Unsupervised learning of clutter-resistant visual representations from natural videos

Qianli Liao, Joel Z Leibo,Tomaso Poggio

Center for Brains, Minds & Machines

Abstract

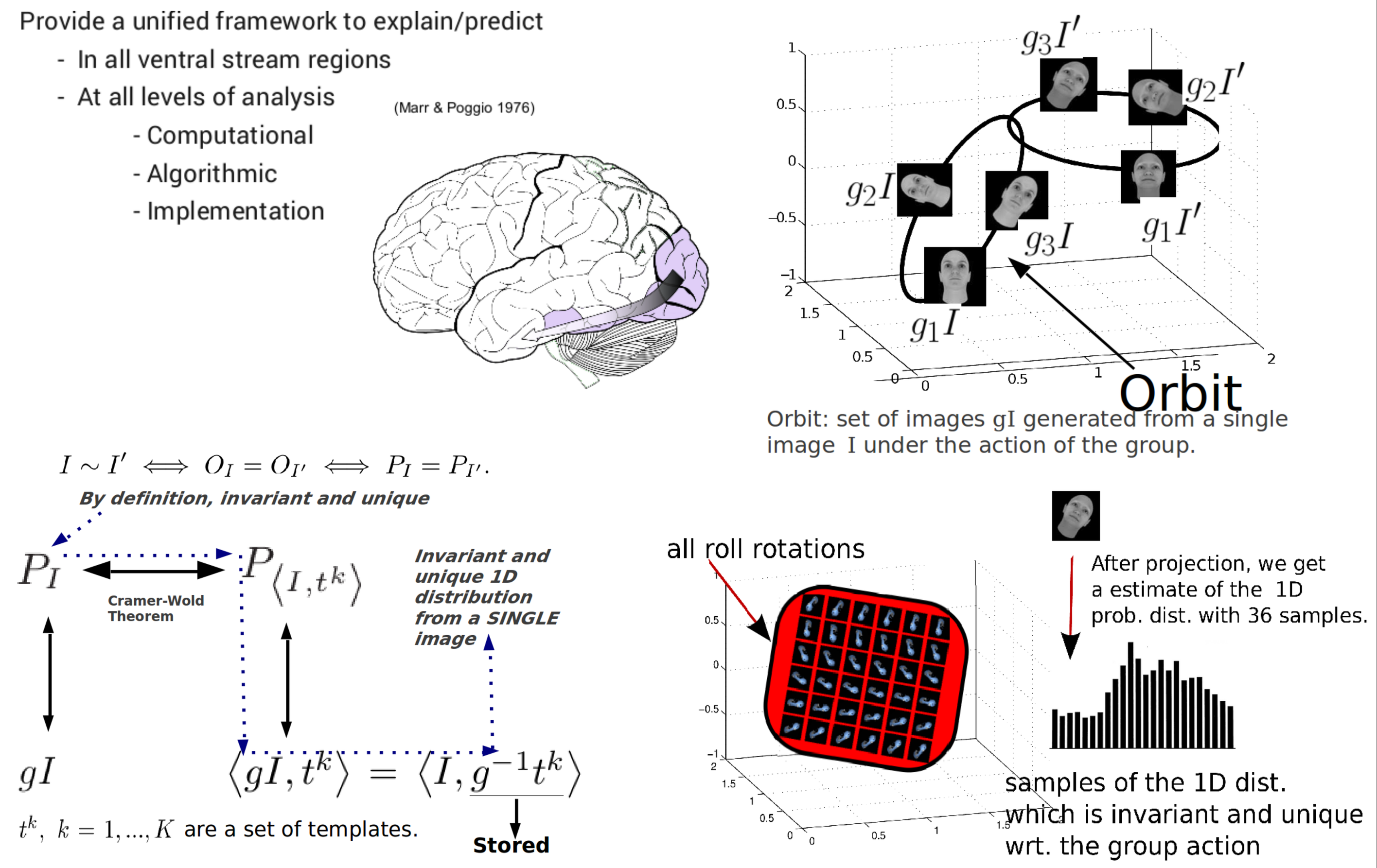
Populations of neurons in inferotemporal cortex (IT) maintain an explicit code for object identity that also tolerates transformations of object appearance e.g., position, scale, viewing angle [1, 2, 3]. Though the learning rules are not known, recent results [4, 5, 6] suggest the operation of an unsupervised temporal-association-based method e.g., Foldiak's trace rule [7]. Such methods exploit the temporal continuity of the visual world by assuming that visual experience over short timescales will tend to have invariant identity content. Thus, by associating representations of frames from nearby times, a representation that tolerates whatever transformations occurred in the video may be achieved. Many previous studies verified that such rules can work in simple situations without background clutter, but the presence of visual clutter has remained problematic for this approach. Here we show that temporal association based on large class-specific filters (templates) avoids the problem of clutter. Our system learns in an unsupervised way from natural videos gathered from the internet, and is able to perform a difficult unconstrained face recognition task on natural images (Labeled Faces in the Wild [8]).

Introduction

In natural videos and naturalistic vision the essential properties of images tend to be those that remain stable for some time. Much of what is incidental about images tends to fluctuate more rapidly. Previous efforts to exploit this principle of temporal continuity as a prior assumption enabling unsupervised learning of useful visual representations have only succeeded in demonstrating its effectiveness in rather contrived cases using synthetic objects on uniform backgrounds e.g., [7, 9, 10, 11, 12, 13]. Here we demonstrate a system that is, to the best of our knowledge, the first that can exploit temporal continuity to learn from cluttered natural videos a visual representation that performs competitively on challenging computer vision benchmarks.
Quite unlike the current "big data" trend, which has used systems trained with millions of labeled examples to produce significant advances in object recognition [14, 15], our proposal is aimed at understanding one of the strategies used by human visual cortex to learn to see from far less labeled data. To that end, we study how a biologically-plausible feedforward hierarchy can learn useful representations from unlabeled natural videos gathered from the internet. The model we propose is very simple: the only operations it performs are normalized dot products and pooling. It is nonparametric; after an initial layer of Gabor filtering, all the other filters it uses are sampled directly from the unlabeled training data. The classifier we use for the same-different task studied in all but the last section of this article is just a normalized dot product.
Despite its simplicity, the model performs well on unconstrained face recognition in natural images as well as a basic level task: categorizing lions and tigers. Moreover, in computer vision, solving this face recognition task is generally thought to require a complete detection - alignment - recognition pipeline. Yet the simple model developed here is able to operate directly on the totally unconstrained data: the original unaligned labeled faces in the wild (LFW) dataset [8].

Template-based invariant encodings for objects unseen during training

M-theory of the ventral stream



$$\mu_n^k(x) = \frac{1}{T} \sum_{t=1}^{T} \sigma(\langle x, g_t \tau_k \rangle + n\Delta)$$

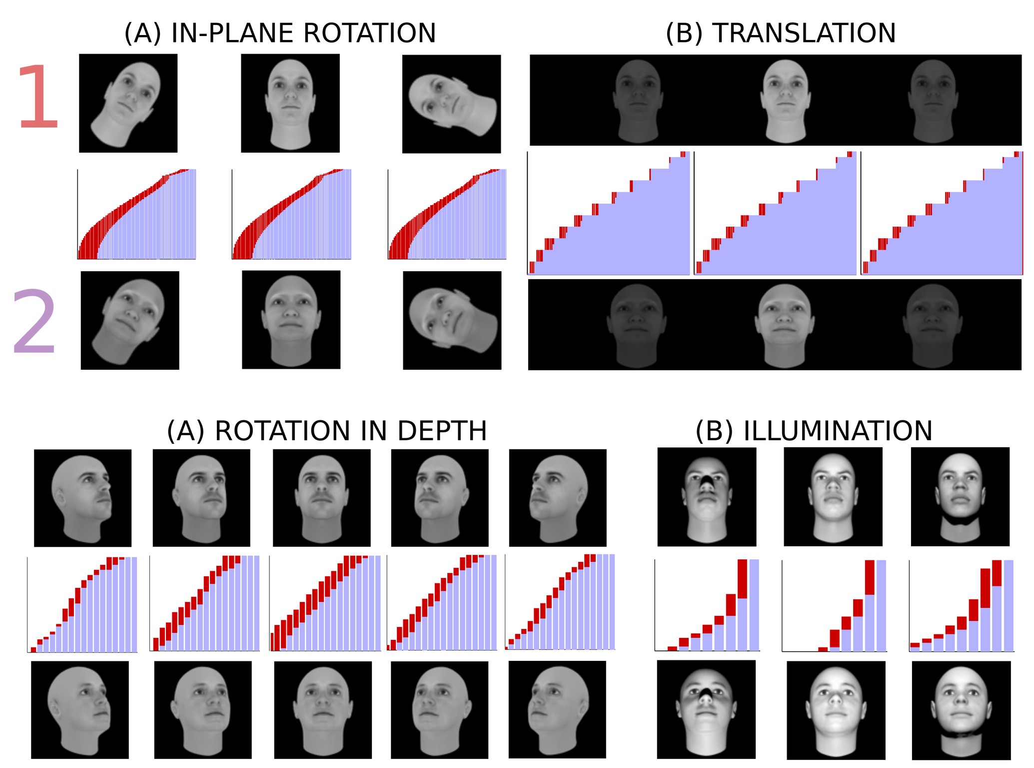


Figure 1: Example signatures (empirical distribution functions—CDFs) of images depicting two different faces under affine and non-affine transformations.

Clutter resistance

As a prelude to the results on natural videos, we first demonstrate the feasibility of the method, and its vulnerability to disruption by clutter using synthetic face data. Figure 1 summarizes the results from two of the tests from the Subtasks of Unconstrained Face Recognition (SUFR) benchmark [18]. The task is called face-verification (also known as pair-matching) — given two images of new faces, never encountered during training, the task is to decide if they depict the same person or not. The dataset consists of 400 faces, with 10,000 images rendered at different orientations (in depth) and different positions in the visual field for each. We perform 10-fold cross validation on the dataset to get the performance. Each fold consists of 360 training and 40 testing faces.
This section used a one-layer model with 360 HW-modules, each template book contained all the images of an individual face in the training set. This is analogous to the model that would be obtained from temporal association if the training data had been frames of a video. The classifier was a thresholded normalized dot product (See Section 4.6). The optimal threshold was determined by maximizing the training accuracy. With this classifier, the raw pixel representation performed at chance. Whereas the pooled representation's performance on the testing set was above 70% correct. However, when clutter is present, the performance of the temporal association approach drops significantly as shown in Figure 1, indicating that a naive implementation of this approach is prone to disruption by clutter. The following sections introduce a hierarchical extension to overcome this problem.

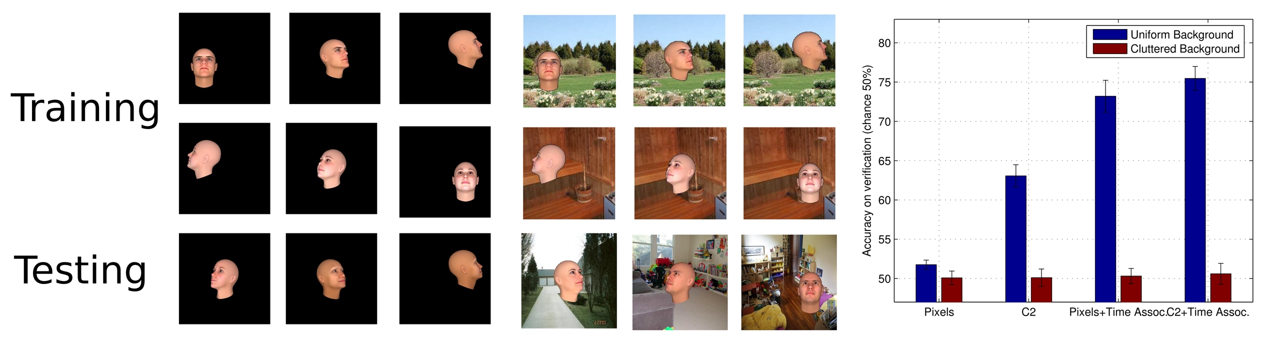


Figure 1: Temporal association with and without clutter: The upper two rows are examples of the training data. The last row shows the examples of the testing data. The leftmost three columns are example pictures with uniform background. The middle three columns are example pictures with cluttered background. The model is almost the same as that of [17], but without doing PCA. The temporal association is modeled by pooling over all the training frames of each individual (e.g., three successive frames on the first row). For the temporal association experiment with clutter, the images of each individual in the training template book have the same background. In the test set, each image has a different background. The C2 features is another type of low-level features, obtained from the second layer of HMAX. The observation is reliable across different low-level features.

Architectures and simulation methods

To mitigate the clutter problem described above, we propose a hierarchical model depicted by Figure 2. It exploits hierarchical processings to factorize transformations and build clutter-invariant representations for high-level temporal associations. The architecture consists of a hierarchy of HW-modules, it can be thought of as a succession of simple and complex cells performing two main operations tuning (projection on a template) and pooling. The final output is the signature μ(I), a vector of top-level HW-module responses, invariant to affine (group) transformations and approximately invariant to class-specific transformations in a short timeframe.

Previous paper by [19] revealed that large class-specific image patches can be exploited to build clutter-invariance. We follow the same procedures to prepare the large image patches for our two models. Examples are shown in the second layers of Figure 2. These patches can be easily generated by sampling large image patches on a face/cats&dogs dataset. Their low-level features are used as the second layer templates.
The model is trained in a layer-wise fashion. For training, (1) compute layer 1 features of all video frames and large image patches mentioned above. The latter become layer 2 templates. (2) compute layer 2 features of all video frames. They become layer 3 templates.
To better explain the algorithm, we will give exemplary dimensions of the features of each layer in the form of yi × xi × zi , where y denotes the height, x denotes the width, z denotes the thickness/feature size, and i is the layer number. Our model is architecturally akin to HMAX and Convolutional Neural Networks. All of these architectures can be described using this yi × xi × zi notation.
Layer 1 features are computed by applying a function L() to the input image ("L" means low-level features).
The function L() transforms the input image (y0 × x0 × 3, i.e., colored image) to (y1 × x1 × z1), where y1 and x1 are usually smaller than y0 and x0 , respectively, while z1 is almost always larger than 3. This is a unified view of a large number of low-level features, including flat features like HOG, LBP, (Gabor) filtering and hierarchical features like HMAX and convolutional networks in general. This abstraction allows us to apply our temporal association methods to any low-level features.
We sample the videos of length T seconds at a rate of f frames/second to get T ∗ f frames. We use Fi to denote a single frame, where i = 1...T ∗ f .

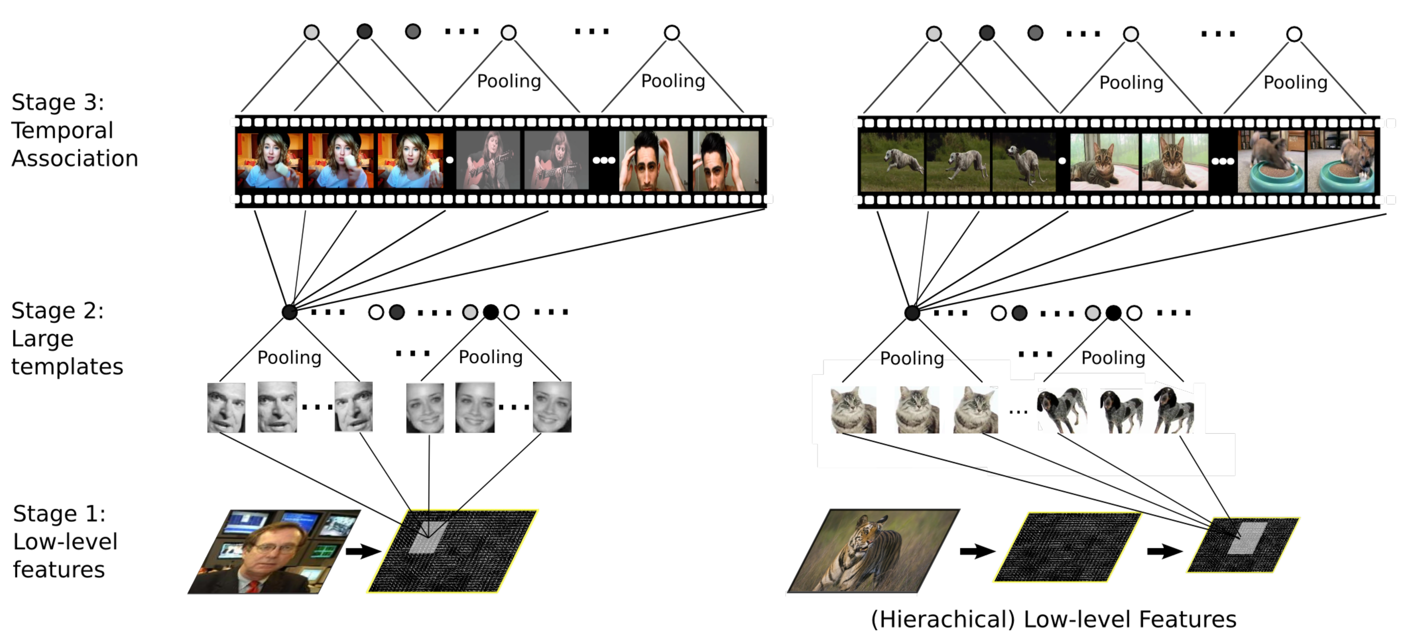


Figure 2: Illustrations of the two models used in this paper. The left one is the face model. It uses closely-cropped faces as the second layer templates. The model on the right hand side is the model for recognizing dogs and cats. It uses closely-cropped cat and dog patches as the second layer templates. The low-level features of the face model is single-layered but that of the latter model is two-layered — a hierarchical HMAX-like architecture. Emperically, one layer works better for faces since faces are more subtle and requires higher resolution.

Face Model

Training

We randomly sample large face patches P from the SUFR-W [18] dataset. For each patch, we generate its in-plane rotations and a horizontal flip. We resize (with aspect ratio preserved) each video frame to a image of height 400 pixels, and build a pyramid F of 20 scales with ratios from 0.26 to 1.0. We tried three types of low-level features (denoted by L()) for the face model: (1) one layer Gabor filtering + 2x2 spatial max pooling with stepsize 2 (2) HOG features (3) one layer of PCA + 2x2 spatial max pooling with stepsize 2.
First Layer: We apply the low-level feature function L() to patches P and multi-scale image pyramid F to get L(P) and L(F), respectively. L(P) become the second layer templates (i.e., "simple cells").
Second Layer: "Simple Cell" Convolution: each scale of the pyramid L(F) is convolved with the second layer templates L(P) separately. This is similar to a convolutional layer in CNN with three differences: (1) we have multiple scales (2) our templates/filters are spatially very large. (3) we adopt normalized dot product while CNN uses dot product."Complex cell" pooling: the result of the above convolution is still a pyramid of 20 scales. For each template, we max-pool over all the scales, spaces and in-plane transformations. The output of the second layer is the concatenation of these pooled results.
Third Layer: For each video frame, run the architecture until the second layer. Store the outputs of the second layer as the third-layer templates (i.e., "simple cells").

Testing

First Layer: For a test image, we compute its low-level features as described above.
Second Layer: The output of the first layer is convolved with L(P). Then we pool over scales, spaces and in-plane transformations as described above.
Third Layer: For each output from the second layer, compute the normalized dot product with the stored third layer training templates (i.e., "simple cells"). Then we pool the responses over temporally adjacent frames within N seconds, where N is a hyper parameter. Let the sampling rate be f frames/second, then there are N * f "simple cells" connected to a "complex cell". Normalized dot product is performed between the input and the "simple cells". The "complex cell" pools over these responses. The final output of the third layer is the concatenation of pooled responses.

Dog/cat Model

The Model for dog/cat recognition is similar to that for faces with the following differences: (1) we use two-layer HMAX-like features as low-level features. (2) we use cat/dog patches as second layer templates instead of faces. (3) other miscellaneous parameter differences (e.g., scales, template numbers, etc. Please refer to the paper for details).

PCA approximation

Note that the matrix P , consisting of all the second layer templates (as column or row vectors), is a low rank matrix. We can perform PCA on P and keep the k largest eigenvectors. By projecting the templates and the windows to the k dimensional space defined by those eigenvectors we can perform faster dot products in the reduced dimensional space. This procedure is adopted by [19] and similar to the one employed by [20].

Training videos and data usage

In our experiments, the face model learns from 375 videos of faces from YouTube. The lion&tiger model learns from 224 videos of dogs and cats from YouTube. These videos may contain things other than faces, cats and dogs. See figure 5.
Simple cells: Each frame corresponds to a simple cell (in the third layer of our model). For speed purpose, videos are sampled at some slow rates: 0.5, 1 or 2, 4, etc. frames/second. So there are 0.5, 1 or 2, 4, etc. "simple cells" on average per second.
Complex cells: How are the "complex cells" placed over the time domain? Each "complex cell" has a pooling domain over time that may or may not overlap with other "complex cells". The placement of "complex cells" over the time domain is a hyper parameter that depends on the experiments. We have two ways of placing the "complex cells" in this paper:
1. For the control experiment in Figure 3, we truncate each video to 60 seconds and place the "complex cells" in a even and non-overlapping way. This is mainly for speed purpose, since none of the "simple cells" are wasted in this case. See the caption of Figure 3 for details.
2. For the final performance in Table 1, in order to avoid being biased by some very long videos, we simply specify that each video has exactly M complex cells. Their pooling domains are equally large over time, and they are spread out evenly in each video to maximize the data (i.e., "simple cell") usage. For some short videos, there may be some overlaps between the pooling domains of its "complex cells". For some long videos, some frames/"simple cells" are wasted

Evaluation Process

Here we briefly describe how the model is evaluated.
Face verification Given a pair of images (xa , xb) the task is to verify whether they depict the same person or not. To test our HW-architecture, we run it on both images and compare them by the angle between their signatures (top-level representations). That is, we take the normalized dot product μ(xa), μ(xb) if it exceeds a threshold τ , our method outputs that xa and xb have the same identity otherwise they are different. We use the training set to pick the optimal τ .
Lion and tiger classification Given an image x, the task is to determine whether it depicts lion or tiger. We trained an SVM on the final output of our feature extraction pipeline.

Subordinate level identification of faces

In this section, we train our face model with 375 natural videos gathered from YouTube and test on the "gold standard" dataset of face recognition — Labeled Faces in the Wild. In the field of face recognition, people tend not to work directly on the end-to-end task like this. Usually they consider the recognition problem on the outcome of a detection-alignment pipeline. This is one of the main reasons why people get high results on LFW. One previous paper [19] considered the original, unaligned LFW images with an approach like ours. However, they did not investigate temporal association, and their system was much more complicated (with local binary patterns, locality-sensitive hashing, etc.) and much less biological.
Example training and testing images are shown in Figure 5.

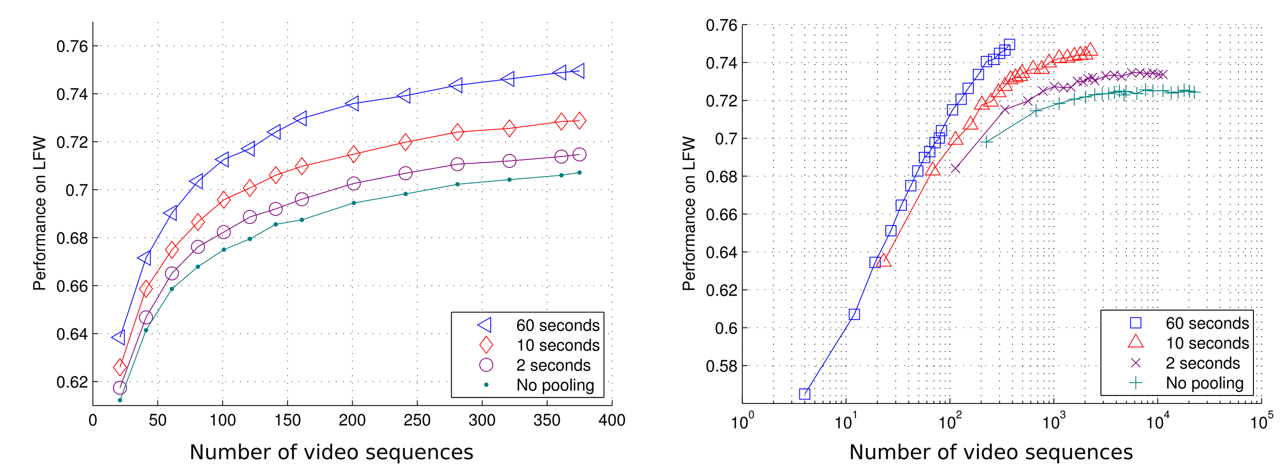


Figure 3: Performance vs. temporal pooling range. There are 375 face videos. In this particular experiment, each video is truncated such that they are all 60 seconds long. The videos are sampled at 1 frame per second. Thus, for each video we get 60 frames. There are 4 pooling options: 60 seconds, 10 seconds, 2 seconds, no pooling. The pooling ranges here are non-overlapping to maximize data usage (but generally overlapping in other experiments of the paper). We observe that the longer the pooling range, the higher the performance is. We show two different views of the result in this figure – one is short range and the other is long-range (and log scale). Note that truncation to 60 seconds is only performed in this control experiment but not in the final model, and it is only for speed purpose. The pattern is robust across different settings.

Basic level categorization: lions and tigers

The goal of temporal association learning is to produce an invariant representation for the image transformations that naturally arise as objects move relative to their observer. So far we have concentrated on a subordinate level task: face recognition. Such tasks which require very similar objects to be discriminated from one another are thought to depend more critically on invariance. Specifically, they are thought to require more complicated class-specific invariances that would be hard to learn in any way other than from natural videos, e.g., rotation in depth or the transformation of a frown to a smile. In contrast, basic level categorization may rely more on discriminative features that appear similarly from any angle than on sophisticated invariance to class-specific transformations. It follows that temporal association should be a more effective strategy in the subordinate level case.
To further explore this direction, we applied our model on a basic level categorization task — categorizing lions and tigers. The result (Figure 6) is preliminary but interesting. We developed our model with 224 natural videos of cats and dogs from YouTube. We also created a lion&tiger dataset consists of 1128 images. For training, only 16 labeled examples are used from the lion&tiger dataset. 400 testing images are used per category. For the second layer templates, we used large patches from dogs and cats (as shown in 2). We found that when pooling over scrambled frames, the performance drops to the level of "no pooling". The results are averaged over 120 trials and the effect is robust to different choices of parameters.

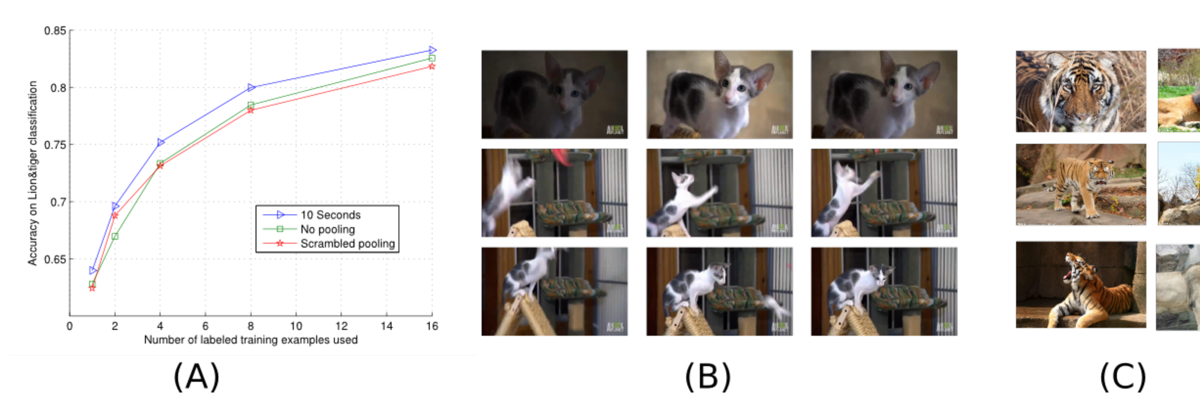


Figure 6: We developed our model with 224 natural videos of cats and dogs from YouTube. In this experiment, frames are sampled at 1 fps. (A) The performance of our model. We tested 10-second pooling, no pooling and 10-second pooling with scrambled frames. The performance is averaged over 120 trials. (B) example training frames from the YouTube videos we gathered. (C) examples of our testing lion & tiger dataset.

LFW Results					
Detected & Cropped				Undetected & Uncropped	
Unsupervised		Supervised & Unaligned		Unaligned	
Model	**Acc.**	**Model**	**Acc.**	**Model**	**Acc.**
LHS (aligned) [21]	73.4%	MERL [22]	70.52%	SIFT-BoW+SVM [19]	57.73±2.53%
MRF-MLBP (aligned) [23]	80.08%	Nowak et al. [24]	72.45%	Our Model(Gabor)	75.57±1.63%
I-LPQ (aligned) [25]	86.2%	Sanderson et al. [26]	72.95%	Our Model (fusion)	**81.32±1.40%**
PAF (aligned) [27]	87.77%	APEM (fusion) [28]	81.70%	Our Model (fusion)+SVM	**83.23±1.07%**

Table 1: Note that all top unsupervised methods on LFW require detected, cropped and aligned faces. The SVM results were obtained by simply replacing the cosine classifier with a SVM. In the final experiment (fusion) in Table 1, we trained three pipelines based on Gabor, PCA and HOG features. The signatures of the third layer of each pipeline were computed separately. They were then fused by a weighted concatenation and fed into the classifier. The concatenation weights were determined by minimizing the training error

Model	LFW	LFW-J
HOG+SVM [19]	74.45±1.79/67.32±1.59%	55.28±2.02%
Our Model (Gabor)	75.57±1.63%	75.48±1.60%
Our Model (Fusion)	81.32±1.40%	81.10±1.15%

Table 2: We report the performance of our model on the LFW-J (jittered) dataset created by [19]. The LFW-J dataset is created by randomly translating, scaling and rotating the original LFW images. The HOG baseline is given by [19]. 74.45% is the closely cropped performance and 67.32% is the non-cropped performance. For LFW-J, one could hardly closely crop a face without detection.

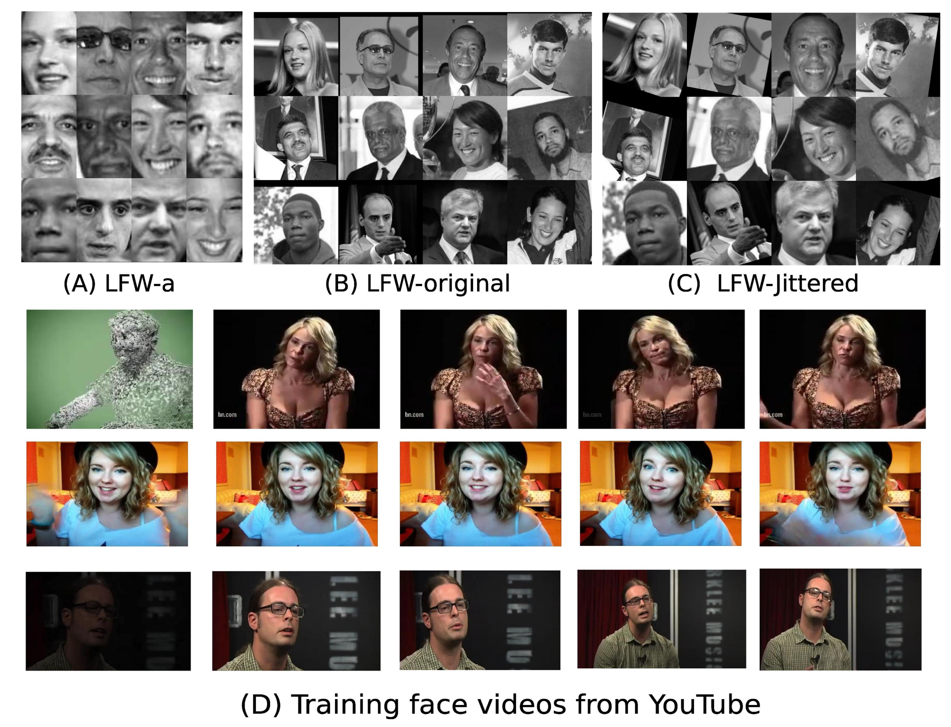


Figure 5: Example testing and training images: (A) LFW-a (LFW-aligned) is the usual dataset used by LFW practitioners. Here we address (B) and (C), which are much more difficult. (D) are some example frames from our training videos.

Acknowledgments

This work was supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF - 1231216.

References

References
[1] C. P. Hung, G. Kreiman, T. Poggio, and J. J. DiCarlo, "Fast Readout of Object Identity from Macaque Inferior Temporal Cortex," Science, vol. 310, pp. 863–866, Nov. 2005.
[2] Y. Yamane, E. T. Carlson, K. C. Bowman, Z. Wang, and C. E. Connor, "A neural code for three-dimensional object shape in macaque inferotemporal cortex," Nature neuroscience, vol. 11, no. 11, pp. 1352–1360, 2008.
[3] W. A. Freiwald and D. Tsao, "Functional Compartmentalization and Viewpoint Generalization Within the Macaque Face-Processing System," Science, vol. 330, no. 6005, p. 845, 2010.
[4] N. Li and J. J. DiCarlo, "Unsupervised natural experience rapidly alters invariant object representation in visual cortex.," Science, vol. 321, pp. 1502–7, Sept. 2008.
[5] N. Li and J. J. DiCarlo, "Unsupervised Natural Visual Experience Rapidly Reshapes Size-Invariant Object Representation in Inferior Temporal Cortex," Neuron, vol. 67, no. 6, pp. 1062–1075, 2010.
[6] N. Li and J. J. DiCarlo, "Neuronal learning of invariant object representation in the ventral visual stream is not dependent on reward," The Journal of Neuroscience, vol. 32, no. 19, pp. 6611–6620, 2012.
[7] P. F¨ldi´k, "Learning invariance from transformation sequences," Neural Computation, vol. 3, no. 2, pp. 194–
o a
200, 1991.
[8] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," in Workshop on faces in real-life images: Detection, alignment and recognition (ECCV), (Marseille, Fr), 2008.
[9] L. Wiskott and T. Sejnowski, "Slow feature analysis: Unsupervised learning of invariances," Neural computation, vol. 14, no. 4, pp. 715–770, 2002.
[10] M. Spratling, "Learning viewpoint invariant perceptual representations from cluttered images," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 5, pp. 753–761, 2005.
[11] M. Franzius, N. Wilbert, and L. Wiskott, "Invariant object recognition and pose estimation with slow feature analysis," Neural Computation, vol. 23, no. 9, pp. 2289–2323, 2011.
[12] L. Isik, J. Z. Leibo, and T. Poggio, "Learning and disrupting invariance in visual recognition with a temporal association rule," Front. Comput. Neurosci., vol. 6, no. 37, 2012.
[13] E. Rolls, "Invariant visual object and face recognition: neural and computational bases, and a model, VisNet," Frontiers in Computational Neuroscience, vol. 6, 2012.
[14] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in Advances in neural information processing systems, vol. 25, (Lake Tahoe, CA), pp. 1106–1114, 2012.
[15] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks," arXiv preprint arXiv:1312.6229, 2013.
[16] F. Anselmi, J. Z. Leibo, J. Mutch, L. Rosasco, A. Tacchetti, and T. Poggio, "Unsupervised Learning of Invariant Representations in Hierarchical Architectures," arXiv:1311.4158v3 [cs.CV], 2013.
[17] Q. Liao, J. Z. Leibo, and T. Poggio, "Learning invariant representations and applications to face verification," in Advances in Neural Information Processing Systems (NIPS), (Lake Tahoe, CA), 2013.
[18] J. Z. Leibo, Q. Liao, and T. Poggio, "Subtasks of Unconstrained Face Recognition," in International Joint Conference on Computer Vision, Imaging and Computer Graphics, VISAPP, (Lisbon, Portugal), SCITEPRESS, 2014.
[19] Q. Liao, J. Z. Leibo, Y. Mroueh, and T. Poggio, "Can a biologically-plausible hierarchy effectively replace face detection, alignment, and recognition pipelines?," arXiv preprint arXiv:1311.4082, 2014.
[20] S. Chikkerur and T. Poggio, "Approximations in the hmax model," 2011.
[21] G. Sharma, S. ul Hussain, and F. Jurie, "Local higher-order statistics (lhs) for texture categorization and facial analysis," in European Conference on Computer Vision (ECCV), 2012.
[22] G. B. Huang, M. J. Jones, E. Learned-Miller, et al., "Lfw results using a combined nowak plus merl recognizer," in Workshop on Faces in'Real-Life'Images: Detection, Alignment, and Recognition, 2008.
[23] S. R. Arashloo and J. Kittler, "Efficient processing of mrfs for unconstrained-pose face recognition," in Biometrics: Theory, Applications and Systems, 2013.
[24] E. Nowak and F. Jurie, "Learning visual similarity measures for comparing never seen objects," in Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, pp. 1–8, IEEE, 2007.
[25] S. Hussain, T. Napol´on, and F. Jurie, "Face recognition using local quantized patterns," in Proc. British
e
Machine Vision Conference (BMCV), vol. 1, (Guildford, UK), pp. 52–61, 2012.
[26] C. Sanderson and B. C. Lovell, "Multi-region probabilistic histograms for robust and scalable identity inference," in Advances in Biometrics, pp. 199–208, Springer, 2009.
[27] D. Yi, Z. Lei, and S. Z. Li, "Towards pose robust face recognition," in Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pp. 3539–3545, IEEE, 2013.
[28] Z. Cui, W. Li, D. Xu, S. Shan, and X. Chen, "Fusing robust face region descriptors via multiple metric learning for face recognition in the wild," in Computer Vision and Pattern Recognition (CVPR), 2013.