Random Convolutional Features and Patch-Based Learning for Multitask Image Classification

Zan Ahmad, James Schmidt
Department of Applied Mathematics and Statistics,
Johns Hopkins University, Baltimore MD, USA
zahmad6@jhu.edu, aschmi40@jhu.edu

Introduction

We explore the effectiveness of random projections in the context of convolutional neural networks (CNNs). Several works have shown that randomness can speed computation while preserving competitive accuracy to state of the art ML models in applied contexts. Here, we consider CNNs with kernels sampled from a distribution on the training data. A random projection into a lower dimensional space is obtained from one forward pass of a CNN. Rather than optimizing the kernels, we optimize only the weights of the random convolutional features obtained. We show that training a shallow architecture by randomly fixing the nonlinearities in the first layer results in a classifier that is comparable to one constructed by optimizing said nonlinearities in an analogous architecture. Our results follow from theory developed by Rahimi and Recht [1].

Theoretical Observations

Consider functions of the form $f(x) = \int_\Omega \alpha(\omega)\phi(x;\omega)d\omega$ with $x \in \mathcal{X}$, ϕ a nonlinear activation function, α scalar weights, and define the norm on a probability distribution p on the parameter space Ω as follows: $\|f\|_p = \sup_{\omega\in\Omega} \frac{|\alpha(\omega)|}{p(\omega)}$ and let

$$F_p \equiv \{f(x) = \int_\Omega \alpha(\omega)\phi(x;\omega)d\omega : \|f\|_p < \infty\}$$

Theorem 1. *Let μ be any measure on $\mathcal{X}$ and fix $f^* \in F_p$. Draw $\omega_1, \dots, \omega_K$ i.i.d from $p(\omega)$. Then with probability at least $1-\delta$, there exists $\alpha_1, \dots, \alpha_K$ s.t. $\hat{f}(x) = \sum_{k=1}^K \alpha_k \phi[x : \omega_k]$ satisfies*

$$\|\hat{f}_K - f^*\|_\mu \leq \mathcal{O}\left(\frac{\|f\|_p}{\sqrt{K}}\sqrt{log(\frac{1}{\delta})}\right)$$

where $\|f\|_\mu = \int_\mathcal{X} f(x)\mu(dx)$

Remark 1: F_p is a very rich space of functions.

- F_p is dense in a Reproducing Kernel Hilbert Space $\mathcal{H}$ which is dense in the space of continuous functions.
- Rate of convergence depends on both K (number of random bases) and probability distribution p on Ω.

Now consider the problem of fitting a function $f : \mathcal{X} \to \mathbb{R}$ to a training dataset S of size N: $\{x_i, y_i\}_{i=1}^N$ sampled i.i.d. from an unknown probability distribution $\mathbb{P}_\mathcal{X}$. We want to find f that minimizes the empirical risk with respect to some cost function c:

$$\mathbf{R}_{\text{emp}}[f] \equiv \frac{1}{N}\sum_{i=1}^N c(f(x_i), y_i)$$

When f is of the weighted sum form above, and rather than minimizing over $\omega_1, \dots, \omega_K \in \Omega$ and $\alpha_1, \dots, \alpha_K \in \mathbb{R}$ we can sample $\{\omega_i\}_{i=1}^K$ i.i.d. from p to obtain the following minimization problem:

$$\hat{f} = \min_{\alpha\in\mathbb{R}^K} \mathbf{R}_{\text{emp}}[\sum_{k=1}^K \phi(x:\omega_k)\alpha_k]$$

Theorem 2. *With probability $1-2\delta$, we have the following difference in true risk*

$$\mathbf{R}[\hat{f}] - \min_{f\in F_p} \mathbf{R}[f] \leq \mathcal{O}\left(\left(\frac{1}{\sqrt{N}} + \frac{1}{\sqrt{K}}\right)log(\frac{1}{\delta})\right)$$

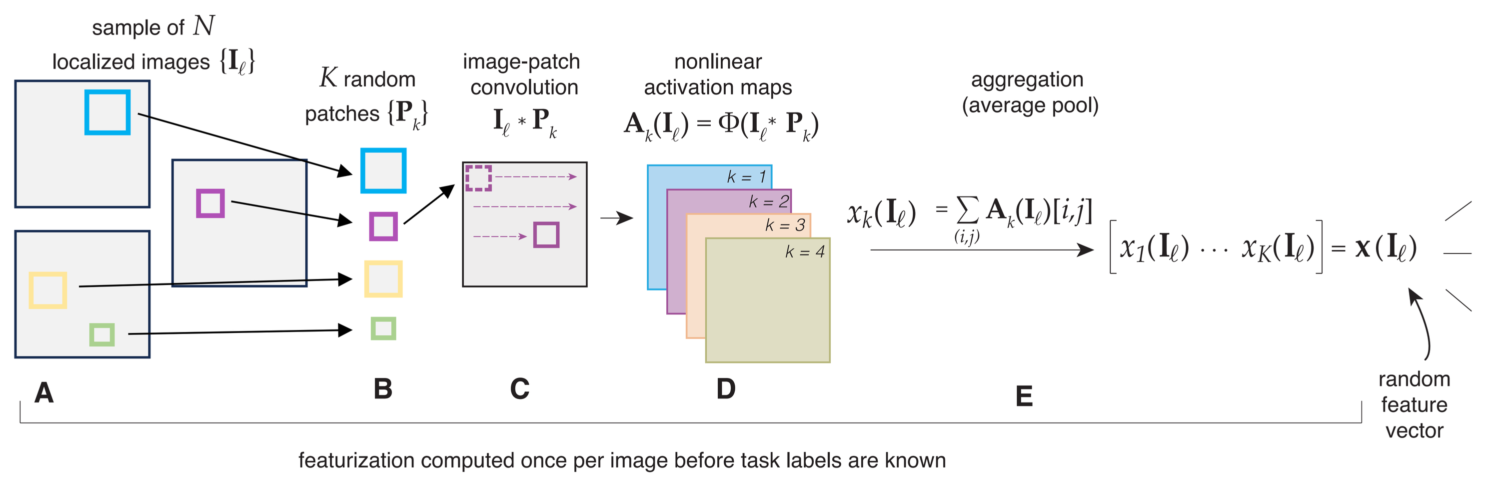
Remark 2: Distance between model and optimal map decays as K gets larger

- Number of random features, K, is directly proportional to how well model approximates the risk minimizer.
- The decay rate is of the same order as when optimizing both ω's and α's: $\mathcal{O}(C/K)$ for some constant C.

Methods

Plan: Adapt theory to the context of image classification with CNNs.

- A convolutional neural network (CNN) is a type of neural network often used for image classification consisting of one or more convolutional layers.
- In these layers, each image is convolved with several filters which are optimized via backpropagation and gradient descent.
- Instead of optimizing the filters, we sample subimages from the data and fix them as our kernels. We modify the framework in [2] by randomizing patch sizes.



Connecting the experimental framework to the theory, we have

- **A** Our input x being a set of N images $\{I_\ell\}_{\ell=1}^N$ of $M_\ell \times M_\ell$ pixels drawn i.i.d. from some distribution $\mathbb{P}_\mathcal{X}$.
- **B** Given a large sample of N images, we randomly draw K patches of varying random size $m_k \times m_k$ (such that $m_k \leq M, \forall k \in [K]$) from a distribution p on all subimages of $\{I_\ell\}_{\ell=1}^N$ (our restricted parameter space Ω), to obtain a patch dictionary $\{P_k\}_{k=1}^K$ which serves as our $\omega_1, \dots \omega_K$.
- **C** These K random patches P_k are then convolved with each image in the dataset $\{I_\ell\}_{\ell=1}^N$.
- **D** The outputs of these convolutions are then passed through a nonlinear activation function $\phi(I_\ell * P_k) = ReLU(I_\ell * P_k) = \max(I_\ell * P_k, 0)$
- **E** The random featurization is obtained by aggregating over all entries of the activation maps generated in **D**:

$$\mathbf{x}_k(I_\ell) = \frac{1}{M'}\sum_{i=1}^{M'}\sum_{j=1}^{M'} \phi(I_\ell * P_k)[i,j]$$

Note on Dimensionality and Generality

- Because a convolution operation is an inner product, the map $\mathbf{x}(I_\ell)$ can be interpreted as a random projection of an image from an $M \times M$ dimensional space to a K-dimensional space.
- The random feature vectors are generated in an unsupervised manner and thus can be used as input into a simple linear classification model with labels appended, where we can learn the scalar weights α for various tasks.

Random Convolutional Features

Algorithm 1: Ablated CNN for multitask classification

Input: Image data $S = \{x_1 \dots x_N\}$, probability distribution p on Ω (subimages of S), integer K, feature function ϕ, probability distribution μ on viable patch sizes, integer D, set of labels for D tasks $\{y_{i,d}\}, i = 1, \dots, N, d = 1, \dots, D$.

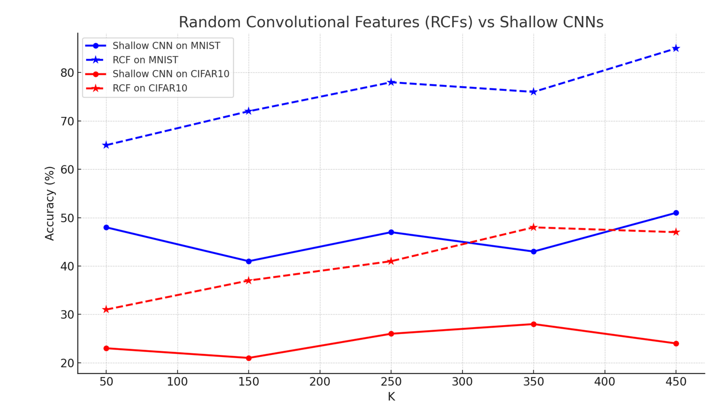
Output: D classification models $\hat{f}_d(x) = \sum_{k=1}^K \phi(x;\omega)\alpha(\omega)$

Draw patches $\omega_1, \dots, \omega_K$ i.i.d. from p on Ω with sizes depending on discrete measure μ.
Featurize data: $\phi(x_1;\omega), \dots, \phi(x_N;\omega)$ to obtain $N \times K$ feature matrix.
Append labels y_d and learn α_d to yield output with low loss.

Experimental Results

Experiments on MNIST and CIFAR10 Datasets

Varying Number of Random Patches or Filters K:

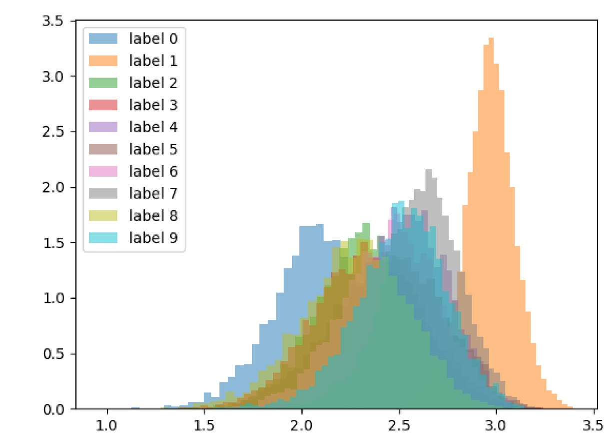


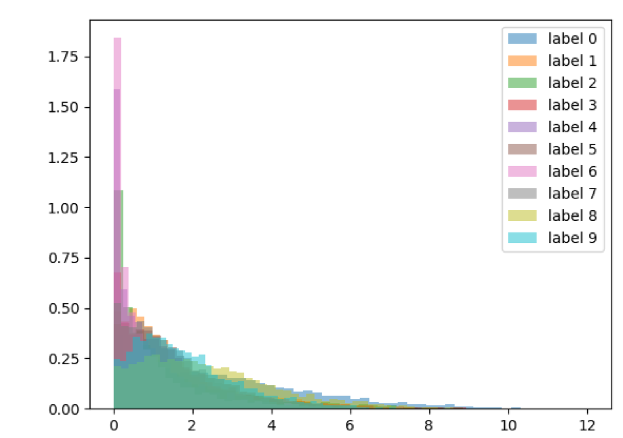
- Compared random patch-based method with a shallow CNN of analogous architecture (i.e., one convolutional layer with K filters and one fully connected layer)
- Random features outperform shallow CNNs. Random feature generation and inference is **10-50 times faster** than optimization/training of shallow CNN.

Multitask Observations on MNIST

- Since random featurization is a task-agnostic/label-independent/unsupervised method, the outputs should be able to maintain predictive value for several different tasks.
- RCFs have an advantage over CNNs in that the patches do not get optimized for a specific task, saving time from having to retrain the model.

Task	CNN Acc	RCF Acc	CNN Train Time (s)	RCF Gen Time (K=400)	RCF Train Time (s)
Standard Digits	51.2%	79.1%	583	66.5s	65.8s
Parity	68.4%	89.7%	640	66.5s	42.4s
Primality	60.4%	92.3%	787	66.5s	66.9s
mod3	81.2%	96.1%	827	66.5s	63.3s
mod4	77.4%	97.2%	830	66.5s	52.9s
loop detection	66.9%	88.36%	965	66.5s	74.2s
0-4 or 5-9	79.1%	78.1%	654	66.5s	60.2s





Histograms of one dimensional random projections of MNIST image data with a random patch (left) and a filter from the fifth training epoch of a shallow CNN (right).

References

[1] A. Rahimi and B. Recht, "Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning," *Advances in neural information processing systems*, vol. 21, 2008.

[2] E. Rolf, J. Proctor, T. Carleton, I. Bolliger, V. Shankar, M. Ishihara, B. Recht, and S. Hsiang, "A generalizable and accessible approach to machine learning with global satellite imagery," *Nature communications*, vol. 12, no. 1, p. 4392, 2021.