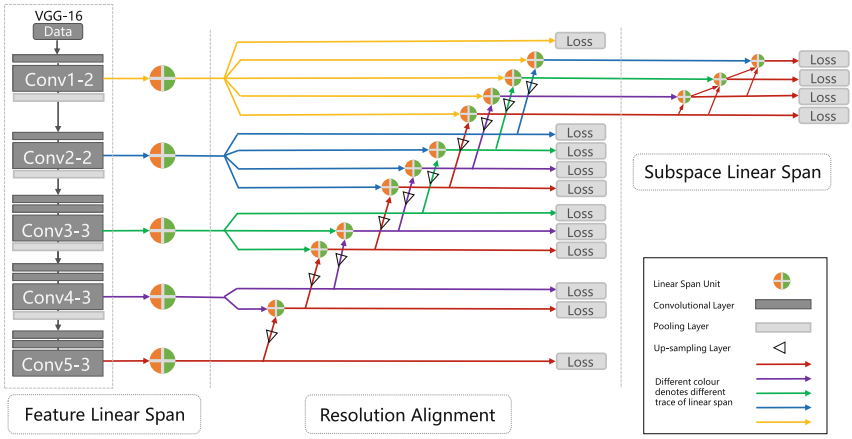


Fig. 4. The architecture of the proposed Linear Span Network (LSN), which leverages Linear Span Units (LSUs) to implement three components of the feature linear span, the resolution alignment, and the subspace linear span. The feature linear span uses convolutional features to build subspaces. The LSU is re-used to unify the resolution among multi-stages in resolution alignment. The subspace linear span summarizes the subspaces to fit the ground-truth space.

As the resolution of the vectors in different subspaces is with large variation, simple up-sampling operation will cause the Mosaic effect, which generates noise in subspace linear span. Without any doubt, the resolution alignment is necessary for LSN. Thus, in Fig. 4, LSUs have been laid between any two adjacent layers with supervision. As a pre-processing component to subspace linear span, it outputs feature vectors with same resolution.

The subspace linear span is also implemented by LSUs, which further concatenates feature vectors from deep to shallow layers and spans the subspaces with Eq. (5). According to *Remark* 3, a step-by-step strategy is utilized to explore the complementary of subspaces. With the loss layers attached on LSUs, it not only enlarges the sum of subspaces spanned by different convolutional layers, but also decorrelates the union of spanning sets of different subspaces. With this architecture, LSN enforces the representative capacity of convolutional features to fit complex ground-truth.

5 Experiments

5.1 Experimental Setting

Datasets: We evaluate the proposed LSN on pubic skeleton datasets including SYMMAX [21], WH-SYMMAX [15], SK-SMALL [18], SK-LARGE [17], and Sym-PASCAL [5]. We also evaluate LSN to edge detection on the BSDS500 dataset [1] to validate its generality.

SYMMAX is derived from BSDS300 [1], which contains 200/100 training and testing images. It is annotated with local skeleton on both foreground and background. WH-SYMMAX is developed for object skeleton detection, but contains only cropped horse images, which are not comprehensive for general object skeleton. SK-SMALL involves skeletons about 16 classes of objects with 300/206 training and testing images. Based on SK-SMALL, SK-LARGE is extended to 746/745 training and testing images. Sym-PASCAL is derived from the PASCAL-VOC-2011 segmentation dataset [4] which contains 14 object classes with 648/787 images for training and testing.

The BSDS500 [1] dataset is used to evaluate LSN's performance on edge detection. This dataset is composed of 200 training images, 100 validation images, and 200 testing images. Each image is manually annotated by five persons on average. For training images, we preserve their positive labels annotated by at least three human annotators.

Evaluation Protocol: Precision recall curve (PR-curve) is use to evaluate the performance of the detection methods. With different threshold values, the output skeleton/edge masks are binarized. By comparing the masks with the ground-truth, the precision and recall are computed. For skeleton detection, the F-measure is used to evaluate the performance of the different detection approaches, which is achieved with the optimal threshold values over the whole dataset, as

$$F = \frac{2PR}{P+R}. \tag{13}$$

To evaluate edge detection performance, we utilize three standard measures [1]: F-measures when choosing an optimal scale for the entire dataset (ODS) or per image (OIS), and the average precision (AP).

Hyper-Parameters: For both skeleton and edge detection, we use VGG16 [19] as the backbone network. During learning we set the mini-batch size to 1, the loss-weight to 1 for each output layer, the momentum to 0.9, the weight decay to 0.002, and the initial learning rate to 1e-6, which decreases one magnitude for every 10,000 iterations.

5.2 LSN Implementation

We evaluate four LSN architectures for subspace linear span and validate the iterative training strategy.

LSN Architectures. If there is no subspace linear span, Fig. 4, LSN is simplified to HED [24], which is denoted as LSN_1. The F-measure of LSN_1 is 49.53%. When the adjacent two subspaces are spanned, it is denoted as LSN_2, which is the same as SRN [5]. LSN_2 achieve significant performance improvement over HED which has feature linear span but no subspace span. We compare LSNs with different number of subspaces to be spanned, and achieve the best F-measure of 66.82%. When the subspace number is increased to 4, the skeleton detection performance drops. The followings explained why LSN_3 is the best choice.

If the subspaces to be spanned are not enough, the complementary of convolutional features from different layers could not be effectively explored. On the contrary, if a LSU fuses feature layers that have large-scale resolution difference, it requires to use multiple up-sampling operations, which deteriorate the features. Although resolution alignment significantly eases the problem, the number of adjacent feature layers to be fused in LSU remains a practical choice. LSN_3 reported the best performance by fusing a adjacent layer of higher resolution and a adjacent layer of lower resolution. On one hand, the group of subspaces in LSN_3 uses more feature integration. On the other hand, there is not so much information loss after an 2× up-sampling operation (Table 1).

Table 1. The performance of different LSN implementations on the SK-LARGE dataset. LSN_3 that fuses an adjacent layer of higher resolution and an adjacent layer of lower resolution reported the best performance.

Architecture	F-measure (%)
LSN_1 (HED, feature linear span only)	49.53
LSN_2 (SRN, feature and 2-subspace linear span)	65.88
LSN_3 (LSN, feature and 3-subspace linear span)	**66.15**
LSN_4 (LSN, feature and 4-subspace linear span)	65.89

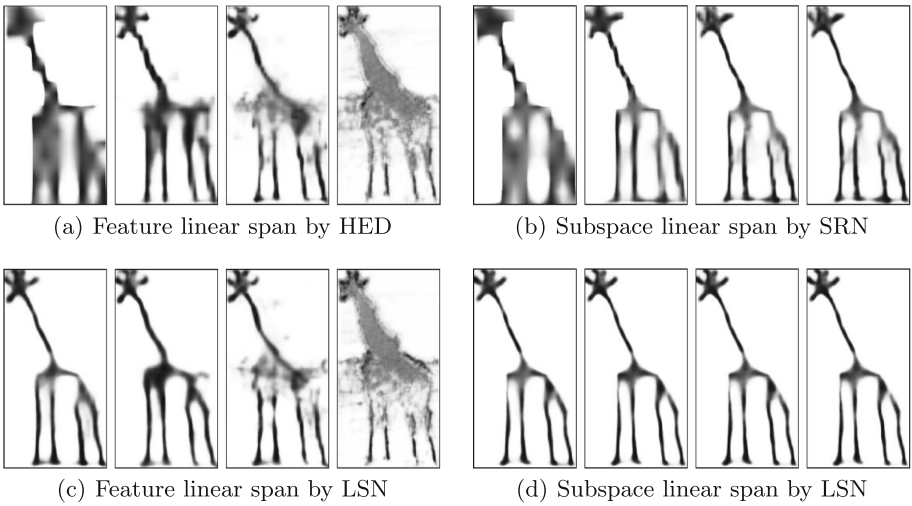
Table 2. The performance for different training strategies.

	w/o RA	end-to-end	iter1	iter2	iter3
F-measure(%)	66.15	66.63	**66.82**	66.74	66.68

Training Strategy. With three feature layers spanned, LSN needs up-sampling the side-output feature layers from the deepest to the shallowest ones. We use the supervised up-sampling to unify the resolution of feature layers.

During training, the resolution alignment is also achieved by stacking LSUs. We propose a strategy that train the two kinds of linear span, *i.e.*, feature linear span with resolution alignment and subspace linear span, iteratively. In the first iteration, we tune the LSU parameters for feature linear span and resolution alignment using the pre-trained VGG model on ImageNet, as well as update the convolutional parameters. Keeping the LSU parameters for resolution alignment unchanged, we tune LSU parameters for feature linear span and subspace linear span using the new model. In other iteration, the model is fine-tuned on the snap-shot of the previous iteration. With this training strategy, the skeleton detection performance is improved from 66.15% to 66.82%, Table 2. The detection performance changes marginally when more iterations are used. We therefore use the single iteration (iter1) in all experiments.

LSU Effect. In Fig. 5, we use a giraffe's skeleton from SK-LARGE as an example to compare and analyze the learned feature vectors (bases) by HED [24], SRN [24], and LSN. In Fig. 5(a) and (c), we respectively visualize the feature vectors learned by HED [24] and the proposed LSN. It can be seen in the first column that the HED's results incorporate more background noise and mosaic effects. This shows that the proposed LSN can better span an output feature space. In Fig. 5(b) and (d), we respectively visualize the subspace vectors learned by SRN [5] and the proposed LSN. It can be seen in the first column that the SRN's results incorporate more background noises. It requires to depress such noises by using a residual reconstruction procedure. In contrast, the subspace vectors of LSN is much clearer and compacter. This fully demonstrates that LSN can better span the output space and enforce the representative capacity of convolutional features, which will ease the problems of fitting complex outputs with limited convolutional layers (Fig. 6).



(a) Feature linear span by HED (b) Subspace linear span by SRN

(c) Feature linear span by LSN (d) Subspace linear span by LSN

Fig. 5. Comparison of output feature vectors of HED [24], SRN [5], and LSN(From left to right results are listed in a deep-to-shallow manner). By comparing (a) and (c), (b) and (d), one can see that LSN can learn better feature vectors and subspaces(basis) to span the output space. It enforces the representative capacity of convolutional features to fit complex outputs with limited convolutional layers.

5.3 Performance and Comparison

Skeleton Detection. The proposed LSN is evaluated and compared with the state-of-the-art approaches, and the performance is shown in Fig. 5 and Table 3. The result of SRN [5] is obtained by running authors' source code on a Tesla K80 GPU, and the other results are provided by [17].

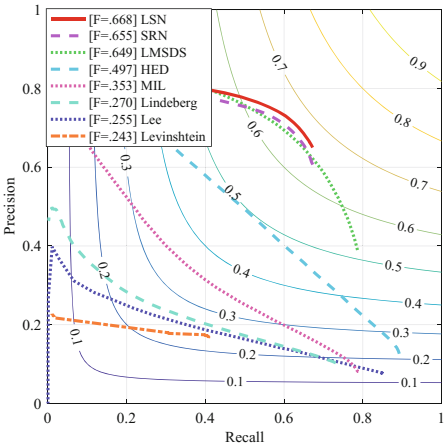


Fig. 6. The PR-curve on SK-LARGE.

Table 3. Performance comparison on SK-LARGE dataset. † GPU time.

Mehods	F-measure	Runtime/s
Lindeberg [11]	0.270	4.05
Levinshtein [9]	0.243	146.21
Lee [7]	0.255	609.10
MIL [21]	0.293	42.40
HED [24]	0.495	**0.05**†
SRN [5]	0.655	0.08†
LMSDS [17]	0.649	**0.05**†
LSN (ours)	**0.668**	0.09†

The conventional approaches including Lindeberg [11], Levinshtein [9], and Lee [7], produce the skeleton masks without using any learning strategy. They are time consuming and achieve very low F-measure of 27.0%, 24.3%, and 25.5%, respectively. The typical learning approach, *i.e.*, multiple instance learning (MIL) [21], achieves F-measure of 29.3%. It extractes pixel-wised feature with multi-orientation and multi-scale, and averagely uses 42.40 s to distinguish skeleton pixels from the backgrounds in a single image.

The CNN based approaches achieve huge performance gain compared with the conventional approaches. HED [24] achieves the F-measure of 49.5% and uses 0.05 s to process an images, while SRN [5] achieves 64.9% and uses 0.08 s. The scale-associated multi-task method, LMSDS [17], achieves the performance of 64.9%, which is built on HED with the pixel-level scale annotations. Our proposed LSN reportes the best detection performance of 66.8% with a little more runtime cost compared with HED and SRN.

The results show that feature linear span is efficient for skeleton detection. As discussed above, HED and SRN are two special case of LSN. LSN that used three spanned layers in each span unit is a better choice than the state-of-the art SRN. Some skeleton detection results are shown in Fig. 7. It is illustrated that HED produces lots of noise while the FSDS is not smooth. Comparing SRN with LSN, one can see that LSN rectifies some false positives as shown in column one and column three and reconstruct the dismiss as shown in column six.

The proposed LSN is also evaluated on other four commonly used datasets, including WH-SYMMAX [15], SK-SMALL [18], SYMMAX [21], and Sym-PASCAL [5]. The F-measure are shown in Table 4. Similar with SK-LARGE, LSN achieves the best detection performance on WH-SYMMAX, SK-SMALL, and SYMMAX, with the F-measure 79.7%, 63.3% and 48.0%. It achieves 5.4%, 8.1%, and 5.3% performance gain compared with HED, and 1.7%, 0.1%, and

Fig. 7. Skeleton detection examples by state-of-the-art approaches including HED [24], FSDS [18], SRN [5], and LSN. The red boxes are false positive or dismiss in SRN, while the blue ones are correct reconstruction skeletons in LSN at the same position. (Best viewed in color with zoon-in.)

Table 4. Performance comparison of the state-of-the-art approaches on the public WH-SYMMAX [15], SK-SMALL [18], SYMMAX [21], and Sym-PASCAL [5] datasets.

	WH-SYMMAX	SK-SMALL	SYMMAX	Sym-PASCAL
Levinshtein [9]	0.174	0.217	–	0.134
Lee [7]	0.223	0.252	–	0.135
Lindeberg [11]	0.277	0.227	0.360	0.138
Particle Filter [23]	0.334	0.226	–	0.129
MIL [21]	0.365	0.392	0.362	0.174
HED [24]	0.743	0.542	0.427	0.369
FSDS [18]	0.769	0.623	0.467	0.418
SRN [5]	0.780	0.632	0.446	**0.443**
LSN (ours)	**0.797**	**0.633**	**0.480**	0.425

2.4% gain compared with SRN. On Sym-PASCAL, LSN achieves comparable performance of 42.5% vs. 44.3% with the state-of-the art SRN.

Edge Detection. Edge detection task has similar implementation with skeleton that discriminate whether a pixel belongs to an edge. It also can be reconstructed by the convolutional feature maps. In this section, we compare the edge detection

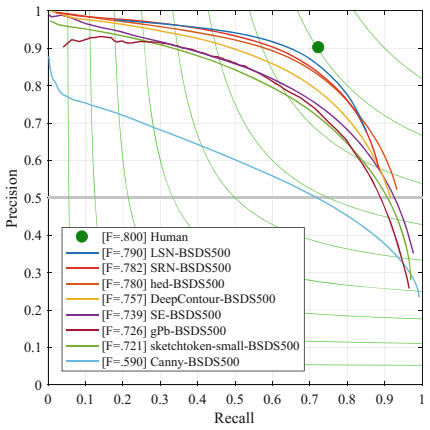


Fig. 8. The PR-curve on the BSDS500 edge detection dataset.

Table 5. Performance comparison on the BSDS500 edge detection dataset. † GPU time.

Mehods	ODS	OIS	AP	FPS
Canny [2]	0.590	0.620	00578	15
ST [10]	0.721	0.739	0.768	1
gPb [1]	0.726	0.760	0.727	1/240
SE [16]	0.739	0.759	0.792	2.5
DC [16]	0.757	0.776	0.790	1/30†
HED [24]	0.780	0.797	0.814	2.5†
SRN [5]	0.782	0.800	0.779	2.3†
LSN (ours)	**0.790**	**0.806**	0.618	2.0†
Human	0.800	0.800	–	–

result of the proposed LSN with some other state-of-the-art methods, such as Canny [2], Sketech Tokens [10], Structured Edge (SE) [3], gPb [1], DeepContour [16], HED [24], and SRN [5], Fig. 8 and Table 5.

In Fig. 8, it is illustrated that the best conventional approach is SE with F-measure (ODS) of 0.739 and all the CNN based approaches achieve much better detection performance. HED is one of the baseline deep learning method, which achieved 0.780. The proposed LSN reportes the highest F-measure of 0.790, which has a very small gap (0.01) to human performance. The F-measure with an optimal scale for the per image (OIS) was 0.806, which was even higher than human performance, Table 5. The good performance of the proposed LSN demonstrates its general applicability to image-to-mask tasks.

6 Conclusion

Skeleton is one of the most representative visual properties, which describes objects with compact but informative curves. In this paper, the skeleton detection problem is formulated as a linear reconstruction problem. Consequently, a generalized linear span framework for skeleton detection has been presented with formal mathematical definition. We explore the Linear Span Units (LSUs) to learn a CNN based mask reconstruction model. With LSUs we implement three components including feature linear span, resolution alignment, and subspace linear span, and update the Holistically-nested Edge Detection (HED) network to Linear Span Network (LSN). With feature linear span, the ground truth space can be approximated by the linear spanned output space. With subspace linear span, not only the independence among spanning sets of subspaces can be increased, but also the spanned output space can be enlarged. As a result, LSN will have better capability to approximate the ground truth

space, *i.e.*, against the cluttered background and scales. Experimental results validate the state-of-the-art performance of the proposed LSN, while we provide a principled way to learn more representative convolutional features.

Acknowledgement. This work was partially supported by the National Nature Science Foundation of China under Grant 61671427 and Grant 61771447, and Beijing Municipal Science & Technology Commission under Grant Z181100008918014.

References

1. Arbelaez, P., Maire, M., Fowlkes, C.C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2011)
2. Canny, J.F.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679–698 (1986)
3. Dollár, P., Zitnick, C.L.: Fast edge detection using structured forests. IEEE Trans. Pattern Anal. Mach. Intell. **37**(8), 1558–1570 (2015)
4. Everingham, M., Gool, L.J.V., Williams, C.K.I., Winn, J.M., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)
5. Ke, W., Chen, J., Jiao, J., Zhao, G., Ye, Q.: SRN: side-output residual network for object symmetry detection in the wild. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 302–310 (2017)
6. Lax, P.: Linear Algebra and Its Applications, vol. 2. Wiley, Hobken (2007)
7. Lee, T.S.H., Fidler, S., Dickinson, S.J.: Detecting curved symmetric parts using a deformable disc model. In: IEEE International Conference on Computer Vision, pp. 1753–1760 (2013)
8. Lee, T.S.H., Fidler, S., Dickinson, S.J.: Learning to combine mid-level cues for object proposal generation. In: IEEE International Conference on Computer Vision, pp. 1680–1688 (2015)
9. Levinshtein, A., Sminchisescu, C., Dickinson, S.J.: Multiscale symmetric part detection and grouping. Int. J. Comput. Vis. **104**(2), 117–134 (2013)
10. Lim, J.J., Zitnick, C.L., Dollár, P.: Sketch tokens: a learned mid-level representation for contour and object detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3158–3165 (2013)
11. Lindeberg, T.: Edge detection and ridge detection with automatic scale selection. Int. J. Comput. Vis. **30**(2), 117–156 (1998)
12. Saha, P.K., Borgefors, G., di Baja, G.S.: A survey on skeletonization algorithms and their applications. Pattern Recogn. Lett. **76**, 3–12 (2016)
13. Sebastian, T.B., Klein, P.N., Kimia, B.B.: Recognition of shapes by editing their shock graphs. IEEE Trans. Pattern Anal. Mach. Intell. **26**(5), 550–571 (2004)
14. Shen, W., Bai, X., Hu, R., Wang, H., Latecki, L.J.: Skeleton growing and pruning with bending potential ratio. Pattern Recogn. **44**(2), 196–209 (2011)
15. Shen, W., Bai, X., Hu, Z., Zhang, Z.: Multiple instance subspace learning via partial random projection tree for local reflection symmetry in natural images. Pattern Recogn. **52**, 306–316 (2016)
16. Shen, W., Wang, X., Wang, Y., Bai, X., Zhang, Z.: Deepcontour: a deep convolutional feature learned by positive-sharing loss for contour detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3982–3991 (2015)

17. Shen, W., Zhao, K., Jiang, Y., Wang, Y., Bai, X., Yuille, A.L.: DeepSkeleton: learning multi-task scale-associated deep side outputs for object skeleton extraction in natural images. IEEE Trans. Image Process. **26**(11), 5298–5311 (2017)
18. Shen, W., Zhao, K., Jiang, Y., Wang, Y., Zhang, Z., Bai, X.: Object skeleton extraction in natural images by fusing scale-associated deep side outputs. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 222–230 (2016)
19. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv abs/1409.1556 (2014)
20. Teo, C.L., Fermüller, C., Aloimonos, Y.: Detection and segmentation of 2D curved reflection symmetric structures. In: IEEE International Conference on Computer Vision, pp. 1644–1652 (2015)
21. Tsogkas, S., Kokkinos, I.: Learning-based symmetry detection in natural images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7578, pp. 41–54. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33786-4_4
22. Wei, S., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 4724–4732 (2016)
23. Widynski, N., Moevus, A., Mignotte, M.: Local symmetry detection in natural images using a particle filtering approach. IEEE Trans. Image Process. **23**(12), 5309–5322 (2014)
24. Xie, S., Tu, Z.: Holistically-nested edge detection. In: IEEE International Conference on Computer Vision5, pp. 1395–1403 (2015)
25. Yu, Z., Bajaj, C.L.: A segmentation-free approach for skeletonization of gray-scale images via anisotropic vector diffusion. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 415–420 (2004)