Journal of Visual Communication and Image Representation 50 (2018) 237–246

Contents lists available at ScienceDirect

Journal of Visual Communication and Image Representation

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

On data-driven Saak transform[☆]

C.-C. Jay Kuo[*], Yueru Chen

Ming-Hsieh Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089-2564, USA

ARTICLE INFO

Keywords:
Data-driven transform
RECOS transform
Saak transform
The Karhunen-Loéve transform (KLT)
Linear subspace approximation
Principal component analysis

ABSTRACT

Being motivated by the multilayer RECOS (REctified-COrrelations on a Sphere) transform, we develop a data-driven Saak (**S**ubspace **a**pproximation with **a**ugmented **k**ernels) transform in this work. The Saak transform consists of three steps: (1) building the optimal linear subspace approximation with orthonormal bases using the second-order statistics of input vectors, (2) augmenting each transform kernel with its negative, (3) applying the rectified linear unit (ReLU) to the transform output. The Karhunen-Loéve transform (KLT) is used in the first step. The integration of Steps 2 and 3 is powerful since they resolve the sign confusion problem, remove the rectification loss and allow a straightforward implementation of the inverse Saak transform at the same time. Multiple Saak transforms are cascaded to transform images of a larger size. All Saak transform kernels are derived from the second-order statistics of input random vectors in a one-pass feedforward manner. Neither data labels nor backpropagation is used in kernel determination. Multi-stage Saak transforms offer a family of joint spatial-spectral representations between two extremes; namely, the full spatial-domain representation and the full spectral-domain representation. We select Saak coefficients of higher discriminant power to form a feature vector for pattern recognition, and use the MNIST dataset classification problem as an illustrative example.

1. Introduction

Signal transforms provide a way to convert signals from one representation to another. For example, the Fourier transform maps a time-domain function into a set of Fourier coefficients. The latter representation indicates the projection of the time-domain function onto a set of orthonormal sinusoidal basis functions. The orthonormal basis facilitates the inverse transform. The original function can be synthesized by summing up all Fourier basis functions weighted by their Fourier coefficients. The basis functions (or transform kernels) are typically selected by humans. One exception is the Karhunen-Loéve transform (KLT) [1]. The KLT kernels are the unit eigenvectors of the covariance matrix of sampled data. It is the optimal transform in terms of energy compaction. That is, to obtain an approximation to an input signal class, we can truncate part of KLT basis functions associated with the smallest eigenvalues. The truncated KLT provides the optimal approximation to the input with the smallest mean-squared-error (MSE).

We develop new data-driven forward and inverse transforms in this work. For a set of images of size $N \times N$, the total number of variables in these images is N^2 and their covariance matrix is of dimension N^4. It is not practical to conduct the KLT on the full image for a large *N*. Instead, we may decompose images into smaller blocks and conduct the KLT on each block. To give an example, the Discrete Cosine Transform (DCT) [2] provides a good approximation to the KLT for image data, and the block DCT is widely used in the image/video compression standards. One question of interest is whether it is possible to generalize the KLT so that it can be applied to images of a larger size in a hierarchical fashion? Our second research motivation comes from the resurgent interest on convolutional neural networks (CNNs) [3,4]. The superior performance of CNNs has been demonstrated in many applications such as image classification, detection and processing. To offer an explanation, Kuo [5,6] modeled the convolutional operation at each CNN layer with the RECOS (REctified-COrrelations on a Sphere) transform, and interpreted the whole CNN as a multi-layer RECOS transform.

By following this line of thought, it would be a meaningful task to define the inverse RECOS transform and analyze its properties. The analysis of the forward/inverse RECOS transform will be conducted in Section 2. Being similar to the forward and inverse Fourier transforms, the forward and inverse RECOS transforms offer tools for signal analysis and synthesis, respectively. However, unlike the data-independent Fourier transform, the RECOS transform is derived from labeled training data, and its transform kernels (or filter weights) are optimized by backpropagation. The analysis of forward/inverse RECOS transforms is challenging due to nonlinearity. We will show that the RECOS transform has two loss terms: the approximation loss and the rectification loss. The approximation loss is caused by the use of a limited

[☆] This paper has been recommended for acceptance by Zicheng Liu.
[*] Corresponding author.
E-mail address: cckuo@sipi.usc.edu (C.-C. Jay Kuo).

https://doi.org/10.1016/j.jvcir.2017.11.023
Received 12 October 2017; Accepted 30 November 2017
Available online 05 December 2017

number of transform kernels. This error can be reduced by increasing the number of filters at the cost of higher computational complexity and higher storage memory. The rectification loss is due to nonlinear activation. Furthermore, since the filters in the RECOS transform are not orthogonal to each other, the inverse RECOS transform demands the solution of a linear system of equations.

It is stimulating to develop a new data-driven transform that has neither approximation loss nor the rectification loss as the RECOS transform. Besides, it has a set of orthonormal transform kernels so that its inverse transform can be performed in a straightforward manner. To achieve these objectives, we propose the Saak (**S**ubspace **a**pproximation with **a**ugmented **k**ernels) transform. As indicated by its name, the Saak transform has two main ingredients: (1) subspace approximation and (2) kernel augmentation. To seek the optimal subspace approximation to a set of random vectors, we analyze their second-order statistics and select orthonormal eigenvectors of the covariance matrix as transform kernels. This is the well-known KLT. When the dimension of the input space is very large (say, in the order of thousands or millions), it is difficult to conduct one-stage KLT. Then, we may decompose a high-dimensional vector into multiple lower-dimensional sub-vectors. This process can be repeated recursively to form a hierarchical representation. For example, we can decompose one image into four non-overlapping quadrants recursively to build a quad-tree whose leaf node is a small patch of size 2×2. Then, a KLT can be defined at each level of the quad-tree.

If two or more transforms are cascaded directly, there is a "sign confusion" problem [5,6]. To resolve it, we insert the Rectified Linear Unit (ReLU) activation function in between. The ReLU inevitably brings up the rectification loss, and a novel idea called kernel augmentation is proposed to eliminate this loss. That is, we augment each transform kernel with its negative vector, and use both original and augmented kernels in the Saak transform. When an input vector is projected onto the positive/negative kernel pair, one will go through the ReLU operation while the other will be blocked. This scheme greatly simplifies the signal representation problem in face of ReLU nonlinear activation. It also facilitates the inverse Saak transform. The integration of kernel augmentation and ReLU is equivalent to the sign-to-position (S/P) format conversion of the Saak transform outputs, which are called the Saak coefficients. By converting the KLT to the Saak transform stage by stage, we can cascade multiple Saak transforms to transform images of a large size. The multi-stage Saak transforms offer a family of joint spatial-spectral representations between two extremes – the full spatial-domain representation and the full spectral-domain representation. The Saak and multi-stage Saak transforms will be elaborated in Sections 3 and 4, respectively.

Although both CNNs and multi-stage Saak transforms adopt the ReLU, it is important to emphasize one fundamental difference in their filter weights (or transform kernels) determination. CNN's filter weights are determined by the training data and their associated labels. After initialization, these weights are updated via backpropagation. A CNN determines its optimal filter weights by optimizing a cost function via backpropagation iteratively. The iteration number is typically huge. The multi-stage Saak transforms adopt another fully automatic method in determining their transform kernels. They are selected based on the second-order statistics of input vectors at each stage. It is a one-pass feedforward process from the leaf to the root. Neither data labels nor backpropagation is needed for transform kernel determination.

The Saak coefficients in intermediate stages indicate the spectral component values in the corresponding covered spatial regions. The Saak coefficients in the last stage represent spectral component values of the entire input vector (i.e., the whole image). We use the MNIST dataset as an example to illustrate the distribution of Saak coefficients of each object class. Based on the ANalysis Of VAriance (ANOVA), we select Saak coefficients of higher discriminant power by computing their F-test score. The larger the F-test score, the higher the discriminant power. Finally, we compare the classification accuracy with the support vector machine (SVM) and the K-Nearest-Neighbors (KNN) classifiers.

The rest of this paper is organized as follows. The forward and inverse RECOS transforms are studied in Section 2. The forward and inverse Saak transforms are proposed in Section 3. The multi-stage Saak transforms are presented in Section 4. The application of multi-stage Saak transforms to image classification for the MNIST dataset is described in Section 5. The CNN approach and the Saak-transform-based machine learning methodology are compared in Section 6. Finally, concluding remarks are given and future research directions are pointed out in Section 7.

2. RECOS transform

The RECOS transform is a data-driven transform proposed in [5,6] to model the convolutional operations in a CNN. In the context of image processing, the forward RECOS transform defines a mapping from a real-valued function defined on a three-dimensional (3D) cuboid to a one-dimensional (1D) rectified spectral vector. The forward and inverse RECOS transforms will be studied in this section.

2.1. Forward RECOS transform

As illustrated in Fig. 1, a spatial-spectral cuboid, denoted by $C(i_p,j_p,L_i,L_j,L_k)$, consists of 3D grid points with indices (i,j,k)

- along the horizontal dimension: $i \in \{i_p,i_p+1,\dots,i_p+L_i-1\}$,
- along the vertical dimension: $j \in \{j_p,j_p+1,\dots,j_p+L_j-1\}$,
- along the spectral dimension: $k \in \{0,1,2,\dots L_k-1\}$,

where $(i,j)=(i_p,j_p)$ is its spatial pivot and L_i,L_j and L_k are its width, height and depth, respectively. For a real-valued function defined on cuboid $C(i_p,j_p,L_i,L_j,L_k)$, one can flatten these values into a one-dimensional (1D) vector,

$$\mathbf{f} \in R^N, \quad \text{where} \quad N = L_i \times L_j \times L_k, \tag{1}$$

by scanning the 3D grid points with a fixed order. All vectors in R^N are generated by the same flattening rule.

Consider an anchor vector set [5,6] that contains L_k vectors of unit length,

$$A = \{\mathbf{a}_0,\mathbf{a}_1,\dots,\mathbf{a}_k,\dots,\mathbf{a}_K\}, \quad \|\mathbf{a}_k\| = 1 \text{ and } K = L_k-1, \tag{2}$$

where $\mathbf{a}_k$ is defined on $C(i_p,j_p,L_i,L_j,L_k)$ and flattened to a vector in R^N. We divide anchor vectors into two types. The vector

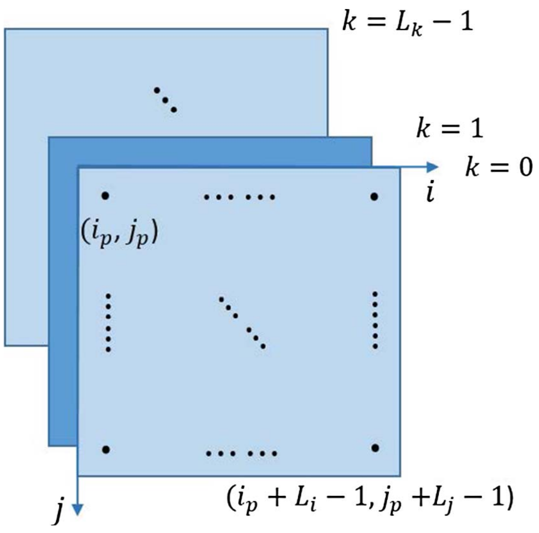

Fig. 1. Illustration of a cuboid with its pivot at $(i,j) = (i_p,j_p)$.