Supplementary Materials of Seeing into Darkness: Scotopic Visual Recognition

Bo Chen Pietro Perona
California Institute of Technology
bchen3, perona@caltech.edu

April 9, 2017

A Appendix

A.1 Time-Adaptation of Hidden Features (Eq. 4)

Here we derive the approximation of the first layer activations $S^H(\boldsymbol{N}_t)$ given photon count image up to time $t\Delta$ (**Eq. 4**), copied as below:

$$S^H(\boldsymbol{N}_t) \approx \alpha(t)\boldsymbol{W}\boldsymbol{N}_t + \boldsymbol{\beta}(t) \tag{A.1}$$

Recall that we put a Gamma prior on the photon emission rate λ_i at pixel i:

$$P(\lambda_i) = Gam(\mu_i\tau, \tau) \tag{A.2}$$

where μ_i is the prior mean rate at pixel i.

After observing $N_{t,i}$ of pixel i in time $[0, t\Delta]$, the posterior estimate for the photon emission rate is:

$$\begin{aligned} P(\lambda_i|N_{t,i}) &\propto P(N_{t,i}|\lambda_i)P(\lambda_i) && \text{(A.3)} \\ &= Gam(\mu_i\tau + N_{t,i}, \tau + t) && \text{(A.4)} \end{aligned}$$

which has a posterior mean of:

$$\hat{\lambda}_i \triangleq \mathbb{E}[\lambda_i|N_{t,i}] = \frac{\mu_i\tau + N_{t,i}}{\tau + t} \tag{A.5}$$

Intuitively, the emission rate is estimated via a smoothed-average of the observed counts. Collectively the expected photon counts $\Delta\boldsymbol{N}$ over all pixels and duration $(t\Delta, T\Delta)$ given the observed photons $\boldsymbol{N}_t$ are:

$$\mathbb{E}[\Delta\boldsymbol{N}|\boldsymbol{N}_t] = \frac{\boldsymbol{\mu}\tau + \boldsymbol{N}_t}{\tau + t}(T - t) \tag{A.6}$$

where $\boldsymbol{\mu}$ is the mean rate vector of all pixels.

Therefore $S^H(\boldsymbol{N}_t)$ may be approximated up to second order accuracy using:

$$\begin{aligned} S^H(\boldsymbol{N}_t) &= \sum_{\Delta \boldsymbol{N}} (\boldsymbol{W}(\boldsymbol{N}_t + \Delta \boldsymbol{N}) + \boldsymbol{b}^H) P(\Delta \boldsymbol{N} | \boldsymbol{N}_t) && \text{(A.7)} \\ &\approx \boldsymbol{W}(\boldsymbol{N}_t + \mathbb{E}[\Delta \boldsymbol{N} | \boldsymbol{N}_t]) + \boldsymbol{b}^H && \text{(A.8)} \\ &= \boldsymbol{W}(\boldsymbol{N}_t + \frac{\boldsymbol{\mu}\tau + \boldsymbol{N}_t}{\tau + t}(T - t)) + \boldsymbol{b}^H && \text{(A.9)} \\ &= \underbrace{\frac{T+\tau}{t+\tau}}_{\alpha(t)} \boldsymbol{W}\boldsymbol{N}_t + \underbrace{\tau \frac{T-t}{\tau+t} \boldsymbol{W}\boldsymbol{\mu} + \boldsymbol{b}^H}_{\boldsymbol{\beta}(t)} && \text{(A.10)} \end{aligned}$$

which proves **Eq. 4**.

The equation above works for weights $\boldsymbol{W}$ that span the entire image. In ConvNet, the weights are instead localized (e.g. occupying only a 5×5 region), and organized into groups (e.g. the first layer in WaldNet for CIFAR10 uses 32 features groups). For simplicity we assume that the mean image $\boldsymbol{\mu}$ is translational invariant within 5×5 regions, so that we only need to model one scalar $\beta_j(t)$ for each feature map W_j.

A.2 Relationship between exposure time and number of bits of signal

Bits of signal and photon counts are equivalent concepts. Furthermore, that photon counts are linearly related to exposure time. Here to derive the relationship between exposure time and the number of bits of signal. To simplify the analysis we will make the assumption that our imaging setup has a constant aperture.

What does it mean for an image to have a given number of bits of signal? Each pixel is a random variable reproducing the brightness of a piece of the scene up to some noise. There are two main sources of noise: the electronics and the quantum nature of light. We will assume that for bright pixels the main source of noise is light. This is because, as will be clear from our experiments, a fairly small number of bits per pixel are needed for visual classification, and current image sensors and AD converters are more accurate than that.

According to the Poisson noise model (**Eq. 1** in main text), each pixel receives photons at rate λ. The expected number of photons collected during a time t is λt and the standard deviation is $\sigma = \sqrt{\lambda t}$. We will ignore the issue of quantum efficiency (QE), i.e. the conversion rate from photons to electrons on the pixel's capacitor, and assume that QE=1 to simplify the notation (real QEs may range from 0.5 to 0.8). Thus, the SNR of a pixel is $SNR = \lambda t / \sqrt{\lambda t} = \sqrt{\lambda t}$ and the number of bits of signal is $b = \log_2 \sqrt{\lambda t} = 0.5 \log_2 \lambda + 0.5 \log_2 t$.

The value of λ depends on the amount of light that is present. This may change dramatically: from 10^{-3} LUX in a moonless night to 10^5 LUX in bright direct sunlight. With a typical camera one may obtain a good quality image in a well lit indoor scene ($E_v \approx 300$ lux) with an exposure time of 1/30s. If a bright pixel has 6.5 bits of signal, the noise is $2^{-6.5} \approx 1\%$ of the dynamic range and $\lambda t / \sqrt{\lambda t} = 100$, i.e. $\lambda \approx 3 \cdot 10^5 \approx$

	Illuminance	exposure time t (s)					
Scene	E_v (LUX)	1/500	1/128	1/8	1	8	60
Moonless	10^{-3}					1.5	3
Full moon	1	0.5	1.5	3.5	5	6.5	8
Office	250	4.5	5.5	7.5	9	10.5	12
Overcast	10^3	5.5	6.5	8.5	10	11.5	13
Bright sun	10^5	9	10	12	13.5	15	16.5

Table 1: (Approximate) number of bits of signal per pixel under different illuminance levels. For instance, in an office scene it takes $1/8$ seconds to obtains a 7.5-bit image. Under full moon, the same high-quality image and the same sensor needs > 8 seconds to capture.

$10^3\mathrm{E}_v \approx 2^{10}\mathrm{E}_v$. Substituting this calculation of λ into the expression derived in the previous paragraph we obtain $b \approx 5 + \frac{1}{2}\log_2 t + \frac{1}{2}\log_2 \mathrm{E}_v$, which is what we used to generate **table 1**.

A.3 Learning dynamic threshold for Bayes risk minimiziation (Eq. 7)

Here we show how thresholds $\tau_\eta(t)$ relate to Bayes risk (**Eq. 5**) in the free-response regime with a cost of time η. The key is to compute $R_t^{(n)}$, the cumulative future risk from time t for the n-th example $\boldsymbol{N}_t^{(n)}$ with label $C^{(n)}$. At every point in time, the classifier first incurs a cost η (assuming time unit of 1) in collecting photons for this time point. Then the classifier either report a result according to $S_c(\boldsymbol{N}_t^{(n)})$, incurring a lost when the predicted label is wrong, or decides to postpone the decision till later, incurring lost $R_{t+1}^{(n)}$. Which one of the two paths to take is determined by whether the max log posterior crosses the dynamic threshold $\tau_\eta(t)$. Therefore, let $c^* = \arg\max_c S_c(\boldsymbol{N}_t^{(n)})$ be the class with the maximum log posterior, the recursion is:

$$R_t^{(n)} = \eta\Delta + \mathbb{I}[S_{c^*}(\boldsymbol{N}_t^{(n)}) > \tau_\eta(t)][c^* \neq C^{(n)}] + \mathbb{I}[S_{c^*}(\boldsymbol{N}_t^{(n)}) \leq \tau_\eta(t)]R_{t+1}^{(n)} \tag{A.11}$$

$$= \eta\Delta + q_t^{(n)} e_t^{(n)} + (1 - q_t^{(n)})R_{t+1}^{(n)} \tag{A.12}$$

and we assume that a decision must be taken after finite amount of time, i.e. $R_\infty^{(n)} = \eta\Delta + e_\infty^{(n)}$. This proves **Eq. 7**.

A.4 Spiking recurrent neural network implementation

Here we show that the recurrent dynamics described in **Eq. 8** implements the approximation of the first hidden layer activations in **Eq. 4**. We use $V(t)$ to denote the membrane potential in the spiking network. The proof is constructive: assume that at time $(t-1)\Delta$, the membrane potential $V(t-1)$ computes $S^H(\boldsymbol{N}_{t-1})$, i.e. $V(t-1) = \alpha(t-1)\boldsymbol{W}\boldsymbol{N}_{t-1} + \boldsymbol{\beta}(t-1)$, then the membrane potential at time $t\Delta$

satisfies:

$$V(t) = r(t)\boldsymbol{V}(t-1) + \alpha(t)\boldsymbol{W}\mathbf{X}_t + \boldsymbol{l}(t) \tag{A.13}$$

$$= r(t)\left(\alpha(t-1)\boldsymbol{W}\boldsymbol{N}_{t-1} + \boldsymbol{\beta}(t-1)\right) + \alpha(t)\boldsymbol{W}\mathbf{X}_t + \boldsymbol{\beta}(t) - r(t)\boldsymbol{\beta}(t-1) \tag{A.14}$$

$$= \frac{\alpha(t)}{\alpha(t-1)}\alpha(t-1)\boldsymbol{W}\boldsymbol{N}_{t-1} + \alpha(t)\boldsymbol{W}\mathbf{X}_t + \boldsymbol{\beta}(t) \tag{A.15}$$

$$= \alpha(t)\boldsymbol{W}(\boldsymbol{N}_{t-1} + \mathbf{X}_t) + \boldsymbol{\beta}(t) = \alpha(t)\boldsymbol{W}\boldsymbol{N}_t + \boldsymbol{\beta}(t) = S^H(\boldsymbol{N}_t) \tag{A.16}$$

Hence proving **Eq. 8**.

A.5 Datasets

MNIST contains gray-scaled 28×28 images of 10 hand-written digits. It has $60k$ training and $10k$ test images. We treat the pixel values as the ground truth intensity[1]. Dark current $\epsilon_{dc} = 3\%$. We use the default LeNet architecture from the MatConvNet package [3] with batch normalization [2] after each convolution layer. The architecture is 784-20-50-500-10[2] with 5×5 receptive fields and 2×2 pooling.

CIFAR10 contains 32×32 color images of 10 visual categories. It has $50k$ training and $10k$ test images. We use the same sythensis procedure above to each color channel[3]. We again use the default 1024-32-32-64-10 LeNet architecture [1] with batch normalization. We use the same setting prescribed in [1] to achieve 18% test error on normal lighting conditions. [1] uses local contrast normalization and ZCA whitening as preprocessing steps. We estimate the local contrast and ZCA from normal lighting images and transforming them according to the lowlight model to preprocess scotopic images.

A.6 Training

We train all models for MNIST and CIFAR10 using stochastic gradient descent with mini-batches of size 100. For MNIST, we use $5k$ training examples for validation and train on the remaining $55k$ examples for 80 iterations. We found that empirically a learning rate of 0.004 works best for WaldNet, and 0.001 works best for the other architectures. As CIFAR10 is relatively data-limited, we do not use a validate set and instead train all models for 75 epochs, where the learning rate is 0.05 for 30 iterations, 0.005 for other 25 then 0.0005 for the rest. Again, quadrupling the learning rate empirically improves WaldNet's performance but not the other architectures.

Our implementation is based on MatCovNet [3], and will be released upon acceptance.

[1]The brightest image we synthesize has about 2^8 photons, which corresponds to a pixel-wise maximum signal-to-noise ratio of 16 (4-bit accuracy), whereas the original MNIST images has (7 to 8-bit accuracy) that corresponds to 2^{14} to 2^{16} photons.

[2]The first and last number represent the input and output dimension, each number in between represents the number of feature maps used for that layer. The number of units is the product of the number of features maps with the size of the input.

[3]For simplicity we do not model the Bayer filter mosaic.

In step one of learning, the scalar functions $\alpha(t)$ and $\beta_j(t)$ in **Eq. 4** are learned as follows. As the inputs to the network are preprocessed, the preprocessing steps alter the algebraic form for α and β. For flexibility we do not impose parametric forms on α and β, but represent them with piecewise cubic Hermite interpolating polynomials with four end points at PPP$=[.22, 2.2, 22, 220]$ (interpolants coded in log-scale). We learned the adapted weights at these end-points by using a different batch normalization module for each PPP. At test time the parameters of the modules are interpolated to accommodate other PPP levels.

In step two of learning, we compute $S^H(\boldsymbol{N}_t)$ for 50 uniformly spaced PPPs in log scale, and train thresholds $\tau(t)$ for each PPP and for each η. A regularizer $0.01\sum_t ||\tau(t) - \tau(t+1)||^2$ is imposed on the thresholds $\tau(t)$ to enforce smoothness. The steepness of Sigmoid σ is annealed over 500 iterations of gradient descent, with initial value 0.5, a decay rate of 0.99 and a floor value of 0.01.

A.7 Rotational jitter

To investigate WaldNet's robustness to camera motion, we inject a rotational jitter to the photon streams. The rotation at PPP follows a normal distribution: $\Delta\theta \sim \mathcal{N}(0, (\frac{\sigma_\theta PPP}{220})^2)$, where σ_θ controls the level of jitter. e.g. $\sigma_\theta = 22^\circ$ means that at $PPP = 220$, the total amount of rotation applied to the image has an std of 22°. The result is shown in **Fig. 5a,b**.

References

[1] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, 2012. 4

[2] C. S. Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. volume 32, pages 448–456, 2015. 4

[3] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. 2015. 4