Convolutional Kernel Networks

Julien Mairal, Piotr Koniusz, Zaid Harchaoui, and Cordelia Schmid

Arthur Gretton's notes

August 6, 2015

What the paper is about

Main ideas: explicitly design feature spaces for ML on images (retrieval, classification) which

1. build more complex (hierarchical) feature spaces from simpler ones
2. achieve similar *local invariance*, and *fine-to-coarse parts-based features*, as convolutional neural nets

The resulting architectures are kernel based (!) and obtain similar performance to CNNs, but much more shallow (2 layers) and with many fewer parameters.

First: a basic kernel for images

A basic kernel between two images:

- The image is on coordinate space $\Omega_0 = [-1, 1]^2$
- A coordinate on the image is $\mathbf{z} \in \Omega_0$.
- The feature at $\mathbf{z}$ is $\varphi_0(\mathbf{z}) \in \mathcal{H}_0$. E.g. vector of RGB values, so $\varphi(\mathbf{z}_k) \in \mathbb{R}^3$.

Then a kernel between two images (as described entirely by features φ, φ') is average over kernels at each coordinate pair

$$\mathfrak{K}(\varphi, \varphi') = \sum_{\mathbf{z} \in \Omega} \sum_{\mathbf{z}' \in \Omega} \|\varphi_0(\mathbf{z})\|_{\mathcal{H}_0} \left\|\varphi_0'(\mathbf{z}')\right\|_{\mathcal{H}_0} e^{-\frac{1}{\beta^2}\|\mathbf{z}-\mathbf{z}'\|^2} e^{-\frac{1}{\sigma^2}\left\|\tilde{\varphi}_0(\mathbf{z})-\tilde{\varphi}_0'(\mathbf{z}')\right\|^2},$$

where

$$\tilde{\varphi}_0(\mathbf{z}) = \frac{\varphi_0(\mathbf{z})}{\|\varphi_0(\mathbf{z})\|_{\mathcal{H}_0}}$$

- β is the "local shift invariance" parameter.

First: a basic kernel for images

Some useful features:

- $\mathcal{H}_0 = \mathbb{R}^2$ is the 2-D gradient of the image at pixel $\mathbf{z}$.Then $\tilde{\varphi}_0(\mathbf{z})$ is orientation and $\|\varphi_0(\mathbf{z})\|_{\mathcal{H}_0}$ is intensity.
- $\mathcal{H}_0 = \mathbb{R}^{m\times m}$ are the pixel intensities centred at patch $\mathbf{z}$. $\tilde{\varphi}_0(\mathbf{z})$ is contrast-normalized map.

Next: kernel defined on a Hilbert space input

- The input feature space is defined as $\varphi_{k-1} : \Omega_{k-1} \to \mathcal{H}_{k-1}$.
  - i.e. the input feature at coordinate $\mathbf{z}_k \in \Omega_{k-1}$ is $\varphi(\mathbf{z}_k)$.
  - From prev. slide: in the case of Ω_0 these are e.g. raw RGB values.
- Define the local patches

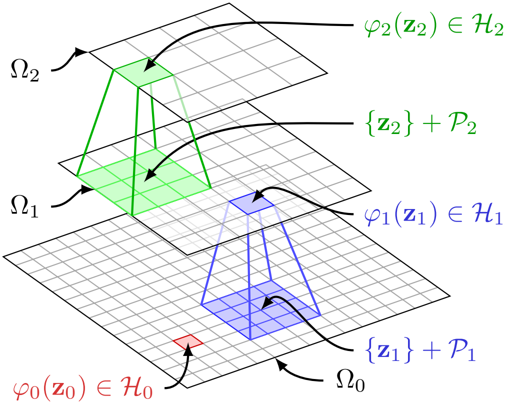
$$\{\mathbf{z}_k\} + \mathcal{P}_{k-1} \subset \Omega_{k-1}$$

Define the feature space $\mathcal{H}_k$ at level k, between two patches at level $k-1$,

$$\begin{aligned}\mathfrak{K}_k(\mathbf{z}_k, \mathbf{z}'_k) &= \sum_{\mathbf{z}\in\mathcal{P}_k} \sum_{\mathbf{z}'\in\mathcal{P}_k} \|\varphi_{k-1}(\mathbf{z}_k + \mathbf{z})\|_{\mathcal{H}_{k-1}} \left\|\varphi'_{k-1}(\mathbf{z}'_k + \mathbf{z}')\right\|_{\mathcal{H}_{k-1}} \\ &\quad \times e^{-\frac{1}{\beta_k^2}\|\mathbf{z}-\mathbf{z}'\|^2} e^{-\frac{1}{\sigma_k^2}\left\|\tilde{\varphi}_{k-1}(\mathbf{z}_k+\mathbf{z})-\tilde{\varphi}'_{k-1}(\mathbf{z}'_k+\mathbf{z})\right\|^2_{\mathcal{H}_{k-1}}} \\ &= \left\langle \varphi_k(\mathbf{z}_k), \varphi'_k(\mathbf{z}'_k)\right\rangle_{\mathcal{H}_k} \end{aligned} \tag{1}$$

- Coordinates $\mathbf{z}_k$ apply to levels Ω_k *and* Ω_{k-1}.
- *Not* average kernel on features over whole image - just the average on two (local) patches

Next: kernel defined on a Hilbert space input



Slightly misleading: so far, the number of coordinates $\mathbf{z}_k$ is identical at all levels. Also features $\varphi_k(\mathbf{z}_k)$ are not known explicitly, infinite dimensional (problem if we want to feed to next layer)

Approximation of the kernel using sums of features

How to approximate the Gaussian kernel?

The goal is to approximate the kernel by a sum of products of features of the inputs. Start with

$$e^{\frac{-1}{2\sigma^2}\|x-x'\|^2} = \left(\frac{2}{\pi\sigma^2}\right)^{m/2} \int_{w\in\mathbb{R}^m} e^{\frac{-1}{2\sigma}\|x-w\|^2} e^{\frac{-1}{2\sigma}\|x'-w\|^2} dw.$$

In high dimensions, use the approximation

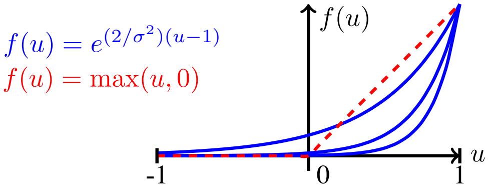
$$\min_{\eta\in\mathbb{R}_+^p, W\in\mathbb{R}^{m\times p}} \left[\frac{1}{n}\sum_{i=1}^{n}\left(e^{\frac{-1}{2\sigma}\|x_i-y_i\|^2} - \sum_{l=1}^{p}\eta_l e^{\frac{-1}{2\sigma}\|x_i-w_l\|^2} e^{\frac{-1}{2\sigma}\|y_i-w_l\|^2}\right)^2\right],$$

where (x_i, y_i) are candidate pairs of points on which you want to evaluate the Gaussian.

Approximation of the kernel using sums of features

Relation to neural network nonlinearity: when $\|x\| = \|y\| = 1$, then $\|w\|$ will be close to 1, and the nonlinear mapping of feature x is

$$u \mapsto e^{(2/\sigma^2)(u-1)} \qquad u = w^\top x$$



η, W optimized via L-BFGS on 300,000 training points, σ by 0.1 quantile of $(\|x_i - y_i\|_2)_{i=1}^n$. "Our goal is to demonstrate the preliminary performance of a new type of convolutional network, and we leave as future work any speed improvement."

Putting it all together

1. Inputs from previous layer: represent $\varphi_{k-1}(\mathbf{z})$ by a vector of length p_{k-1}, written $\xi_{k-1}(\mathbf{z})$.
2. Define $\psi_{k-1}(\mathbf{z}_i)$ to be a patch of these ξ_{k-1} centered at $\mathbf{z}_i$. Dimension is $\mathbb{R}^{|\mathcal{P}_{k-1}|p_{k-1}}$. NOT the same feature space as the average over patches in (1).
3. Compute a vector of p_k output features at each $z_k \in \Omega_{k-1}$,

$$\zeta_k(\mathbf{z}) = \|\psi_{k-1}\|_2 \left[\sqrt{\eta_k} e^{-1/\sigma_k^2 \|\tilde{\psi}_{k-1}(\mathbf{z}) - \mathbf{w}_{kl}\|_2^2} \right]_{l=1}^{p_k}.$$

4. Outputs are pooled with Gaussian weights (after downsampling by a factor of 2),

$$\xi_k(\mathbf{z}) = \sqrt{2/\pi} \sum_{u \in \Omega_{k-1}} e^{-1/\beta_k^2 \|\mathbf{u}-\mathbf{z}\|^2} \zeta_k(\mathbf{z})$$

where the u are summed over a grid in Ω_{k-1}.

Putting it all together