PCA-UNET FOR OBJECT SEGMENTATION

Cheng Long, Sayantika Nag, Adrian Barbu

Statistics Department, Florida State University

ABSTRACT

This paper introduces a PCA-based shape model for representing object shapes, and uses it as a decoder together with a CNN for instance and semantic segmentation. As opposed to standard PCA-based shape models that need point correspondences, the proposed model represents shapes as binary images of a certain size and performs PCA on the binary vectors of the training shapes. To obtain a well trained shape model, standard data augmentation techniques are applied, and an online PCA approach is introduced to be able to deal with the large number of training examples. The PCA shape model can be computed on a grid of object patches instead of the entire object. Experiments on PascalVOC reveal that this grid-based PCA has good generalization to model even shapes of novel object types. This PCA shape model is then incorporated as a decoder together with a ResNet CNN and is trained end-to-end for instance and semantic segmentation. Experiments on two popular datasets reveal that the proposed model outperforms many existing state-of-the art segmentation methods such as Masked Autoencoder, Mask-RCNN, DeepLabV3 and U-Net, while being very simple and interpretable.

***Index Terms*—** shape modeling, object segmentation, semantic segmentation

1. INTRODUCTION

Nowadays, larger and larger CNN models are trained, with billions and even trillions of parameters. While these models deliver good performance on many tasks, it is practically impossible to understand their inner working and representation. For that reason, there is a growing demand for Explainable or Interpretable AI, to find models whose inner representation is easy to understand, visualize, and even modify by the user.

One popular semantic segmentation method is the U-Net [1], which consists of an encoder that has a number of layers where the output is gradually reduced in spatial resolution but increases in the number of channels, and a decoder that gradually increases the spatial resolution back to the desired size. Skip connections between the encoder layers and the corresponding decoder layers are essential for obtaining good segmentation results. Such a model has on the order of 50-100 million trainable parameters, but it is not clear what the output features of the decoder represent, or the skip connections, making the model not very interpretable.

This paper introduces a simple UNet-like segmentation method that is more explainable by having no skip connections and replacing the decoder with a well-trained PCA model. This way, the outputs of the encoder have a simple and clear meaning: they are the PCA coefficients of the shape representation.

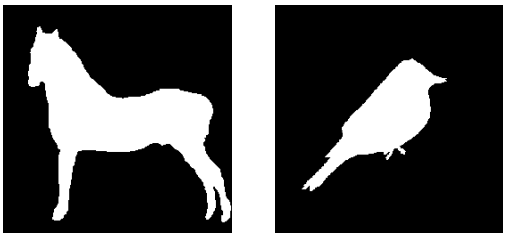


Fig. 1. Example of object shapes represented as binary images.

In summary, the paper brings the following contributions:

- It presents an approach to train a PCA model in an online manner using running averages, which enables training with an arbitrary number of data augmentation perturbations for better generalization.
- It introduces a novel PCA-based shape model that uses the binary image shape representation from [2]. This model is different from the Active Shape Model [3] because it does not need point correspondences.
- By breaking away from the common practice of representing shapes using a signed distance map [4, 5], the paper is able to introduce a local shape representation that decomposes the shape into a grid of shape patches, each modeled by the same PCA model. Experiments on PascalVOC segmentations show that the local shape model on patches of size 32×32 is very versatile, being capable to represent very well most of the PascalVOC objects without retraining.
- It introduces a method to incorporate the local PCA shape model as a decoder with a CNN such as a ResNet[6] as encoder, which is used to predict the PCA coefficients for a grid of 32×32 patches. This way the ResNet + PCA model form a UNet-like architecture that has a bottleneck (the PCA coefficients) but it has no skip connections and is more interpretable.
- Conducts experiments on two datasets for Instance/Semantic Segmentation and Shape Denoising, where it observes that the proposed segmentation method outperforms many popular methods including the UNet

[1], DeepLab-V3 [7], Mask-RCNN [8], Mask-RCNN V2 [9], and the Masked Auto-Encoder (MAE) [10]. It also conducts an ablation study where the importance of the local model is clearly demonstrated.

1.1. Related Work

Recent instance and semantic segmentation works [10, 11] use transformers as decoders for segmentation.

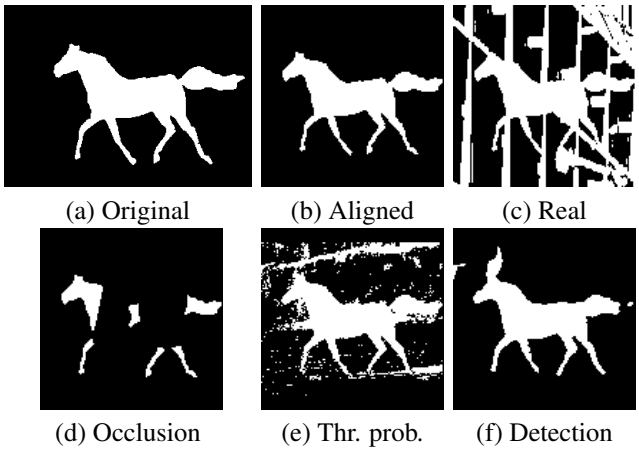
A set of filters based on PCA models was used in [12] for image classification. A somewhat similar PCA model was used in [4] as a shape model for organ segmentation, with the following differences: First, they used the signed distance transform to represent each shape while we use binary images. Second, they used a single PCA for the whole shape while we use a local PCA model for a grid of patches to improve locality and generalization. In fact, the signed distance transform cannot be used locally because the signed distance transform is a global transformation. In contrast, representing shapes using binary images allows us to divide the shape into a grid of shape patches, which can be modeled using the same or separate PCA models. Third, they did not use a CNN feature generator to predict the PCA coefficients, and instead used a variant of EM for segmentation.

Our previous work [2] introduced the representation of shapes as binary images as well as the shape denoising problem. However, the shapes were modeled indirectly using segmentation techniques such as UNet, DeepLab-V3, and MAE. In this work we model the shapes using PCA on the binary images directly, and thresholding is used to obtain a segmentation from the PCA representation. Thus the current work takes the binary shape representation further by modeling it using PCA either on the entire shape or on a grid of patches, and introduces methods for data augmentation and for dealing with the large number of examples. Finally, the current work incorporates the novel PCA shape model as a decoder for shape denoising and instance and semantic segmentation.

A local deep implicit model (LDIM) based on a number of deep implicit shapes and local Gaussian influence functions was introduced in [13]. The LDIM uses different models for the different parts, which makes it difficult to integrate into a segmentation framework and to infer the shape model parameters from images. These problems are addressed by our approach by employing the same PCA model on a grid to represent different parts of the shape locally.

Deep implicit models have also been used for medical image segmentation [5]. We tried to apply this model for vision data (horse and bird shapes) and found that the latent shape parameters, obtained by loss minimization, were unstable and hard to predict for segmentation. Consequently, the DISSM approach had major overfitting problems, and the test IOU was very low, as it will be seen in Table 4. This was the motivation for our work, since the PCA is easy to train and has stable shape parameters.

Besides these works, to the best of our knowledge, we are unaware of any other works that either represent shapes locally using PCA on binary images or use PCA as a decoder for instance or semantic segmentation.



(a) Original (b) Aligned (c) Real

(d) Occlusion (e) Thr. prob. (f) Detection

Fig. 2. Illustration of shape alignment and the four types of shape noise used in this paper. (a) Original shape, (b) Shape after alignment, (c) Real image noise, (d) Occlusion noise, (e) Thresholded probability noise, (f) Detection image noise.

2. PROPOSED METHOD

To introduce the proposed method, we first need to show how one can train a good PCA model with strong data augmentation in an online fashion.

2.1. Online PCA

PCA can be computed online from an arbitrarily large number of training examples (e.g. obtained by data augmentation), in one pass, without needing to store the data. For that, running averages (RAVE) [14, 15] can be used to obtain the exact value of the covariance matrix on all the training examples.

The running averages for a set of observations $\mathbf{x}_i \in \mathbb{R}^d, i = 1, ..., n$ are defined as $(n, \mathbf{M}_x, \mathbf{M}_{xx})$ where:

$$\mathbf{M}_x = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i, \quad \mathbf{M}_{xx} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^T. \tag{1}$$

These running averages can be updated online from a batch of b observations organized as rows in a matrix $\mathbf{B}$ as:

$$\mathbf{M}_x \leftarrow \frac{n\mathbf{M}_x}{n+b} + \frac{\mathbf{B}^T\mathbf{1}_b}{n+b},\ \mathbf{M}_{xx} \leftarrow \frac{n\mathbf{M}_{xx}}{n+b} + \frac{\mathbf{B}^T\mathbf{B}}{n+b}, \tag{2}$$

where $\mathbf{1}_b = (1, ..., 1)^T$ is the vector of ones of size b.

From the running averages, the sample covariance matrix can be computed exactly as:

$$\hat{\mathbf{\Sigma}} = \frac{n}{n-1}(\mathbf{M}_{xx} - \mathbf{M}_x\mathbf{M}_x^T) \tag{3}$$

and the sample mean is $\hat{\boldsymbol{\mu}} = \mathbf{M}_x$. From these quantities the PCA can be easily obtained by SVD.

2.2. Modeling Shapes using PCA

Similar to [2], in this paper, shapes are represented as binary images of a predefined size, e.g. 256×256 pixels, with foreground value 1 and background value 0, as shown in Figure 1. This is a different representation than the Active Shape Models [3] where the shapes are represented as a chain of contour points that have correspondence across shapes.

Shape alignment. The shapes are aligned to be roughly the same size and centered at the same location, similar to [2]. For a shape (binary image) $\mathbf{S}$, let $\mathbf{S}(x, y) \in \{0, 1\}$ be the pixel intensity value at coordinate (x, y). The shape's center is defined as its center of mass:

$$\bar{\mathbf{S}} = (\bar{\mathbf{x}}, \bar{\mathbf{y}}) = (\frac{1}{|F|} \sum_{(x,y)\in F} x, \frac{1}{|F|} \sum_{(x,y)\in F} y), \quad (4)$$

where $F = \{(x, y), \mathbf{S}(x, y) = 1\}$ are the foreground pixels.

The shape's scale is defined as the 80 percentile of the set of distances of the foreground pixels to the center:

$$D = \{\|\mathbf{x} - \bar{\mathbf{S}}\|, \mathbf{x} = (x, y) \in F\}.$$

For 256×256 images, the training shapes are aligned by resizing the image so that the scale is 80 (for the horse dataset) or 60 (for the bird dataset), then by translation so that the center is at location $(128, 128)$. An example of an aligned shape is shown in Figure 2,b).

PCA computation. The PCA shape model is obtained as described in Algorithm 1. The algorithm involves a number N^{it} of iterations of perturbing all the shapes using random rotation, translation and scaling, and updating the RAVEs.

Algorithm 1 Shape Model Training by Online PCA

Input: Aligned training shapes $\{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_k\}$
Output: Trained PCA model $\theta = (\boldsymbol{\mu}, \mathbf{P}) \in \mathbb{R}^d \times \mathbb{R}^{d\times q}$

1: Initialize RAVE as $(n, \mathbf{M}_x, \mathbf{M}_{xx}) = (0, \mathbf{0}, \mathbf{0})$
2: **for** $i = 1$ to N^{it} **do**
3: Perturb the training shapes, obtaining vectorized perturbed shapes $\mathbf{x}_1, ..., \mathbf{x}_k \in \mathbb{R}^d$.
4: Update RAVE $(n, \mathbf{M}_x, \mathbf{M}_{xx})$ using Eq. (2) with $\mathbf{B} = (\mathbf{x}_1, ..., \mathbf{x}_k)^T$.
5: **end for**
6: Compute $\hat{\boldsymbol{\Sigma}} = \frac{n}{n-1}(\mathbf{M}_{xx} - \mathbf{M}_x\mathbf{M}_x^T)$
7: Decompose $\hat{\boldsymbol{\Sigma}} = \mathbf{V}\mathbf{D}\mathbf{V}^T$ by SVD
8: Obtain $\boldsymbol{\mu} = \mathbf{M}_x$, and $\mathbf{P}$ as the first q columns of $\mathbf{V}$.

The RAVE for a $128 \times 128 = 2^{16}$ pixel image contains the matrix $\mathbf{M}_{xx}$ of size $d \times d$ with $d = 2^{16} = 16,384$, which is quite large and non-sparse.

Grid PCA. For more modeling flexibility and accuracy and to be able to handle large images, the shape space (of binary images) can be divided into a grid of non-overlapping rectangles (patches) and each rectangle can be modeled by PCA.

In our application, the 256×256 shape images are divided in $8 \times 8 = 64$ squares of size 32×32 each. This way one RAVE (or a grid of 64 RAVEs) with $d = 1024$ is used in Algorithm 1 instead of one RAVE with $d = 65535$.

PCA Shape Reconstruction. A reconstructed binary PCA shape is obtained as $\hat{\mathbf{y}} = I(\hat{\mathbf{S}} > 0.5)$, where $I(\cdot)$ is the indicator function ($I(v) = 1$ if v is true, otherwise 0) applied elementwise, and

$$\hat{\mathbf{S}} = \boldsymbol{\mu} + \mathbf{P}\mathbf{c}, \quad (5)$$

where $\boldsymbol{\mu}$ is the vectorized mean, $\mathbf{P}$ are the vectorized PCs, and $\mathbf{c} \in \mathbb{R}^q$ is a parameter vector.

Similarly, given a grid of parameter vectors $\mathbf{c}_{ij} \in \mathbb{R}^q$, $(i, j) \in \{1, ..., K\} \times \{1, ...L\}$, Eq. (5) can be used to reconstruct corresponding patches

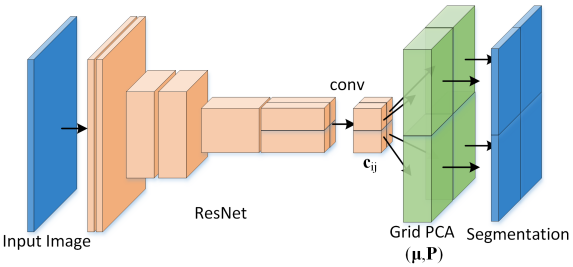


Fig. 3. Diagram of the PCA-UNet.

$$\hat{\mathbf{S}}_{ij} = \boldsymbol{\mu}_{ij} + \mathbf{P}_{ij}\mathbf{c}_{ij}, (i, j) \in \{1, ..., K\} \times \{1, ...L\}, \quad (6)$$

which are placed at the appropriate grid locations (i, j) to obtain the whole reconstructed shape $\hat{\mathbf{y}}$ after thresholding.

Here a grid of PCAs $(\boldsymbol{\mu}_{ij}, \mathbf{P}_{ij})$ can be used for aligned images of the same size (see Figure 3), or a single PCA $(\boldsymbol{\mu}, \mathbf{P})$ can be used on the grid of patches

$$\hat{\mathbf{S}}_{ij} = \boldsymbol{\mu} + \mathbf{P}\mathbf{c}_{ij}, (i, j) \in \{1, ..., K\} \times \{1, ...L\} \quad (7)$$

to obtain the whole $\hat{\mathbf{S}}$ and the reconstructed shape $\hat{\mathbf{y}} = I(\hat{\mathbf{S}} > 0.5)$. Experiments from Section 3.2 will reveal that a single PCA $(\boldsymbol{\mu}, \mathbf{P})$ is as effective as a grid of PCAs for modeling object shapes using a grid of 32×32 pixel patches.

2.3. PCA-UNet for Semantic Segmentation

For semantic segmentation, a fully convolutional neural network (FCNN) such as a ResNet[6] is used as a feature generator for predicting the PCA coefficients for each patch, as illustrated in Figure 3.

The size of the patch needs to align with the output of the FCNN. For example, a ResNet-50 obtains from an input image of size 256×384 a feature vector of size $8 \times 12 \times 2048$ before the average pooling layer. In this case, the shape is decomposed into a grid of 8×12 patches of size 32×32 each. Then the PCA coefficients are predicted from the CNN outputs using a 1×1 convolutional layer. For example, if the 32×32 shape patches are represented with a PCA $(\boldsymbol{\mu}, \mathbf{P})$ with $q = 64$ PCs, then the convolutional layer has size $1 \times 1 \times 64$, obtaining a $8 \times 12 \times 64$ output of predicted PCs $\mathbf{c}_{i,j} \in \mathbb{R}^{64}, (i, j) \in \{1, .., 8\} \times \{1, .., 12\}$ from the $8 \times 12 \times 2048$ output of the ResNet-50. Optionally, an attention layer [16] could be introduced before the convolution layer.

Then the PCA model $(\boldsymbol{\mu}, \mathbf{P})$ is used to reconstruct the corresponding patches of the output segmentation from the predicted PCs using Eq. (7).

Training the PCA-UNet. The PCA-UNet's trainable parameters are $(\mathbf{R}, \mathbf{C})$, consisting of the CNN Layers $\mathbf{R}$ and the added convolutional layer $\mathbf{C}$, which together form a FCNN $f(\mathbf{x}; \mathbf{R}, \mathbf{C}) = (\mathbf{c}_{11}(\mathbf{x}; \mathbf{R}, \mathbf{C}), ..., \mathbf{c}_{KL}(\mathbf{x}; \mathbf{R}, \mathbf{C}))$. Observe that the PCA model $(\boldsymbol{\mu}, \mathbf{P})$ is frozen. Training is done in an end-to-end fashion by standard backpropagation using the IOU Loss, which for a training example $(\mathbf{x}, \mathbf{y})$ is:

$$L(\hat{\mathbf{y}}, \mathbf{y}) = \frac{\sum_j \hat{\mathbf{y}}_j \mathbf{y}_j}{\sum_j \max(\hat{\mathbf{y}}_j, \mathbf{y}_j)}, \quad (8)$$

where the sums are taken over the pixels, $\hat{\mathbf{y}} = \sigma(\alpha(\hat{\mathbf{S}}))$, with $\sigma(\cdot)$ as the sigmoid, α is a tuning parameter ($\alpha = 50$ in our experiments), and $\hat{\mathbf{S}}$ is obtained as in Eq (5), or reconstructed from patches as in Eqns. (6) and (7).

2.4. A Two Step Approach for Instance Segmentation

For instance segmentation, one can use a standard object detector such as Faster-RCNN [17] to find the objects of interest. Then for each detected box a square image is cropped with the same center and with the side as the largest of the width and height of the detected box. Then the cropped square boxes are resized to 256×256 and input into the PCA-UNet, whose segmentation output is then resized back to the original cropped box size. The segmentation result is obtained by initializing an all-zero matrix of the same size as the input image and placing the obtained segmentations at the locations where their corresponding input was cropped from.

3. EXPERIMENTS

Datasets. Experiments will be mainly conducted on two datasets: The Weizmann Horse dataset [18] and the CUB-200 Birds dataset [19]. Both datasets contain images with a single object per image, making semantic segmentation and instance segmentation equivalent in this case.

The Weizmann Horse dataset [18] contains 327 horse color images and their corresponding manual segmentations as binary mask images. From the 327 images, 159 images were randomly selected as the training set and the other 168 images as the test set.

The Caltech-UCSD Birds (CUB) 200 dataset [19] contains photos of 200 bird species with ground truth segmentations. We used 417 images of seven Flycatcher species for our experiments, of which 207 images were randomly selected as the training set and the other 210 as the test set.

Tasks. Experiments will be conducted for three tasks: shape reconstruction, shape denoising [2] and color image instance segmentation.

Shape Reconstruction. This task only evaluates how well can the proposed PCA model reconstruct segmentations.

Object Segmentation. The object segmentation task is the task of segmenting each instance of the object of interest (horse or bird) from the color image. Since there is only one object in each image for these datasets, this task is both instance segmentation and semantic segmentation.

Shape Denoising. The shape denoising task is about recovering the object shape from a binary image containing the shape perturbed by some kind of noise. We apply four types of noise introduced in [2] on the ground truth segmentation (mask) images. These noises are the real image, occlusion, thresholded probability and detection noise, illustrated in Figure 2.

Evaluation Measure. For all experiments, we measure the average IOU (Intersection Over Union) over the test set. The IOU for two binary images M, Y is defined as:

$$IoU(M,Y) = \frac{|C_M \cap C_Y|}{|C_M \cup C_Y|}, \tag{9}$$

where $C_J = \{(x,y)|J(x,y) = 1\}$ are the pixels with value 1.

Table 1. Shape reconstruction capability (IoU) on S_{test}^{m-all} of PCA models trained with different number of perturbations per image. Average IOU (std) obtained over 5 runs.

Patch size	# PCA	Img. size	Number of perturbations per training shape 1	10	100	1000
Weizmann Horse dataset						
128	1	128	0.746(.029)	0.860(.002)	0.872(.001)	0.874(.000)
	2×2	256	0.864(.013)	0.930(.001)	0.938(.000)	0.939(.000)
64	4×4	256	0.902(.008)	0.964(.000)	0.970(.000)	0.971(.000)
32	8×8	256	0.935(.019)	0.982(.000)	0.987(.000)	0.988(.000)
	1	256	0.984(.000)	0.987(.000)	0.987(.000)	0.987(.000)
CUB 200 Birds dataset						
128	1	128	0.813(.023)	0.886(.001)	0.896(.000)	0.897(.000)
	2×2	256	0.882(.014)	0.942(.001)	0.948(.000)	0.949(.000)
64	4×4	256	0.924(.004)	0.962(.000)	0.970(.000)	0.971(.000)
32	8×8	256	0.939(.011)	0.982(.000)	0.988(.000)	0.989(.000)
	1	256	0.985(.000)	0.987(.000)	0.987(.000)	0.987(.000)
Horses & Birds combined dataset						
32	1	256	0.986(.000)	0.987(.000)	0.987(.000)	0.987(.000)

3.1. Implementation Details

All methods were implemented in PyTorch. The CNNs were trained with the IOU Loss (8) and the Adam optimizer [20]. The shape denoising PCA-UNets were trained with 120 epochs with batch size 32 and learning rate 0.001. The RGB segmentation PCA-UNets for were trained with 2000 epochs with learning rate 0.00003 and batch size 32, initialized with a pretrained model. Training the PCA-UNet for RGB segmentation takes about 1h on a GeForce 3090 GPU, which is 20 times faster than U-Net and MAE.

3.2. Shape Reconstruction Experiments

In this section we will evaluate the capability of the proposed PCA model to represent the object shapes. We will evaluate a one PCA model on 128×128 pixel aligned shapes and PCA models on grids of patches on 256×256 pixel aligned shapes. All PCA models have 64 principal coefficients (PCs). We did not train a single PCA model on 256×256 because it involves working with a 65536×65536 matrix, which takes 18GB of memory and is computationally prohibitive to run SVD on it.

The shape reconstruction capability is measured by the IOU of reconstructing the input shape from the PCA coefficients. The PCAs were trained using online PCA and different numbers of data augmentation perturbations per image. The data augmentation perturbations were: random resizing the 256×256 input in the interval $[241, 271]$, random rotation up to ± 15 degrees, and random cropping a 256×256 image with padding up to 15 pixels.

The reconstruction results on the test set (which was not used for training the PCAs) are shown in Table 1. From Table 1 one can see that a grid of 8×8 PCAs with patch size 32×32 and 64 PCs can do a very good job reconstructing the shapes.

Moreover, a single PCA trained on the 32×32 pixel patches can do as good a job as a grid of PCAs.

Out of curiosity, we also trained a single PCA on the combined Horse and Birds train sets and show the reconstruction

Table 2. Shape reconstruction IoUs on the PascalVOC validation set of PCA models trained on either the Horses & Birds dataset or on the PascalVOC train set. Average IOU and std obtained over 5 runs.

	Trained on			Trained on	
Object	Horses&Birds	VOC	Obj	Horses&Birds	VOC
aeroplane	0.954(.000)	0.955(.000)	bus	0.995(.000)	0.995(.000)
bicycle	0.813(.000)	0.811(.000)	car	0.977(.000)	0.976(.000)
bottle	0.982(.000)	0.981(.000)	chair	0.938(.000)	0.939(.000)
diningtable	0.993(.000)	0.993(.000)	dog	0.990(.000)	0.990(.000)
horse	0.980(.000)	0.980(.000)	train	0.993(.000)	0.993(.000)
motorbike	0.977(.000)	0.977(.000)	cow	0.986(.000)	0.985(.000)
person	0.974(.000)	0.974(.000)	sofa	0.996(.000)	0.995(.000)
pottedplant	0.876(.000)	0.875(.000)	cat	0.994(.000)	0.994(.000)
sheep	0.978(.000)	0.978(.000)	boat	0.959(.000)	0.961(.000)
tvmonitor	0.994(.000)	0.995(.000)	bird	0.968(.000)	0.968(.000)

results on the combined test sets in Table 1. One can see that the reconstruction (with at least 10 perturbations per image) is as good as the one trained on a single dataset (e.g. Horse). This might mean that the trained PCA on 32×32 patches is capable of reconstructing arbitrary shapes.

To test this hypothesis, we used the shape model trained on the combined Horses & Birds train sets to reconstruct the shapes (segmentations) of the the PascalVOC validation set. The results are shown in Table 2. Also shown are results of equivalent PCA models trained on the segmentations of the PascalVOC train set. The results are very similar, sometimes differing by 0.001. From Table 2 one could see that most of the shapes can be accurately reconstructed by the PCA model, except for the bicycles and potted plants.

3.3. Real Data Experiments

The real images with different sizes were resized for batch processing to have the largest size 384 and were padded with zeros to make all of them of size 384×384. The corresponding masks were resized the same way. Evaluation was done on each image after removing the zero padding.

All PCA-UNets used the same PCA model trained on 32×32 pixel patches, which was evaluated in Table 1 as the "Horses & Birds combined dataset".

We compare the proposed PCA-UNet with UNet [1], and MAE [10]. The UNet has 4 blocks in the encoder and decoder respectively. Each block consisted of two 3×3 convolutions, each convolution followed by Batch Normalization (no batch normalization in the first block) and ReLU. In the encoder, 2×2 max pooling with stride 2 is used for downsampling. The initial number of channels is 128, which is doubled at each downsampling step. Bilinear interpolation is used for upsampling in the decoder, and the number of channels is halved at each upsampling step. The MAE uses the ViT-base (12 self-attention layers) as encoder, which is initialized by a model pretrained on the ImageNet-1K dataset. The decoder consisted of 4 self-attention layers and 2 fully connected layers.

Table 3. Segmentation on original color images. Average train and test IoU (std) obtained over 4 runs.

Method	Backbone	# pars. $\times 10^6$	Weizmann Horse train	Weizmann Horse test	CUB-200 Birds train	CUB-200 Birds test
DeepLabV3 [7]	RN50	42	0.869	0.861	0.841	0.838
Mask-RCNN [8]	RN50	44	0.736	0.797	0.838	0.836
Mask-RCNN V2. [9]	RN50	46	0.877	0.866	0.872	0.879
U-Net [1]	UNet	126	0.972(.001)	0.889(.003)	0.960(.001)	0.823(.003)
MAE [10]	VIT base	114	0.953(.001)	**0.917**(.001)	0.951(.000)	0.860(.001)
PCA-UNet	RN50	24	0.960(.000)	**0.917**(.000)	0.953(.001)	**0.885**(.000)
PCA-UNet-ATT	RN50	40	0.948(.000)	0.911(.000)	0.945(.000)	0.881(.001)
FRCNN+PCA-UNet	RN50	44+24	0.956(.000)	0.916(.000)	0.958(.000)	**0.885**(.000)
FRCNN+PCAUNet-ATT	RN50	44+40	0.946(.001)	0.898(.000)	0.950(.000)	0.877(.000)

3.3.1. Color Segmentation Experiments

The UNet, MAE, and PCA-UNet with or without attention were trained on the same dataset of images resized to 384×384. Data augmentation involved random translation by up to 50 pixels, and random rotation by up to ± 25 degrees.

The PCA-UNet for the Faster RCNN was trained with 256×256 images cropped and resized as described in Section 2.4 based on the Faster RCNN output. The images and the corresponding masks were perturbed by random translation up to 12 pixels and random rotation up to ± 12 degrees.

Also shown are results from DeepLabV3 [7], Mask-RCNN [8], and Mask-RCNN V2 [9], all using the pretrained ResNet50 models available in PyTorch. The segmentation time for PCA-Unet is hundreds of times faster than the other methods, due to its architecture and batch processing.

The color segmentation results are shown in Table 3, averaged over 4 random initializations and with standard deviations shown in parentheses. From Table 3 one could see that the PCA-UNet and its two-step version with Faster-RCNN have the best test IOU performance on both datasets. The MAE comes second, with similar results on the Horse data but is outperformed on the Birds data. The PCA-UNet-ATT has difficulties obtaining a small training loss, which is reflected in a subpar training and test IOUs.

3.3.2. Shape denoising experiments

Dataset. Shape denoising involves recovering the object shape from a perturbed binary image. The training set consists of original training shapes perturbed by real image, occlusion and thresholded probability noises (see Figure 2, c, d and e), obtaining a training set containing 31387 images for Horses and 28065 images for Birds. The test sets contains the original test shapes perturbed by the four noise types: real, occlusion, detection image and thresholded probability, for a total of 19778 Horse images and 21951 Bird images.

The results are shown in Table 4. We also evaluated DISSM [5], with an implementation based on the DISSM paper authors' implementation from GitHub. From Table 4 one could see that again the PCA-UNet outperforms the other methods and that attention helps a little bit.

3.4. Ablation Study and Parameter Analysis

This section evaluates the importance of different hyperparameters and architecture choices such as the size of the

Table 4. Shape denoising performance (IoU) on original test masks corrupted by four types of noise $S^{m-all}_{test-original}$.

Noise Type	DISSM	U-Net	MAE	PCAUNet (one step)	PCAUNetATT (one step)	FasterRCNN +PCAUNet	FasterRCNN+ PCAUNetATT
Weizmann Horse dataset							
Real Image	0.461	0.900	0.883	**0.945**	**0.945**	0.944	0.942
Detection	0.644	0.863	0.861	0.868	0.870	0.871	**0.877**
Thr. Prob	0.590	0.865	0.865	0.900	**0.904**	0.899	0.901
Occlusion	0.531	0.871	0.861	0.910	**0.914**	0.906	0.905
Overall	0.557	0.875	0.867	0.906	**0.908**	0.905	0.906
CUB-200 Birds dataset							
Real Image	0.246	0.866	0.847	**0.926**	0.903	0.917	0.920
Detection	0.451	0.712	0.716	0.691	0.701	**0.719**	0.717
Thr. Prob	0.316	0.676	0.676	**0.774**	0.725	0.766	0.772
Occlusion	0.439	0.843	0.833	**0.896**	0.880	0.883	0.884
Overall	0.363	0.774	0.768	0.822	0.802	0.821	**0.823**

PCA patches to cover the reconstructed shape, the number of principal components (PCs) and the number of data augmentation perturbations to train the PCA models. This evaluation will be conducted for the color segmentation task.

All experiments use ResNet50 as encoder, followed by the 1×1 convolutional layer to obtain the desired number of predicted principal coefficients (PCs).

The experiments on patches of size 32×32 and 64×64 are conducted on images resized to 384×384 as described in Section 3.3.1. In this case, the output of the ResNet before the average pooling layer is 12×12, which for patches of size 32×32 results in a reconstructed segmentation of the same size as the input. For the patches of size 64×64 we used a 2×2 average pooling after the 1×1 convolutional layer to obtain a segmentation of the correct size.

The experiments on patches of size 128×128 are using Faster-RCNN as described in Section 2.4, and trained as described in Section 3.3.1. The ResNet50 is followed by 8×8 average pooling, which results in single set of PCs and an output of size 128×128, which is then resized and placed at the location detected by Faster-RCNN.

The PCA used as a decoder for the 32×32 pixel patches is trained on the "Horses & Birds" combined training set. The PCAs for the 64×64 and 128×128 pixel patches are trained on the respective training sets (horses or birds).

In Table 5 are shown the average IOU results for the three patch sizes considered, different numbers of PCs and for PCA models trained with different numbers of perturbations per training shape.

Size of the PCA patches. By changing the size of the grid PCA patches to reconstruct the segmentation, we are making a trade-off between a local and a global shape model.

From Table 5 one could see that a finer grid results in better IOU results. A finer grid is also easier to train, since a PCA on a $d \times d$ patch involves computing the SVD of a $d^2 \times d^2$ matrix, which could be prohibitive if d is large. Fortunately, using a grid of patches alleviates the problem by greatly reducing the patch size and hence the size of the matrix for SVD.

Number of PCs. Since larger patch sizes give inferior results, we will focus on 32×32 pixel patches. From Table 5 one could see that in this case good results can be obtained with as few as 16 PCs and that with 64 PCs, results are optimal regardless of the number of data augmentation perturbations.

Table 5. Test set color segmentation IoU of PCA-UNet using PCA models of different sizes trained with different number of perturbations per training shape.

Patch size	#PC	Weizmann Horses				CUB-200 Birds			
		1	10	100	1000	1	10	100	1000
128×128	64	0.684	0.717	0.724	0.717	0.729	0.738	0.730	0.733
	256	0.713	0.722	0.732	0.734	0.735	0.733	0.742	0.733
	1024	0.714	0.729	0.720	0.718	0.711	0.740	0.735	0.733
64×64	32	0.891	0.896	0.896	0.898	0.867	0.872	0.872	0.870
	64	0.898	0.898	0.899	0.899	0.869	0.874	0.875	0.872
	128	0.897	0.898	0.897	0.897	0.871	0.872	0.871	0.870
32×32	8	0.911	0.911	0.911	0.910	0.881	0.881	0.883	0.881
	16	0.917	0.917	0.916	0.916	0.885	0.885	0.886	0.886
	32	0.916	0.917	0.917	0.918	0.885	0.885	0.885	0.887
	64	0.917	0.918	0.917	0.917	0.887	0.886	0.885	0.885
	128	0.916	0.916	0.917	0.916	0.883	0.884	0.884	0.885

Size of data augmentation for PCA training. Data augmentation is important for training some of these PCA models. For example, for a single PCA on 128×128 size images, we only have about 200 data points in a 16384 dimensional space without data augmentation. These are very few points in a very high dimensional space.

In Table 5 are shown the average test segmentation IOU results of PCA-UNet using a PCA trained with different numbers of perturbations per image. From Table 5 one could see that data augmentation is important for the 128×128 model, and the number of perturbations that give the best result depends on the number of PCA coefficients. For the 32×32 and 64×64 models, data augmentation seems not to be important. This is probably because there are many training observations examples obtained from each image (at least 16).

4. CONCLUSION AND FUTURE WORK

This paper introduced a PCA model trained on aligned binary images or image patches. Experiments on PascalVOC showed that the patch-based PCA model is versatile, being able to approximate shapes of objects that it was not trained on, except for bicycles and potted plants.

The paper also introduced a Fully Convolutional segmentation architecture that uses the PCA model as a decoder and obtains state of the art semantic segmentation results on two datasets, outperforming modern segmentation methods such as the Masked Auto-Encoder and the Masked-RCNN V2.

While the method performs well for object segmentation, its main limitation is that the output is binary, so it is capable of segmenting a single type of object (unless it is combined with a Faster-RCNN like in Section 2.4). However, the approach can be directly generalized to multi-object segmentation, which we plan to do in the future.

In the future we also plan to generalize the proposed method to 3D segmentation, for example for 3D organ or tumor segmentation in medical imaging or for 3D reconstruction of objects such as humans from a single image.

5. REFERENCES

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*. Springer, 2015, pp. 234–241.

[2] Cheng Long and Adrian Barbu, "A study of shape modeling against noise," in *ICIP*. IEEE, 2022, pp. 611–615.

[3] Timothy F Cootes, Christopher J Taylor, David H Cooper, and Jim Graham, "Active shape models-their training and application," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38–59, 1995.

[4] Kilian M Pohl, Simon K Warfield, Ron Kikinis, W Eric L Grimson, and William M Wells, "Coupling statistical segmentation and pca shape modeling," in *MICCAI*. Springer, 2004, pp. 151–159.

[5] Ashwin Raju, Shun Miao, Dakai Jin, Le Lu, Junzhou Huang, and Adam P Harrison, "Deep implicit statistical shape models for 3d medical image delineation," in *AAAI*, 2022, vol. 36, pp. 2135–2143.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[7] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask r-cnn," in *ICCV*, 2017, pp. 2961–2969.

[9] Yanghao Li, Saining Xie, Xinlei Chen, Piotr Dollar, Kaiming He, and Ross Girshick, "Benchmarking detection transfer learning with vision transformers," *arXiv preprint arXiv:2111.11429*, 2021.

[10] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick, "Masked autoencoders are scalable vision learners," in *CVPR*, 2022, pp. 16000–16009.

[11] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al., "Segment anything," in *CVPR*, 2023, pp. 4015–4026.

[12] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma, "Pcanet: A simple deep learning baseline for image classification?," *IEEE Trans. on Image Processing*, vol. 24, no. 12, pp. 5017–5032, 2015.

[13] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser, "Local deep implicit functions for 3d shape," in *CVPR*, 2020, pp. 4857–4866.

[14] Harold Kushner and G George Yin, *Stochastic approximation and recursive algorithms and applications*, vol. 35, Springer Science & Business Media, 2003.

[15] Lizhe Sun, Mingyuan Wang, Siquan Zhu, and Adrian Barbu, "A novel framework for online supervised learning with feature selection," *Journal of Nonparametric Statistics*, pp. 1–27, 2024.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *NeurIPS*, vol. 30, 2017.

[17] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *NeurIPS*, vol. 28, 2015.

[18] Eran Borenstein, Eitan Sharon, and Shimon Ullman, "Combining top-down and bottom-up segmentation," in *CVPR Workshop*, 2004, pp. 46–46.

[19] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD Birds 200," Tech. Rep. CNS-TR-2010-001, California Institute of Technology, 2010.

[20] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.