One Shot Learning via Compositions of Meaningful Patches

Alex Wong
University of California, Los Angeles
alexw@cs.ucla.edu

Alan Yuille
University of California, Los Angeles
yuille@stat.ucla.edu

Abstract

The task of discriminating one object from another is almost trivial for a human being. However, this task is computationally taxing for most modern machine learning methods; whereas, we perform this task at ease given very few examples for learning. It has been proposed that the quick grasp of concept may come from the shared knowledge between the new example and examples previously learned. We believe that the key to one-shot learning is the sharing of common parts as each part holds immense amounts of information on how a visual concept is constructed. We propose an unsupervised method for learning a compact dictionary of image patches representing meaningful components of an objects. Using those patches as features, we build a compositional model that outperforms a number of popular algorithms on a one-shot learning task. We demonstrate the effectiveness of this approach on hand-written digits and show that this model generalizes to multiple datasets.

1. Introduction

Perhaps one of the more impressive feats of human intelligence is the ability to learn a concept from few examples — or even just one. At a young age, children easily learn their first language without complete exposure to the entire language and can even make inferences on novel concepts from their limited knowledge [8]. In fact, they can acquire a new word based on a single encounter [4]. However, if we survey state of the art learning methods, the results presented are the product of training from thousands of examples, where even a simple method such as logistic regression can perform very well [17]. Such performance becomes difficult to attain with only a single example.

We believe that the basis for one-shot learning stems from the sharing of similar structures amongst objects of the same class. A bicycle can be parsed into a set of handles connected to the frame with two wheels and a seat. We can easily recognize a similar visual concept (eg. tricycle or motor bike) when it can be decomposed to a similar set of parts that embodies the structure of the objects. These parts
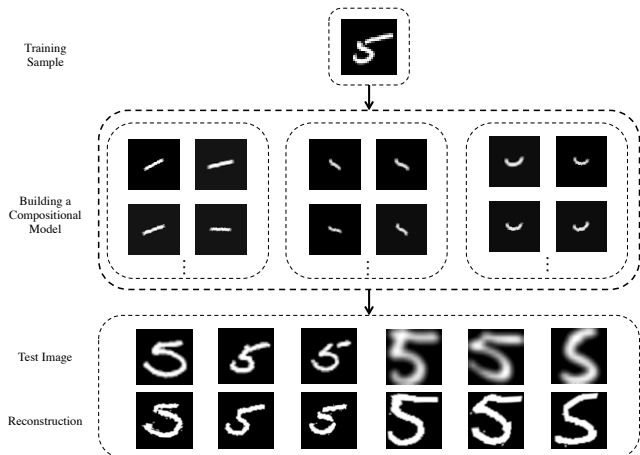

Figure 1. Examples of reconstructions produced by our method. The model was trained on MNIST and can generalize to the USPS hand-written digit dataset.

and their relations give us the basis for representing a number of other similar vehicles. We seek to exploit the innate structures within visual concepts by learning a set of parts for a compositional model that can tackle one-shot learning.

Our work is motivated by [19] who showed that a part-based model is an effective means of achieving one-shot learning. Their work highlights a compositional model that showed promising results on a one-shot character recognition task. After building an in-house dataset recording both characters and the individual strokes human participants used to draw them, they trained their model on a single image from each class leveraging this set of strokes. Although the authors showed that one-shot learning can indeed be done, their method requires extensive human aid in generating a set of labeled strokes that compose each character. The need for these hand-crafted features in turn limited their work to a non-standard dataset (not yet released to the research community). Motivated by these limitations, our goal is to extend the use of part-based models to one-shot learning without the need for human supervision so that it can be applied to common datasets. Our method uses symmetry axis [24] as an object descriptor (Fig. 2, 3) and

learns a set of meaningful components by parsing the skeleton. We then build an AND-OR graph that describes each class of objects and perform recognition on a new image by selecting the grammar that best reconstructs the image.

We specifically apply our work to hand-written digits. Although hand-written digits appear to be very simple objects, there exists a surprisingly large amount of variation for writing a single digit. Yet, there still exists common components amongst digits of the same class that we can leverage. Each digit contains rich internal structures that describe the formation of the general class as a whole. Our goal is to learn these components (strokes) using just the digits given to us (without the aid of a global stroke set) and perform digit recognition as a proof of concept. In the future, we plan to apply this technique to more general shapes and objects.

Our contributions are two-fold. We first present a robust method for extracting meaningful patches from visual concepts. We do so by finding the symmetry axis in each object and partitioning it into components (describing the structure of a local region within the object), which we convert into image patches to be used as features. Secondly, we use these patches to construct an AND-OR graph that represents the object as a composition of the extracted patches. We apply a set of deformations to the patches to generate a dictionary accounting for intra-class variation. Recognition is accomplished by reconstructing new images using our compositional model — we choose the class of the best reconstruction as our label. We show that our generative model not only outperforms a number of popular learning techniques on a one-shot learning task, but is also transferable between datasets — achieving similar accuracies when tested on different datasets.

2. Related Work

Current state-of-the-art learning algorithms are able to learn complex visual concepts and achieve high recognition accuracies. For example, [6] has surveyed many techniques discussing the performance of state-of-the-art algorithms on hand written digits datasets, with each classifier reporting extremely low error rates. The MNIST dataset, proposed by [22], has become a baseline for many classifiers, most of which can obtain near-perfect accuracy ($\approx 99\%$) [21]. Popular methods such as k-Nearest Neighbors [10], Support Vector Machine [9], and more recently Deep Boltzmann Machines [27], and Convolution Neural Networks [7] have shown that the dataset poses no challenge when provided with a sufficient number of training examples. Common datasets, like MNIST, provide thousands of training examples for each class and the aforementioned models requires large amounts of training examples to achieve such impressive results. In contrast, a human only needs a few examples to learn to distinguish one object from another with ease. It
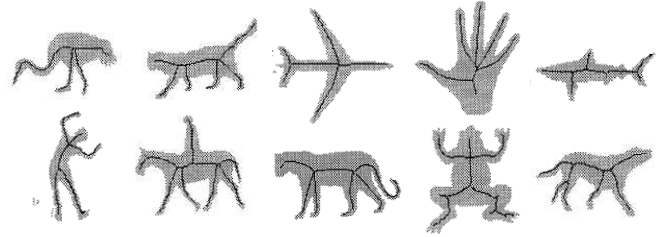

Figure 2. Symmetry axis used as a robust object descriptor to automatically extract skeletons of objects [31]. The components of the symmetry axis are connected by complex junctions joining 3 or more pixels.

Figure 3. Symmetry axis being applied to hand-written digits 0-9.

is safe to say these state-of-the-art approaches are still far from reaching the proficiency of a human being.

2.1. One-Shot Learning

One shot learning is an object categorization task where very few examples (1–5) are given for training. In recent years, one-shot learning has made significant strides forward [26, 14, 19, 20]. Earlier work on one-shot digit learning focused on the concept of transferable knowledge through image deformations. The authors of [26] discussed the use of scale and rotation to represent the notion of knowledge transfer. They reported low errors rates in their experiments; however, their method may not converge and also creates additional large artificial datasets based from their one shot samples for training. [14] explored one-shot learning in the realm of object categorization by taking advantage of features learned from previous categories and representing them as probabilistic models. Specifically, they created a constellation model to generate a set of hypothesis for selecting the best fit class. The graph connections of the model were created based on the location and appearance of the features. However, the model suffered from complexity issues and is only able to use very few features for each hypothesis.

A more recent study of one-shot learning in hand-written characters proposed that similar visual concepts are composed by a set of common components. [19] suggested that the sequence of strokes used to produce a character contains large amounts of information about the internal structure of the character. They collected a new dataset of 1600 characters by having participants draw characters online —
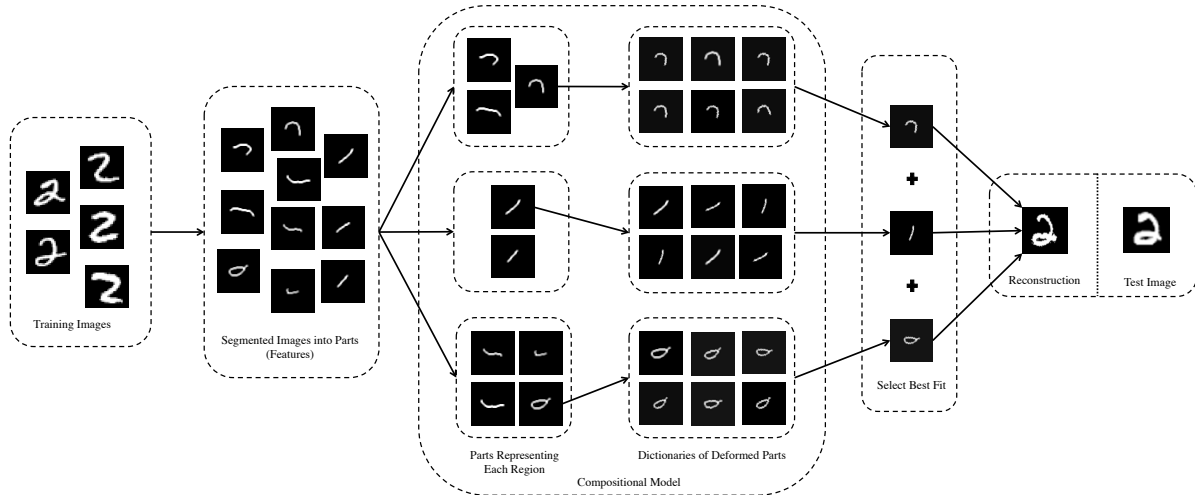
Figure 4. Overview of our approach applied to hand-written digits. Objects are decomposed into parts by segmenting their symmetry axes. We represent the objects using an AND-OR graph composed of image patches that describes regions of the objects. Deformations are applied to the patches to create dictionaries. We select the best patches from the dictionaries to reconstruct a test image.

collecting the strokes as well as how the strokes construct each character. Their probabilistic model learns from a global set of strokes for the character set and infers a set of latent strokes from an example to create a part based-representation of the characters. The approach of Lake et al. [19] boasts a higher accuracy than the Deep Boltzmann Machine, beating the deep learning approach by a 15% margin, when both are trained on a single image per class. Lake et al. presented a second method [20] similar to his earlier work that uses a Hierarchical Bayesian model based on compositionality and causality. They boasted a human-like performance when presenting human participants with a set of images generated by their method in a "visual Turing Test". Their performance suggests promising avenues for this field.

2.2. Patch-based Model

Recent literature has involved a number of algorithms with successful patch-based models [12, 23, 25, 30]. Learning dictionaries of generative image features showcases a number of desirable qualities as they provide an intuitive and economical mid-level representation for visual processing systems. Each image patch contains large amounts of information, acting as a great mid-level feature that allows for versatility in reconstruction as well as transferability in learning. Our method also tries to exploit these properties and we model our approach after the work by [25] and [30].

[25] described an approach that was able to produce state-of-the-art results on textures. They provide a dictionary of active patches that undergo spatial transformations to adjust themselves to best fit an image. The method is able to perform on datasets ranging from homogenous to inhomogenous appearance of general object categories. This

is mainly due to the nature of the active patches model and the flexibility it provides for matching textures. The active patches model can be applied to a wide range of tasks to achieve desirable results.

In the domain of hand-written digits, [30] has proven successful using a dictionary of deformable patches. They propose a simple method for learning a dictionary of deformable patches for simultaneous shape recognition and reconstruction. Similar to [25], the authors of [30] introduced a pre-defined set of transformations on image patches. They designed a GPU framework for matching a large number of deformable templates to a large set of images efficiently. Their dictionary of deformable patches has reported state-of-the-art recognition performance on both MNIST [22] and USPS [15]. In addition, they also showed that the dictionary learning method can perform well when transferring the learned dictionary between different datasets.

This paper is organized as follows: we present our approach in Sec. 3. Specifically, we detail our process for extracting meaningful patches as features in Sec. 3.1 and how we build our compositional model using these patches in Sec. 3.2. Next, we then apply our model to novel images in Sec. 3.3. Implementation details are presented in Sec. 4, including the parameters we used to achieve our results. We present experimental results on hand-written digit recognition in Sec. 5 and conclude with potential drawbacks and future directions in Sec. 6.

3. Our Approach

Our goal is to learn a set of patches that captures the underlying structures shared by each set of objects using only a small number of examples. We do so by applying symme-
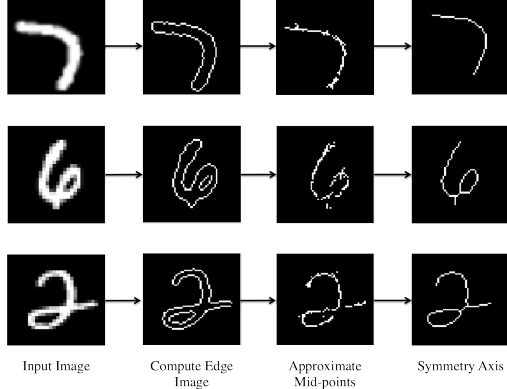
Figure 5. Hand-written digits skeletonized via symmetry axis. Given an input image, we compute the edge image and compute $a_i \in \mathcal{A}$ from a pair of points, p_i^l and p_i^r. Missing pixels along the axis are filled in and dangling branches are pruned.

Figure 6. Preliminary stroke models, $\mathcal{S}^t \in \mathcal{S}$, composed of $\mathbf{R}_k^t \in \mathcal{R}^t$. Each region $\mathbf{R}_k^t$ is generated by hashing the set of segment patches, $\mathbf{s}_i \in \mathcal{U}^t$, centered at c_i to the nearest anchor. We chose 3 anchors on a 56×56 grid to represent the top, middle and bottom regions of the stroke model.

try axis to each object and segmenting the skeleton into a set of components; these components are in turn converted to image patches. We then learn a compositional patch model by creating an AND-OR graph composed of dictionaries of meaningful patches to represent each object. This generative model is used to recognize new images by matching candidate patches from our dictionaries to the images and selecting the best fit grammar to reconstruct the novel object. We ensure the quality of the set of reconstructions proposed by minimizing a cost function, which incorporates penalties for misfits and lack of coverage. The transformations between the proposals and the test image are computed and the test image is reconstructed by warping the proposals. The class of the best fit reconstruction is selected as our label. Fig. 4 represents an overview of our approach.

3.1. Learning a set of Meaningful Patches

We present an unsupervised method for generating a dictionary of meaningful patches from an image by finding its symmetry axis to produce a skeleton of the object. We then parse the skeleton into components by identifying the endpoints and branch-points. We join these components into meaningful parts by defining a set of points on the image containing our object and hashing the components to the closest point to create a set of image patches. Each patch represents a mid-level feature that describes the structure of the object at a given region. Unlike traditional dictionary learning, only a small the number of patches are produced during the feature extraction. We demonstrate the effectiveness of this approach on a set of hand-written digits.

The idea of separating characters into parts (strokes) has been an integral part of not only how humans recognize characters, but also how we form them. Chan and Nunes [5] have suggested that a number of Asian scripts, in particular Chinese, follows a methodical approach of using strokes to
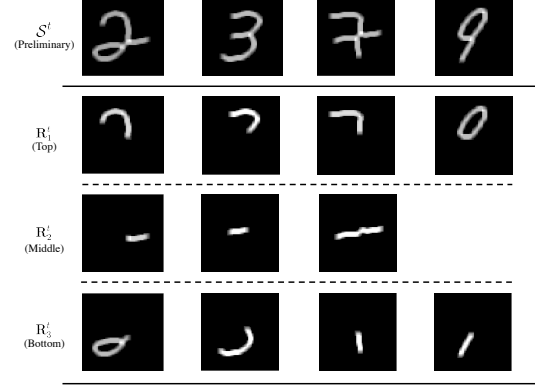
produce characters; these same strokes are also used to aid the recognition of the script. More importantly, strokes are language agnostic as each script can be separated into a set of parts, making them a great mid-level representation for characters. The authors of [19] have also used this cue by learning from a series of strokes produced by online participants. However, human aid in generating the strokes for a character set is often times unavailable and expensive.

The authors of [2] and [3] proposed that the symmetry axis (or skeleton) of an object can be used for shape description. Our algorithm for finding the symmetry axis is based on the work of [24] and [13]. We define the symmetry axis of a character as a skeleton, $\mathcal{A}$, where each pixel $a_i \in \mathcal{A}$ is symmetrically centered between two points, p_i^l and p_i^r, located on either side of a_i.

To find $\mathcal{A}$, we first extract the edges from the binary mask of an image using Sobel operators. We take each point p in the edge image and cast a ray along the gradient (normal) direction d_p to find another point q. We define the corresponding points p and q as the left and right pair of points, p_i^l and p_i^r, that lie on the boundaries of the character. For each pair of p_i^l and p_i^r, we can compute its a_i as the midpoint of p_i^l and p_i^r given by

$$a_i = \frac{1}{2}(p_i^l + p_i^r) \text{ for } a_i \in \mathcal{A} \tag{1}$$

However, results of edge detection are commonly faulty and inconsistent; therefore, we add the additional constraint that the width of the stroke $\|p_i^r - p_i^l\|$ must remain approximately the same. This constraint also allows us to approximate the symmetry axis in the case of missing edge pixels to produce a robust skeleton. Once the preliminary skeleton has been formed, we aggregate sets of end-points and branch-points together in the skeleton to form our set of terminal points. We use Dijkstra's algorithm [11] to find the
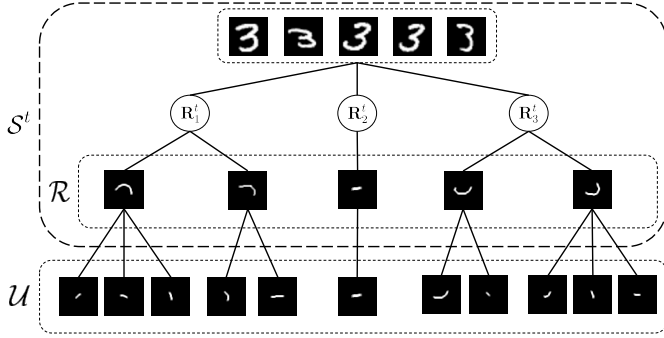
Figure 7. An example of an AND-OR graph representing the digit 3. Each model $\mathcal{S}^t$ is composed of three regions related by a set of AND-relations. Each region is represented as a set of OR-relations amongst meaningful patches $\mathbf{R}_k \in \mathcal{R}$ that was built from low-level segments of $\mathcal{U}$.
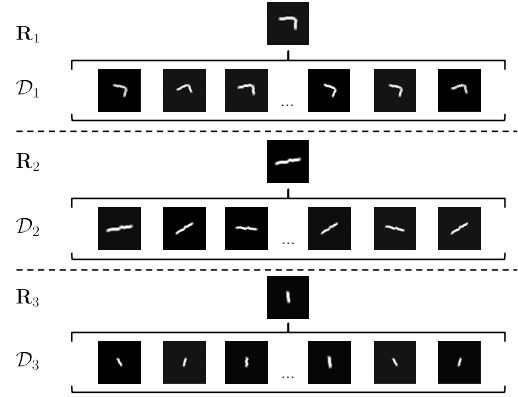

Figure 8. Applying the active patches model to the three regions of a digit 7. Each patch, $\mathbf{R}_k$, representing a region is associated with the set of deformed patches $\mathcal{D}_k$, generated by applying the transformation $T = (s^x, s^y, \theta)$.

shortest path from one terminal point to another, to produce a set of segments. We prune out the small branches connected to complex branch-points (joining 3 or more pixels) to complete our symmetry axis. We center the final product to make it invariant to translation (Fig. 5).

To generate a set of low level features, we first locate the components connected to complex branch-points. Each component is labeled as a separate segment. We compute the gradient direction, ϕ, using Sobel operators on each pixel along the segments. As we traverse the segments of the symmetry axis, we break a segment where there exists a sharp change in ϕ.

The resulting segments are then convolved with a Gaussian kernel, G, and converted into a set of segment patches, $\mathcal{U}$. These patches of stroke segments serve as low-level features representing the character. For each segment patch $\mathbf{s}_i \in \mathcal{U}$, we associate a centroid c_i based on the location of the extracted segment. Each centroid can be computed as the weighted average of intensity, w_j, at each pixel position $\langle x_j, y_j \rangle$ for n pixels, shown below:

$$c_i = \langle \frac{1}{n} \sum^n w_j x_j, \frac{1}{n} \sum^n w_j y_j \rangle \qquad (2)$$

Using the set of segment patches $\mathcal{U}$, our goal is to build a set of larger patches $\mathcal{R}$ that is able to describe the local regions of an object (Fig. 7). These patches will in turn be used as the building blocks for our compositional model. To create a set of meaningful patches that represents the regions of an object, we first define an $M \times N$ grid where M and N are the dimensions of the training image. We select m points on the grid as anchors where each point represents the center of a region in the object. We simply let each segment patch, $\mathbf{s}_i \in \mathcal{U}$, hash to the nearest anchor by measuring the Euclidean distance between its centroid, c_i and the anchor. The patches hashed to a particular region are combined to form a larger patch, $\mathbf{R}_k \in \mathcal{R}$ for $k = 1, 2, 3, ..., m$.

A new centroid, c_k is computed from $\mathbf{R}_k$ and associated with each region patch. In reference to hand-written digits, we denote each of these region patches as a stroke.

3.2. Building a Compositional Model using Patches

For an object t, our goal is to create a generative model that best represents the object as a composition of parts. Given a set of meaningful patches $\mathcal{R}^t$ extracted from the t, we define a compositional model, $\mathcal{S}^t$, as an AND-OR graph that is comprised of $\mathbf{R}_k^t \in \mathcal{R}^t$ where each node in the AND-OR graph is represented as a patch centered at centroid c_k. In order to create a compact model representing a class $\mathcal{S}$, we enable the sharing of knowledge by allowing each model, $\mathcal{S}^t \in \mathcal{S}$, to share parts; any models sharing similar patches are aggregated in a greedy fashion. We measure the similarity between two patches via a match score generated by Normalized Cross Correlation (NCC).

The model, $\mathcal{S}$, for each object class is composed of a set of compositional patch models, $\mathcal{S}^t$, represented by AND-OR graphs. To create such a generative model, we begin by constructing a set of preliminary patch models from each given example (Fig. 6). The structure preliminary model is simply the set of AND-relations joining the set of meaningful patches $\mathbf{R}_k^t \in \mathcal{R}^t$ extracted from an object t:

$$\mathcal{S}^t = (\mathbf{R}_1^t \wedge \mathbf{R}_2^t \wedge \mathbf{R}_3^t \wedge ... \wedge \mathbf{R}_m^t) \text{ for } \mathcal{S}^t \in \mathcal{S} \qquad (3)$$

To create a compact dictionary representing each region, we identify similar patches amongst our set of preliminary models and aggregate those that share resembling parts. For each region $\mathbf{R}_k^t$ in $\mathcal{S}^t$, we apply rotational deformations to generate a small dictionary of templates composed of the deformed patch, $\mathbf{R'}_k^t$, that will be used to match against $\mathbf{R}_k^u$ in another model $\mathcal{S}^u$. We allow each patch to rotate by δ degrees to account for similar patches that are slightly ro-
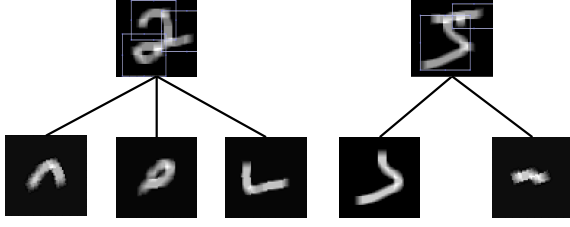
Figure 9. Matching the set of deformed stroke patches in each region, $\mathbf{R}_k$, to the blurred images of skeletonized hand-written digits. Each $\mathbf{D}_j$ matches to a position (x, y) near c_k using Normalized Cross Correlation (NCC). We choose the maximum response given by NCC to ensure the targeted area has minimal error.

tated. We adopt NCC as our method to find the best fit $\mathbf{R}'^t_k$ that matches to the patch $\mathbf{R}^u_k$ by computing a match score γ. Should γ exceed some threshold τ, we merge the two AND-OR graphs together – combining the similar regions and adding OR-relations to the dissimilar regions to produce $\mathcal{S}'^t$. We add the size constraint that a patch $\mathbf{R}'^t_k$ much smaller than $\mathbf{R}^u_k$ cannot be merged together to prevent larger patches from dominating the set. If $\mathcal{S}^t$ and $\mathcal{S}^u$ share the region $\mathbf{R}_k$ then our resulting AND-OR graph (Fig. 7) becomes the following:

$$\mathcal{S}'^t = (\mathbf{R}^t_1 \vee \mathbf{R}^u_1) \wedge ... \wedge \mathbf{R}^t_k \wedge ... \wedge (\mathbf{R}^t_m \vee \mathbf{R}^u_m) \quad (4)$$

Given the set of AND-OR graphs, $\mathcal{S}$, whose similar components has been aggregated, we will apply the active patches model [25] with transformations, T, to each region to generate a dictionary of deformed patches $\mathcal{D}_k$ associated with $\mathbf{R}_k$. We denote T as the set of transformations involving a combination of scaling and rotation of an image patch represented by $T = (s^x, s^y, \theta)$ where s^x and s^y denotes the width and height of the patch after scaling and θ, the angle of rotation. We allow each patch, $\mathbf{R}_k$, to expand and shrink by s pixels and rotate by θ degrees to create a deformed patch $\mathbf{D}_j$ for $j = 1, 2, ..., m$ to produce the set $\mathcal{D}_k$. Each patch in our dictionary of active patches, $\mathcal{D}_j$, maps to a single patch $\mathbf{R}_k$ (Fig. 8). Our model thus becomes the set of and-or-relations of regions, where each region corresponds to a dictionary of active patches.

3.3. Applying the Compositional Model to New Images

Given a new $M \times N$ image, $\mathbf{I}$, we allow our stroke models, $\mathcal{S}$, to propose the best set of reconstructions for $\mathbf{I}$ based on the active patches dictionaries associated to the regions of each model. We measure the goodness of fit for each proposal by computing a cost function that accounts for similarity and coverage between the shapes of the proposal and a processed $\mathbf{I}$. We find the best fit proposal from each class by minimizing a cost function and amongst those select the
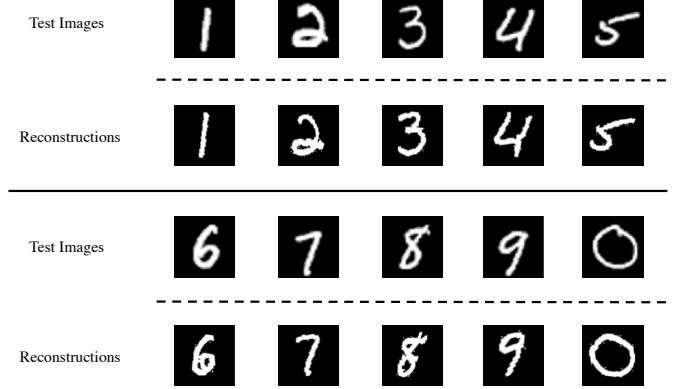

Figure 10. Examples of reconstructed images that were selected as the best fit proposal for a given hand-written digit test image. The reconstructions were fine-tuned by applying the transformations from Shape Context to adjust for variable affine transformations.

top candidates. We compute the transformation between the shapes of candidates and our processed test image via Shape Context, [1]. We warp the candidates to better fit our test image and minimize an energy function to find the best reconstruction, selecting its class as our label.

We begin by finding the symmetry axis in image, $\mathbf{I}$, using the approach described in Sec. 3.1. The skeleton of $\mathbf{I}$ is then convolved with a Gaussian kernel, G, to produce a composite image $\mathbf{I}'$ that is consistent with the patches in our dictionary. We use NCC to find the best fit patch to a region in $\mathbf{I}'$ – a higher NCC score implies a better fit (Fig. 9). We allow each stroke model to make proposals for a crude reconstruction of $\mathbf{I}'$ by computing a match score between each deformed patch $\mathbf{D}_j$ and $\mathbf{I}'$ to represent each region $\mathbf{R}_k$ in our stroke model. We choose the optimal patch, $\hat{\mathbf{R}}_k$ amongst the set of deformed patches, $\mathbf{D}_j \in \mathcal{D}_k$ associated via a set of OR-relations by choosing the patch with the maximal response from NCC. We add the constraint that a match is only valid if it occurs near the centroid, c_k.

$$\hat{\mathbf{R}}_k = \arg \max_{\mathbf{D}_j \in \mathcal{D}_k} NCC(\mathbf{D}_j, \mathbf{I}'_{c_k}) \quad (5)$$

The reconstruction, $\mathbf{P}^t$, proposed by our and-or graph, $\mathcal{S}^t$, is the set of AND-relations composed of the optimal patch, $\hat{\mathbf{R}}_k$, representing each region. We define $\mathcal{P}$ as our set of propositions generated by each stroke model $\mathcal{S}^t$.

$$\mathbf{P}^t = (\hat{\mathbf{R}}_1 \wedge \hat{\mathbf{R}}_2 \wedge \hat{\mathbf{R}}_3 \wedge ... \wedge \hat{\mathbf{R}}_m) \text{ for } \mathbf{P}^t \in \mathcal{P} \quad (6)$$

To choose the best reconstruction from each label, we minimize a cost function, f, between each proposal $\mathbf{P}^t$ and the image, $\mathbf{I}'$, incorporating similarity and coverage.

$$f(\mathbf{B}^{P_t}, X, \mathbf{B}^{I'}, Y) = d_H(X, Y) \times SSD(\mathbf{B}^{P_t}, \mathbf{B}^{I'}) \quad (7)$$
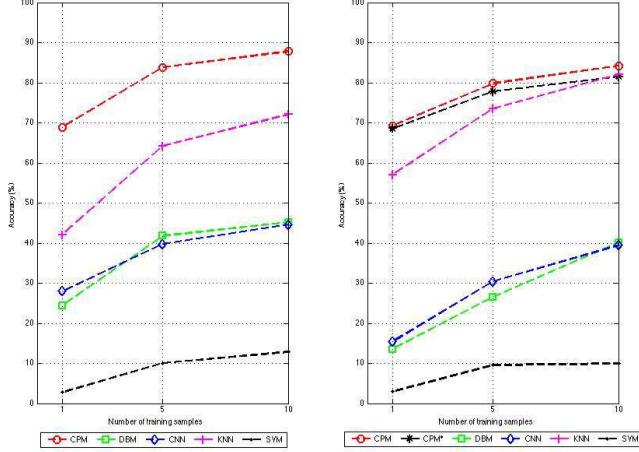
Figure 11. Training on 1, 5, and 10 examples for each class from MNIST (left) and USPS (right). Our compositional patch model (CPM) consistently outperforms other methods on one shot digit recognition. CPM* denotes the compositional patch model that was trained on MNIST and used for testing on USPS.

Method	MNIST n=5	MNIST n=1	USPS n=5	USPS n=1
CPM	**83.79**	**68.86**	**79.88**	**69.31**
CPM*	-	-	77.81	68.58
DBM	41.76	24.37	26.60	13.56
CNN	39.80	28.01	30.42	15.37
K-NN	64.26	42.08	73.59	56.98
SVM	10.08	2.78	9.55	2.93

Table 1. One shot performances of methods compared on MNIST and USPS hand-written digits datasets. The results are averaged over 15 runs. CPM* demonstrates that our method is transferable when learned on MNIST and tested on USPS.

We model the similarities between the two image as a shape comparison problem. To compute the coverage between $\mathbf{P}^t$ and $\mathbf{I}'$, we create a binary mask of the two images, $\mathbf{B}^{P_t}, \mathbf{B}^{I'} \in [0,1]^{M \times N}$, respectively. We then take the Sum of Squared Distances (SSD) between the two masks to find the number of mismatched pixels. We measure the shape similarity between $\mathbf{P}_t$ and $\mathbf{I}'$ using Hausdorff distance [16] as our metric. We computed the edge image of $\mathbf{B}^{P_t}$ and $\mathbf{B}^{I'}$ to produce the set of edge pixels X and Y to determine the Hausdorff distance (d_H) between the two sets of points. Due to the nature of Active Patches and NCC matching, our $\mathbf{B}^{P_t}$ and $\mathbf{B}^{I'}$ are closely aligned and similarly for the points in X and Y.

We define the set of top proposals from each class as the set $\hat{\mathcal{P}}$. We compute the transformation between the binary masks of each top proposal, $\mathbf{B}^{P_t} \in \hat{\mathcal{P}}$, and the image, $\mathbf{B}^{I'}$ via Shape Context. We then refine our crude reconstructions of $\mathbf{I}'$ by warping each $\mathbf{B}^{P_t}$ by their respectively transformations to produce $\mathbf{B}_w^{P_t}$. We define the affine cost, $\alpha^{\mathcal{P}_t}$, of Shape Context as the cost to warp $\mathbf{B}^{P_t}$ to $\mathbf{B}_w^{P_t}$. We finally compute the energy function E for reconstructing $\mathbf{I}'$ as the product of the SSD between $\mathbf{B}_w^{P_t}$ and $\mathbf{B}^{I'}$ and the cost of transformation, α.

$$E(\mathbf{B}_w^{P_t}, \mathbf{B}^{I'}) = SSD(\mathbf{B}_w^{P_t}, \mathbf{B}^{I'})(1 + \alpha^{\mathcal{P}_t}) \quad (8)$$

We select the the label for the test image, $\mathbf{I}$, by choosing the class with the best reconstruction that minimizes E (Fig. 10).

4. Implementation Details

The following section describes the set of parameters used in our experiments. We begin with a preprocessing step of resizing all images to 56×56 as this yields better edge detection results for computing the Symmetry Axis. When decomposing characters into strokes in Sec. 3.1, we break a stroke if the stroke experiences a sharp change in gradient direction where $\phi > 90°$. We also use a Gaussian filter, G, with $\sigma = 4$ and a window size of $[3, 3]$ to produced the set of stroke patches after extracting the low level stroke segments from each character.

We used a 56×56 grid in Sec. 3.2 and selected the number of anchors, m, to be 3 where each is located at $\{[19, 28], [28.5, 28], [38, 28]\}$. This is based on the observation that the each example in MNIST dataset can intuitively be separated into 3 regions. To produce a compact model, we allow each stroke to vary by $-10° < \delta < 10°$ and we merge two stroke models if the match score, γ, from NCC exceeds a threshold $\tau = 0.70$. Once the stroke models have been aggregated, we defined a set of transformations to produced our active patches for the set of rotations $-15° < \theta < 15°$ with increments of $7.5°$. The adopted widths and heights for scaling ranges between -10 to 10 pixels with increments of 5 pixels.

For Shape Context described in Sec. 3.3, we computed the shape transformations between our reconstructions and the test image using 5 iterations with a minimum of 85 sample points of correspondences and an annealing rate of 1.

Our experiments were run on an Intel processor with 8 cores and 32GB of physical memory, but our training procedures involves mostly inexpensive computations, which allow us to train the same model on a conventional laptop. Training takes 1.44 and 5.23 seconds for 1 and 5 samples, respectively, on an Intel 2.26 GHz Core 2 Duo machine with 4GB of memory. With a short training time using few examples, our framework is well-suited to learning (new) characters online on memory and computationally constrained devices (e.g. mobile, embedded), a space where state of the art

methods may be computationally prohibitive—DBM takes approximately 9 and 20 minutes, respectively, to train on 1 and 5 examples on the laptop. An optimized implementation of our work could permit this in real-time.

5. Experimental Results

We tested five models on one shot learning: our compositional patch model (CPM), k-Nearest Neighbors(K-NN), Support Vector Machines (SVM), Convolution Neural Network (CNN), Deep Boltzmann Machines (DBM).The performances were evaluated on a 10-way classification where each class is provided with 1, 5, and 10 training examples to show the growth in accuracy. The models were tested on two hand-written datasets: MNIST and USPS.For a given run, each model is given a set of hand-written digits picked at random from each class. In addition, we also provide experiments showing the transferability of the stroke model by training on MNIST and testing on USPS.

The implementation of K-NN and SVM is based on that of VL Feat Toolbox [28]. Specifically, our K-NN approach is constructed using a single kd-tree. For CNN, we used the implementation of MatConvNet provided by [29] with four convolutional layers and two pooling layers. For DBM, we use the implementation provided by [27], which contains two hidden layers with 1000 units each. We tested CNN and DBM using 200 and 300 epochs, respectively, and the epoch with the maximum score is used for the results of each run.

The results of our experiments are summarized by Table 1 and Fig. 11, averaged over 15 runs. Our compositional model consistently outperforms other methods on the one-shot learning task. Without the use of Shape Context (in order to fine tune the reconstructions), our model averages 78.11% on MNIST with five examples. In contrast, the traditional methods are generally unable to achieve high recognition accuracies with so few examples, save for K-NN, which performs well on USPS largely due to the low dimensionality of the dataset. Even our transferable model CMP* (trained on MNIST and tested on USPS) outperforms the comparison approaches. While our model currently achieves mid-80% accuracy with five examples, the parameters used are not optimal. A systematic parameter search would yield greater quantitative scores.

In addition to the parameters provided in Sec. 4, we tried increasing the number of iterations and the number of correspondences for Shape Context. We found that the results did not differ by more than 1–2%. In general, more correspondences and iterations tend to yield higher accuracies. However, recognition time similarly increase due to the use of the Hungarian algorithm [18] in Shape Context. Although our method extracts a set of meaningful patches representing the general structures of objects, it is difficult to predict all of the variations that will exist in novel images. Gener-
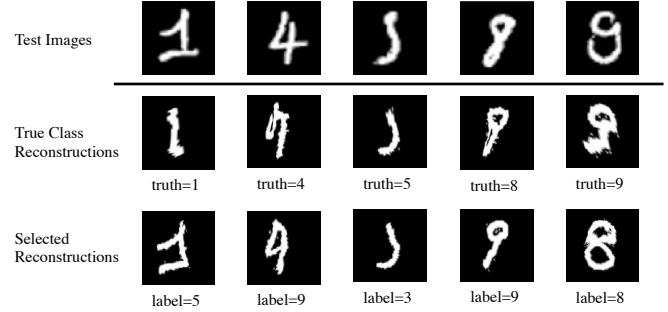

Figure 12. Examples of mis-classifications due to variations in the test image being too far from the limited training set causing affine cost $\alpha^{\mathcal{P}_t}$ to become extremely large.

ally, misclassifications occur in examples that have specific regions missing from the objects in our training set (Fig. 12), causing the warping costs to significantly increase.

6. Discussion

This paper introduces a technique to produce a compact dictionary of meaningful patches from visual concepts by segmenting the objects into parts. We also present a generative patch-based model that mimics the construction of these concepts by relating the set of parts that composes them. Given a new object in an image, the model attempts to reconstruct the object of interest based on a set of deformed patches learned from a small set of examples. This method performs well on the one-shot learning task of handwritten digit recognition, beating popular algorithms by a wide margin.

Our method, however, is far from human-level competence. As illustrated in Fig. 12, our approach still makes mistakes. In addition, although we boast a fast training time, we use 2.86 seconds to perform recognition on a new image at test time on the workstation in Sec. 4. This could be reduced by restricting the number of correspondences used for Shape Context or by utilizing GPUs to compute the NCC score between patches and images [25]

Nevertheless, our method has proven an effective framework for object recognition using a small set of training examples. Future interesting directions include exploring the robustness of our model in recognizing objects in novel examples with noise, significant occlusion, or even in the wild. Given the fast training time of our approach and the need for so few examples, we are also interested in applying this method in memory and computationally constrained settings such as mobile devices for real-time uses. These are all future directions that we will explore given the promising results of our current algorithm.

References

[1] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:509–522, 2002. 6

[2] H. Blum. Biological shape and visual science (part i). *Journal of theoretical Biology*, 38(2):205–287, 1973. 4

[3] H. Blum and R. N. Nagel. Shape description using weighted symmetric axis features. *Pattern recognition*, 10(3):167–180, 1978. 4

[4] S. Carey and E. Bartlett. *Acquiring a single new word*. ERIC, 1978. 1

[5] L. Chan and T. Nunes. Children's understanding of the formal and functional characteristics of written chinese. *Applied Psycholinguistics*, 19(01):115–131, 1998. 4

[6] L. Cheng-Lin, N. Kazuki, S. Hiroshi, and F. , Hiromichi. Handwritten digit recognition: benchmarking of state-of-the-art techniques. *Pattern Recognition*, 36(10):2271–2285, 2003. 2

[7] D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012. 2

[8] E. V. Clark. *First language acquisition*. Cambridge University Press, 2009. 1

[9] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. 2

[10] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967. 2

[11] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959. 4

[12] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 341–346. ACM, 2001. 3

[13] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2963–2970. IEEE, 2010. 4

[14] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(4):594–611, 2006. 2

[15] J. J. Hull. A database for handwritten text recognition research. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(5):550–554, 1994. 3

[16] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing images using the hausdorff distance. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(9):850–863, 1993. 7

[17] Y. Jin and S. Geman. Context and hierarchy in a probabilistic image model. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2145–2152. IEEE, 2006. 1

[18] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 8

[19] B. M. Lake, R. Salakhutdinov, J. Gross, and J. B. Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, volume 172, 2011. 1, 2, 3, 4

[20] B. M. Lake, R. R. Salakhutdinov, and J. Tenenbaum. One-shot learning by inverting a compositional causal process. In *Advances in neural information processing systems*, pages 2526–2534, 2013. 2, 3

[21] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. 2

[22] Y. LeCun and C. Cortes. The mnist database of handwritten digits, 1998. 2, 3

[23] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum. Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics (ToG)*, 20(3):127–150, 2001. 3

[24] T.-L. Liu, D. Geiger, and A. L. Yuille. Segmenting by seeking the symmetry axis. In *Pattern Recognition, International Conference on*, volume 2, pages 994–994. IEEE Computer Society, 1998. 1, 4

[25] J. Mao, J. Zhu, and A. L. Yuille. An active patch model for real world texture and appearance classification. In *Computer Vision–ECCV 2014*, pages 140–155. Springer, 2014. 3, 6, 8

[26] E. G. Miller, N. E. Matsakis, and P. A. Viola. Learning from one example through shared densities on transforms. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 464–471. IEEE, 2000. 2

[27] R. Salakhutdinov and G. E. Hinton. Deep boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, pages 448–455, 2009. 2, 8

[28] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008. 8

[29] A. Vedaldi and K. Lenc. Matconvnet-convolutional neural networks for matlab. *arXiv preprint arXiv:1412.4564*, 2014. 8

[30] X. Ye and A. Yuille. Learning a dictionary of deformable patches using gpus. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 483–490. IEEE, 2011. 3

[31] S. C. Zhu and A. L. Yuille. Forms: a flexible object recognition and modelling system. *International Journal of Computer Vision*, 20(3):187–212, 1996. 2