Siamese Neural Networks for One-shot Image Recognition

Gregory Koch GKOCH@CS.TORONTO.EDU
Richard Zemel ZEMEL@CS.TORONTO.EDU
Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Department of Computer Science, University of Toronto. Toronto, Ontario, Canada.

Abstract

The process of learning good features for machine learning applications can be very computationally expensive and may prove difficult in cases where little data is available. A prototypical example of this is the *one-shot learning* setting, in which we must correctly make predictions given only a single example of each new class. In this paper, we explore a method for learning *siamese neural networks* which employ a unique structure to naturally rank similarity between inputs. Once a network has been tuned, we can then capitalize on powerful discriminative features to generalize the predictive power of the network not just to new data, but to entirely new classes from unknown distributions. Using a convolutional architecture, we are able to achieve strong results which exceed those of other deep learning models with near state-of-the-art performance on one-shot classification tasks.

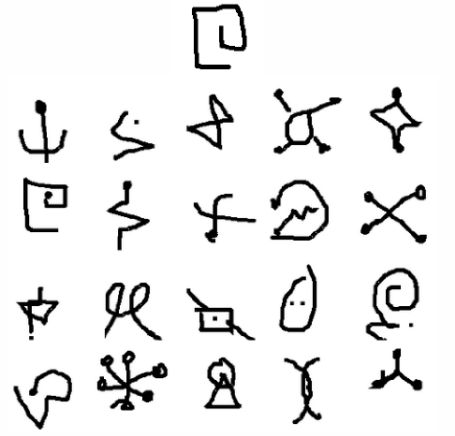

Figure 1. Example of a 20-way one-shot classification task using the Omniglot dataset. The lone test image is shown above the grid of 20 images representing the possible unseen classes that we can choose for the test image. These 20 images are our only known examples of each of those classes.

Humans exhibit a strong ability to acquire and recognize new patterns. In particular, we observe that when presented with stimuli, people seem to be able to understand new concepts quickly and then recognize variations on these concepts in future percepts (Lake et al., 2011). Machine learning has been successfully used to achieve state-of-the-art performance in a variety of applications such as web search, spam detection, caption generation, and speech and image recognition. However, these algorithms often break down when forced to make predictions about data for which little supervised information is available. We desire to generalize to these unfamiliar categories without necessitating extensive retraining which may be either expensive or impossible due to limited data or in an online prediction setting, such as web retrieval.

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

One particularly interesting task is classification under the restriction that we may only observe a single example of each possible class before making a prediction about a test instance. This is called *one-shot learning* and it is the primary focus of our model presented in this work (Fei-Fei et al., 2006; Lake et al., 2011). This should be distinguished from *zero-shot learning*, in which the model cannot look at any examples from the target classes (Palatucci et al., 2009).

One-shot learning can be directly addressed by developing domain-specific features or inference procedures which possess highly discriminative properties for the target task. As a result, systems which incorporate these methods tend to excel at similar instances but fail to offer robust solutions that may be applied to other types of problems. In this paper, we present a novel approach which limits assumptions on the structure of the inputs while automatically acquiring features which enable the model to generalize successfully from few examples. We build upon the deep learn-

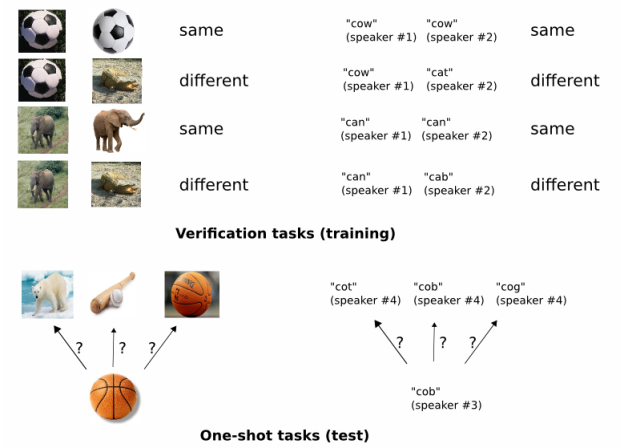

Figure 2. Our general strategy. 1) Train a model to discriminate between a collection of same/different pairs. 2) Generalize to evaluate new categories based on learned feature mappings for verification.

ing framework, which uses many layers of non-linearities to capture invariances to transformation in the input space, usually by leveraging a model with many parameters and then using a large amount of data to prevent overfitting (Bengio, 2009; Hinton et al., 2006). These features are very powerful because we are able to learn them without imposing strong priors, although the cost of the learning algorithm itself may be considerable.

1. Approach

In general, we learn image representations via a supervised metric-based approach with siamese neural networks, then reuse that network's features for one-shot learning without any retraining.

In our experiments, we restrict our attention to character recognition, although the basic approach can be replicated for almost any modality (Figure 2). For this domain, we employ large siamese convolutional neural networks which **a)** are capable of learning generic image features useful for making predictions about unknown class distributions even when very few examples from these new distributions are available; **b)** are easily trained using standard optimization techniques on pairs sampled from the source data; and **c)** provide a competitive approach that does not rely upon domain-specific knowledge by instead exploiting deep learning techniques.

To develop a model for one-shot image classification, we aim to first learn a neural network that can discriminate between the class-identity of image pairs, which is the standard *verification* task for image recognition. We hypothesize that networks which do well at at verification should generalize to one-shot classification. The verification model learns to identify input pairs according to the probability that they belong to the same class or different classes. This model can then be used to evaluate new images, exactly one per novel class, in a pairwise manner against the test image. The pairing with the highest score according to the verification network is then awarded the highest probability for the one-shot task. If the features learned by the verification model are sufficient to confirm or deny the identity of characters from one set of alphabets, then they ought to be sufficient for other alphabets, provided that the model has been exposed to a variety of alphabets to encourage variance amongst the learned features.

2. Related Work

Overall, research into one-shot learning algorithms is fairly immature and has received limited attention by the machine learning community. There are nevertheless a few key lines of work which precede this paper.

The seminal work towards one-shot learning dates back to the early 2000's with work by Li Fei-Fei et al. The authors developed a variational Bayesian framework for one-shot image classification using the premise that previously learned classes can be leveraged to help forecast future ones when very few examples are available from a given class (Fe-Fei et al., 2003; Fei-Fei et al., 2006). More recently, Lake et al. approached the problem of one-shot learning from the point of view of cognitive science, addressing one-shot learning for character recognition with a method called Hierarchical Bayesian Program Learning (HBPL) (2013). In a series of several papers, the authors modeled the process of drawing characters generatively to decompose the image into small pieces (Lake et al., 2011; 2012). The goal of HBPL is to determine a structural explanation for the observed pixels. However, inference under HBPL is difficult since the joint parameter space is very large, leading to an intractable integration problem.

Some researchers have considered other modalities or transfer learning approaches. Lake et al. have some very recent work which uses a generative Hierarchical Hidden Markov model for speech primitives combined with a Bayesian inference procedure to recognize new words by unknown speakers (2014). Maas and Kemp have some of the only published work using Bayesian networks to predict attributes for Ellis Island passenger data (2009). Wu and Dennis address one-shot learning in the context of path planning algorithms for robotic actuation (2012). Lim focuses on how to "borrow" examples from other classes in the training set by adapting a measure of how much each category should be weighted by each training exemplar in the loss function (2012). This idea can be useful for data

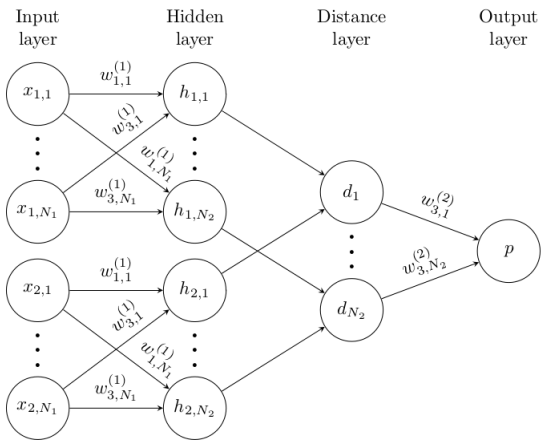

Figure 3. A simple 2 hidden layer siamese network for binary classification with logistic prediction p. The structure of the network is replicated across the top and bottom sections to form twin networks, with shared weight matrices at each layer.

sets where very few examples exist for some classes, providing a flexible and continuous means of incorporating inter-class information into the model.

3. Deep Siamese Networks for Image Verification

Siamese nets were first introduced in the early 1990s by Bromley and LeCun to solve signature verification as an image matching problem (Bromley et al., 1993). A siamese neural network consists of twin networks which accept distinct inputs but are joined by an energy function at the top. This function computes some metric between the highest-level feature representation on each side (Figure 3). The parameters between the twin networks are tied. Weight tying guarantees that two extremely similar images could not possibly be mapped by their respective networks to very different locations in feature space because each network computes the same function. Also, the network is symmetric, so that whenever we present two distinct images to the twin networks, the top conjoining layer will compute the same metric as if we were to we present the same two images but to the opposite twins.

In LeCun et al., the authors used a contrastive energy function which contained dual terms to decrease the energy of like pairs and increase the energy of unlike pairs (2005). However, in this paper we use the weighted L_1 distance between the twin feature vectors $\mathbf{h}_1$ and $\mathbf{h}_2$ combined with a sigmoid activation, which maps onto the interval $[0, 1]$. Thus a *cross-entropy* objective is a natural choice for training the network. Note that in LeCun et al., they directly learned the similarity metric, which was implictly defined by the energy loss, whereas we fix the metric as specified above, following the approach in Facebook's DeepFace paper (Taigman et al., 2014).

Our best-performing models use multiple convolutional layers before the fully-connected layers and top-level energy function. Convolutional neural networks have achieved exceptional results in many large-scale computer vision applications, particularly in image recognition tasks (Bengio, 2009; Krizhevsky et al., 2012; Simonyan & Zisserman, 2014; Srivastava, 2013).

Several factors make convolutional networks especially appealing. Local connectivity can greatly reduce the number of parameters in the model, which inherently provides some form of built-in regularization, although convolutional layers are computationally more expensive than standard nonlinearities. Also, the convolution operation used in these networks has a direct filtering interpretation, where each feature map is convolved against input features to identify patterns as groupings of pixels. Thus, the outputs of each convolutional layer correspond to important spatial features in the original input space and offer some robustness to simple transforms. Finally, very fast CUDA libraries are now available in order to build large convolutional networks without an unacceptable amount of training time (Mnih, 2009; Krizhevsky et al., 2012; Simonyan & Zisserman, 2014).

We now detail both the structure of the siamese nets and the specifics of the learning algorithm used in our experiments.

3.1. Model

Our standard model is a siamese convolutional neural network with L layers each with N_l units, where $\mathbf{h}_{1,l}$ represents the hidden vector in layer l for the first twin, and $\mathbf{h}_{2,l}$ denotes the same for the second twin. We use exclusively rectified linear (ReLU) units in the first $L-2$ layers and sigmoidal units in the remaining layers.

The model consists of a sequence of convolutional layers, each of which uses a single channel with filters of varying size and a fixed stride of 1. The number of convolutional filters is specified as a multiple of 16 to optimize performance. The network applies a ReLU activation function to the output feature maps, optionally followed by max-pooling with a filter size and stride of 2. Thus the kth filter map in each layer takes the following form:

$$a^{(k)}_{1,m} = \text{max-pool}(\max(0, \mathbf{W}^{(k)}_{l-1,l} \star \mathbf{h}_{1,(l-1)} + \mathbf{b}_l), 2)$$
$$a^{(k)}_{2,m} = \text{max-pool}(\max(0, \mathbf{W}^{(k)}_{l-1,l} \star \mathbf{h}_{2,(l-1)} + \mathbf{b}_l), 2)$$

where $\mathbf{W}_{l-1,l}$ is the 3-dimensional tensor representing the feature maps for layer l and we have taken $\star$ to be the *valid* convolutional operation corresponding to returning

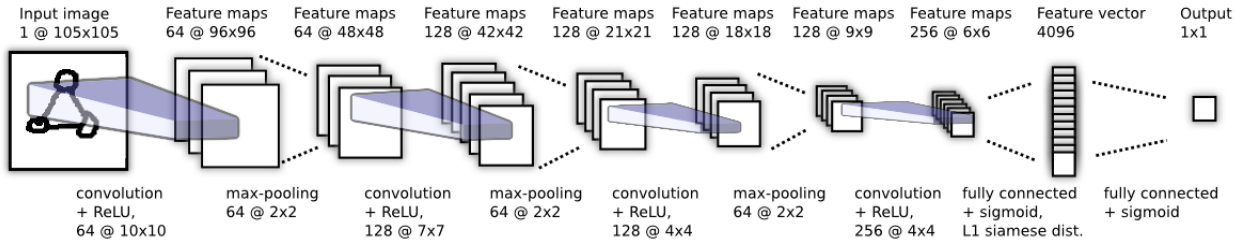

Figure 4. Best convolutional architecture selected for verification task. Siamese twin is not depicted, but joins immediately after the 4096 unit fully-connected layer where the L1 component-wise distance between vectors is computed.

only those output units which were the result of complete overlap between each convolutional filter and the input feature maps.

The units in the final convolutional layer are flattened into a single vector. This convolutional layer is followed by a fully-connected layer, and then one more layer computing the induced distance metric between each siamese twin, which is given to a single sigmoidal output unit. More precisely, the prediction vector is given as $\mathbf{p} = \sigma(\sum_j \alpha_j |\mathbf{h}^{(j)}_{1,L-1} - \mathbf{h}^{(j)}_{2,L-1}|)$, where σ is the sigmoidal activation function. This final layer induces a metric on the learned feature space of the $(L-1)$th hidden layer and scores the similarity between the two feature vectors. The α_j are additional parameters that are learned by the model during training, weighting the importance of the component-wise distance. This defines a final Lth fully-connected layer for the network which joins the two siamese twins.

We depict one example above (Figure 4), which shows the largest version of our model that we considered. This network also gave the best result for any network on the verification task.

3.2. Learning

Loss function. Let M represent the minibatch size, where i indexes the ith minibatch. Now let $\mathbf{y}(x_1^{(i)}, x_2^{(i)})$ be a length-M vector which contains the labels for the minibatch, where we assume $y(x_1^{(i)}, x_2^{(i)}) = 1$ whenever x_1 and x_2 are from the same character class and $y(x_1^{(i)}, x_2^{(i)}) = 0$ otherwise. We impose a regularized cross-entropy objective on our binary classifier of the following form:

$$\mathcal{L}(x_1^{(i)}, x_2^{(i)}) = \mathbf{y}(x_1^{(i)}, x_2^{(i)}) \log \mathbf{p}(x_1^{(i)}, x_2^{(i)}) + (1 - \mathbf{y}(x_1^{(i)}, x_2^{(i)})) \log (1 - \mathbf{p}(x_1^{(i)}, x_2^{(i)})) + \boldsymbol{\lambda}^T |\mathbf{w}|^2 \longrightarrow \min$$

Optimization. This objective is combined with standard backpropagation algorithm, where the gradient is additive across the twin networks due to the tied weights. We fix a minibatch size of 128 with learning rate η_j, momentum μ_j, and L_2 regularization weights λ_j defined layer-wise, so that our update rule at epoch T is as follows:

$$\mathbf{w}^{(T)}_{kj}(x_1^{(i)}, x_2^{(i)}) = \mathbf{w}^{(T)}_{kj} + \Delta\mathbf{w}^{(T)}_{kj}(x_1^{(i)}, x_2^{(i)}) + 2\lambda_j |\mathbf{w}_{kj}|$$
$$\Delta\mathbf{w}^{(T)}_{kj}(x_1^{(i)}, x_2^{(i)}) = -\eta_j \nabla w^{(T)}_{kj} + \mu_j \Delta\mathbf{w}^{(T-1)}_{kj}$$

where ∇w_{kj} is the partial derivative with respect to the weight between the jth neuron in some layer and the kth neuron in the successive layer.

Weight initialization. We initialized all network weights in the convolutional layers from a normal distribution with zero-mean and a standard deviation of 10^{-2}. Biases were also initialized from a normal distribution, but with mean 0.5 and standard deviation 10^{-2}. In the fully-connected layers, the biases were initialized in the same way as the convolutional layers, but the weights were drawn from a much wider normal distribution with zero-mean and standard deviation 2×10^{-1}.

Learning schedule. Although we allowed for a different learning rate for each layer, learning rates were decayed uniformly across the network by 1 percent per epoch, so that $\eta_j^{(T)} = 0.99\eta_j^{(T-1)}$. We found that by annealing the learning rate, the network was able to converge to local minima more easily without getting stuck in the error surface. We fixed momentum to start at 0.5 in every layer, increasing linearly each epoch until reaching the value μ_j, the individual momentum term for the jth layer.

We trained each network for a maximum of 200 epochs, but monitored *one-shot validation error* on a set of 320 one-shot learning tasks generated randomly from the alphabets and drawers in the validation set. When the validation error did not decrease for 20 epochs, we stopped and used the parameters of the model at the best epoch according to the one-shot validation error. If the validation error continued to decrease for the entire learning schedule, we saved the final state of the model generated by this procedure.

Hyperparameter optimization. We used the beta version of Whetlab, a Bayesian optimization framework, to

$P(x_1^{(i)}, x_2^{(i)}) = \sigma\left(\sum \alpha_j |h_{1j} - h_{2j}|\right)$
if $x_1^{(i)} = x_1^{(i)} \Rightarrow P(x_1, x_2) = \frac{1}{2}$
$\log \frac{1}{2} = -1$
If $\sum \alpha_j |h_{1j} - h_{2j}| \to \infty \Rightarrow P(x_1, x_2) \to 1 \Rightarrow \log P(x_1, x_2) \to 0$

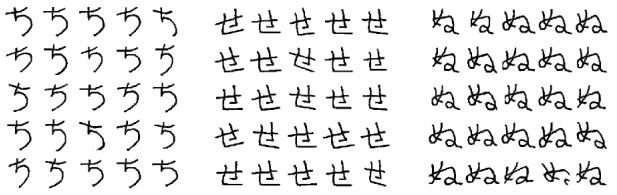

Figure 5. A sample of random affine distortions generated for a single character in the Omniglot data set.

Figure 6. The Omniglot dataset contains a variety of different images from alphabets across the world.

perform hyperparameter selection. For learning schedule and regularization hyperparameters, we set the layer-wise learning rate $\eta_j \in [10^{-4}, 10^{-1}]$, layer-wise momentum $\mu_j \in [0, 1]$, and layer-wise L_2 regularization penalty $\lambda_j \in [0, 0.1]$. For network hyperparameters, we let the size of convolutional filters vary from 3x3 to 20x20, while the number of convolutional filters in each layer varied from 16 to 256 using multiples of 16. Fully-connected layers ranged from 128 to 4096 units, also in multiples of 16. We set the optimizer to maximize one-shot validation set accuracy. The score assigned to a single Whetlab iteration was the highest value of this metric found during any epoch.

Affine distortions. In addition, we augmented the training set with small affine distortions (Figure 5). For each image pair $\mathbf{x_1}, \mathbf{x_2}$, we generated a pair of affine transformations T_1, T_2 to yield $\mathbf{x'_1} = T_1(\mathbf{x_1}), \mathbf{x'_2} = T_2(\mathbf{x_2})$, where T_1, T_2 are determined stochastically by a multi-dimensional uniform distribution. So for an arbitrary transform T, we have $T = (\theta, \rho_x, \rho_y, s_x, s_y, t_x, t_x)$, with $\theta \in [-10.0, 10.0], \rho_x, \rho_y \in [-0.3, 0.3], s_x, s_y \in [0.8, 1.2]$, and $t_x, t_y \in [-2, 2]$. Each of these components of the transformation is included with probability 0.5.

4. Experiments

We trained our model on a subset of the Omniglot data set, which we first describe. We then provide details with respect to verification and one-shot performance.

4.1. The Omniglot Dataset

The Omniglot data set was collected by Brenden Lake and his collaborators at MIT via Amazon's Mechanical Turk to produce a standard benchmark for learning from few examples in the handwritten character recognition domain (Lake et al., 2011).[1] Omniglot contains examples from 50 alphabets ranging from well-established international languages like Latin and Korean to lesser known local dialects. It also includes some fictitious character sets such as Aurek-Besh and Klingon (Figure 6).

The number of letters in each alphabet varies considerably from about 15 to upwards of 40 characters. All characters across these alphabets are produced a single time by each of 20 drawers Lake split the data into a 40 alphabet *background set* and a 10 alphabet *evaluation set*. We preserve these two terms in order to distinguish from the normal training, validation, and test sets that can be generated from the background set in order to tune models for verification. The background set is used for developing a model by learning hyperparameters and feature mappings. Conversely, the evaluation set is used only to measure the one-shot classification performance.

4.2. Verification

To train our verification network, we put together three different data set sizes with 30,000, 90,000, and 150,000 training examples by sampling random *same* and *different* pairs. We set aside sixty percent of the total data for training: 30 alphabets out of 50 and 12 drawers out of 20.

We fixed a uniform number of training examples per alphabet so that each alphabet receives equal representation during optimization, although this is not guaranteed to the individual character classes within each alphabet. By adding

Table 1. Accuracy on Omniglot verification task (siamese convolutional neural net)

Method	Test
30k training	
no distortions	90.61
affine distortions x8	91.90
90k training	
no distortions	91.54
affine distortions x8	93.15
150k training	
no distortions	91.63
affine distortions x8	**93.42**

[1] The complete data set can be obtained from Brenden Lake by request (`brenden@cs.nyu.edu`). Each character in Omniglot is a 105x105 binary-valued image which was drawn by hand on an online canvas. The stroke trajectories were collected alongside the composite images, so it is possible to incorporate temporal and structural information into models trained on Omniglot.

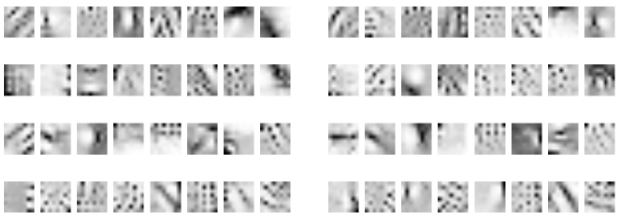

Figure 7. Examples of first-layer convolutional filters learned by the siamese network. Notice filters adapt different roles: some look for very small point-wise features whereas others function like larger-scale edge detectors.

affine distortions, we also produced an additional copy of the data set corresponding to the augmented version of each of these sizes. We added eight transforms for each training example, so the corresponding data sets have 270,000, 810,000, and 1,350,000 effective examples.

To monitor performance during training, we used two strategies. First, we created a validation set for verification with 10,000 example pairs taken from 10 alphabets and 4 additional drawers. We reserved the last 10 alphabets and 4 drawers for testing, where we constrained these to be the same ones used in Lake et al. (Lake et al., 2013). Our other strategy leveraged the same alphabets and drawers to generate a set of 320 one-shot recognition trials for the validation set which mimic the target task on the evaluation set. In practice, this second method of determining when to stop was at least as effective as the validation error for the verification task so we used it as our termination criterion.

In the table below (Table 1), we list the final verification results for each of the six possible training sets, where the listed test accuracy is reported at the best validation checkpoint and threshold. We report results across six different training runs, varying the training set size and toggling distortions.

In Figure 7, we have extracted the first 32 filters from both of our top two performing networks on the verification task, which were trained on the 90k and 150k data sets with affine distortions and the architecture shown in Figure 3. While there is some co-adaptation between filters, it is easy to see that some of the filters have assumed different roles with respect to the original input space.

4.3. One-shot Learning

Once we have optimized a siamese network to master the verification task, we are ready to demonstrate the discriminative potential of our learned features at one-shot learning. Suppose we are given a test image $\mathbf{x}$, some column vector which we wish to classify into one of C categories. We are also given some other images $\{\mathbf{x}_c\}_{c=1}^{C}$, a set of column vectors representing examples of each of those C categories. We can now query the network using $\mathbf{x}, \mathbf{x}_c$ as our input for a range of $c = 1, \ldots, C$.[2] Then predict the class corresponding to the maximum similarity.

$$C^* = \text{argmax}_c \mathbf{p}^{(c)}$$

Table 2. Comparing best one-shot accuracy from each type of network against baselines.

Method	Test
Humans	95.5
Hierarchical Bayesian Program Learning	95.2
Affine model	81.8
Hierarchical Deep	65.2
Deep Boltzmann Machine	62.0
Simple Stroke	35.2
1-Nearest Neighbor	21.7
Siamese Neural Net	58.3
Convolutional Siamese Net	92.0

To empirically evaluate one-shot learning performance, Lake developed a 20-way within-alphabet classification task in which an alphabet is first chosen from among those reserved for the evaluation set, along with twenty characters taken uniformly at random. Two of the twenty drawers are also selected from among the pool of evaluation drawers. These two drawers then produce a sample of the twenty characters. Each one of the characters produced by the first drawer are denoted as test images and individually compared against all twenty characters from the second drawer, with the goal of predicting the class corresponding to the test image from among all of the second drawer's characters. An individual example of a one-shot learning trial is depicted in Figure 7. This process is repeated twice for all alphabets, so that there are 40 one-shot learning trials for each of the ten evaluation alphabets. This constitutes a total of 400 one-shot learning trials, from which the classification accuracy is calculated.

The one-shot results are given in Table 2. We borrow the baseline results from (Lake et al., 2013) for comparison to our method. We also include results from a non-convolutional siamese network with two fully-connected layers.

At 92 percent our convolutional method is stronger than any model except HBPL itself. which is only slightly behind human error rates. While HBPL exhibits stronger results overall, our top-performing convolutional network did

[2]This can be processed efficiently by appending C copies of $\mathbf{x}$ into a single matrix $\mathbf{X}$ and stacking $\mathbf{x}_c^T$ in rows to form another matrix $\mathbf{X}_C$ so that we can perform just one feedforward pass with minibatch size C using input $\mathbf{X}, \mathbf{X}_C$.

Table 3. Results from MNIST 10-versus-1 one-shot classification task.

Method	Test
1-Nearest Neighbor	26.5
Convolutional Siamese Net	70.3

not include any extra prior knowledge about characters or strokes such as generative information about the drawing process. This is the primary advantage of our model.

4.4. MNIST One-shot Trial

The Omniglot data set contains a small handful of samples for every possible class of letter; for this reason, the original authors refer to it as a sort of "MNIST transpose", where the number of classes far exceeds the number of training instances (Lake et al., 2013). We thought it would be interesting to monitor how well a model trained on Omniglot can generalize to MNIST, where we treat the 10 digits in MNIST as an alphabet and then evaluate a 10-way one-shot classification task. We followed a similar procedure to Omniglot, generating 400 one-shot trials on the MNIST test set, but excluding any fine tuning on the training set. All 28x28 images were upsampled to 35x35, then given to a reduced version of our model trained on 35x35 images from Omniglot which were downsampled by a factor of 3. We also evaluated the nearest-neighbor baseline on this task.

Table 3 shows the results from this experiment. The nearest neighbor baseline provides similar performance to Omniglot, while the performance of the convolutional network drops by a more significant amount. However, we are still able to achieve reasonable generalization from the features learned on Ominglot without training at all on MNIST.

5. Conclusions

We have presented a strategy for performing one-shot classification by first learning deep convolutional siamese neural networks for verification. We outlined new results comparing the performance of our networks to an existing state-of-the-art classifier developed for the Omniglot data set. Our networks outperform all available baselines by a significant margin and come close to the best numbers achieved by the previous authors. We have argued that the strong performance of these networks on this task indicate not only that human-level accuracy is possible with our metric learning approach, but that this approach should extend to one-shot learning tasks in other domains, especially for image classification.

In this paper, we only considered training for the verification task by processing image pairs and their distortions using a global affine transform. We have been experimenting with an extended algorithm that exploits the data about the individual stroke trajectories to produce final computed distortions (Figure 8). By imposing local affine transformations on the strokes and overlaying them into a composite image, we are hopeful that we can learn features which are better adapted to the variations that are commonly seen in new examples.

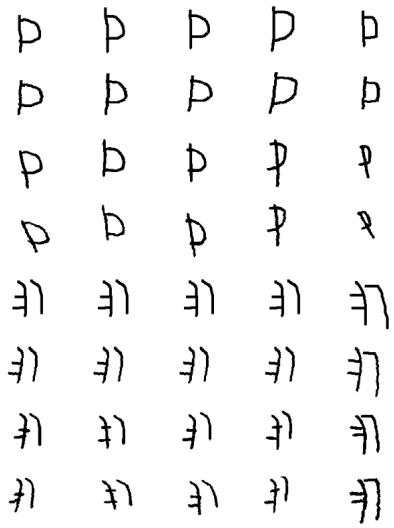

Figure 8. Two sets of stroke distortions for different characters from Omniglot. Columns depict characters sampled from different drawers. Row 1: original images. Row 2: global affine transforms. Row 3: affine transforms on strokes. Row 4: global affine transforms layered on top of stroke transforms. Notice how stroke distortions can add noise and affect the spatial relations between individual strokes.

References

Bengio, Yoshua. Learning deep architectures for ai. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

Bromley, Jane, Bentz, James W, Bottou, Léon, Guyon, Isabelle, LeCun, Yann, Moore, Cliff, Säckinger, Eduard, and Shah, Roopak. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7 (04):669–688, 1993.

Chopra, Sumit, Hadsell, Raia, and LeCun, Yann. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pp. 539–546. IEEE, 2005.

Fe-Fei, Li, Fergus, Robert, and Perona, Pietro. A bayesian approach to unsupervised one-shot learning of object categories. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 1134–1141. IEEE, 2003.

Fei-Fei, Li, Fergus, Robert, and Perona, Pietro. One-shot learning of object categories. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(4):594–611, 2006.

Hinton, Geoffrey, Osindero, Simon, and Teh, Yee-Whye. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Lake, Brenden M, Salakhutdinov, Ruslan, Gross, Jason, and Tenenbaum, Joshua B. One shot learning of simple visual concepts. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, volume 172, 2011.

Lake, Brenden M, Salakhutdinov, Ruslan, and Tenenbaum, Joshua B. Concept learning as motor program induction: A large-scale empirical study. In *Proceedings of the 34th Annual Conference of the Cognitive Science Society*, pp. 659–664, 2012.

Lake, Brenden M, Salakhutdinov, Ruslan R, and Tenenbaum, Josh. One-shot learning by inverting a compositional causal process. In *Advances in neural information processing systems*, pp. 2526–2534, 2013.

Lake, Brenden M, Lee, Chia-ying, Glass, James R, and Tenenbaum, Joshua B. One-shot learning of generative speech concepts. Cognitive Science Society, 2014.

Lim, Joseph Jaewhan. Transfer learning by borrowing examples for multiclass object detection. Master's thesis, Massachusetts Institute of Technology, 2012.

Maas, Andrew and Kemp, Charles. One-shot learning with bayesian networks. Cognitive Science Society, 2009.

Mnih, Volodymyr. Cudamat: a cuda-based matrix class for python. 2009.

Palatucci, Mark, Pomerleau, Dean, Hinton, Geoffrey E, and Mitchell, Tom M. Zero-shot learning with semantic output codes. In *Advances in neural information processing systems*, pp. 1410–1418, 2009.

Simonyan, Karen and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Srivastava, Nitish. Improving neural networks with dropout. Master's thesis, University of Toronto, 2013.

Taigman, Yaniv, Yang, Ming, Ranzato, Marc'Aurelio, and Wolf, Lior. Deepface: Closing the gap to human-level performance in face verification. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 1701–1708. IEEE, 2014.

Wu, Di, Zhu, Fan, and Shao, Ling. One shot learning gesture recognition from rgbd images. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pp. 7–12. IEEE, 2012.