ESTIMATING PARAMETERS FOR SPATIAL POOLING WITHIN THE WALNUTIQ NEOCORTEX MODEL WITH QNSTOP

STUDENTS: QUINN LIU AND BRANDON AMOS
MENTOR: DR. LAYNE WATSON

{QUINNLIU,BDAMOS}@VT.EDU

DEPT. OF COMPUTER SCIENCE
VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY
BLACKSBURG, VIRGINIA, USA

Abstract. Cortical Learning Algorithms (CLA) are a set of biologically inspired machine learning techniques modeling the human neocortex. This paper presents an implementation and study on the spatial pooling portion of CLA applied to bitmap images. Experiments show that changing internal parameters of spatial pooling can make biologically infeasible results. We present two cost functions and apply the QNSTOP parameter estimation algorithm to the cost functions to obtain the set of parameters corresponding to the most biologically feasible results.

1. Introduction. Emerging Trends and Challenges. Recent work by Hawkins at Numenta [4] describe a set of machine learning algorithms called Cortical Learning Algorithms (CLA) to realistically model the human neocortex. The human neocortex performs diverse cognitive functions such as vision, hearing, touch, movement, and language, but the underlying neural activity for these cognitive functions is believed to be governed by a common learning algorithm. The spatial pooling part of the CLA is biologically inspired and efficient at generating consistent output when varying the amount of noise in the input. In this paper, we apply the spatial pooling portion of the CLA to represent black and white bitmap images efficiently.

Open Problem: Tuning parameters for spatial pooling. The spatial pooling model has parameters that can be tuned and control how neurons are connected to an input image. Changing how the neurons are connected affect the output of the spatial pooling algorithm, and some parameters can result in biologically infeasible results. Spatial pooling is used as a component of other algorithms in CLA, and having biologically feasible results is desirable.

Contribution: Biologically stable results from spatial pooling We propose objective functions in which the minimum values correspond to consistent biologically stable results for the WalnutiQ implementation of spatial pooling on a set of input images. The objective function is optimized with the parameter estimation algorithm QNSTOP.

The remainder of this paper is organized as follows. Section 2 details the WalnutiQ CLA implementation; Section 3 describes the QNSTOP algorithm for parameter estimation; Section 4 presents empirical results; and Section 5 presents concluding remarks, lessons learned, and future work.

2. WalnutiQ. WalnutiQ is an experimental implementation of Hawkins[4] CLA algorithm implemented in Java and open sourced at `https://github.com/WalnutiQ/WalnutiQ`. The implementation of the spatial pooling algorithm is considered here.

The WalnutiQ implementation of the spatial pooling algorithm can be understood by imagining a simplified retina as a chessboard of cells in the retina that are either active represented with a black cell or inactive represented with a white cell. An example of **input** to the retina would be the image "0" shown in Figure 1.

The spatial pooling algorithm connects the **retina** to multiple regions with **synapses**, where each region is smaller than the previous region. In Figure 1, the **bottom re-**

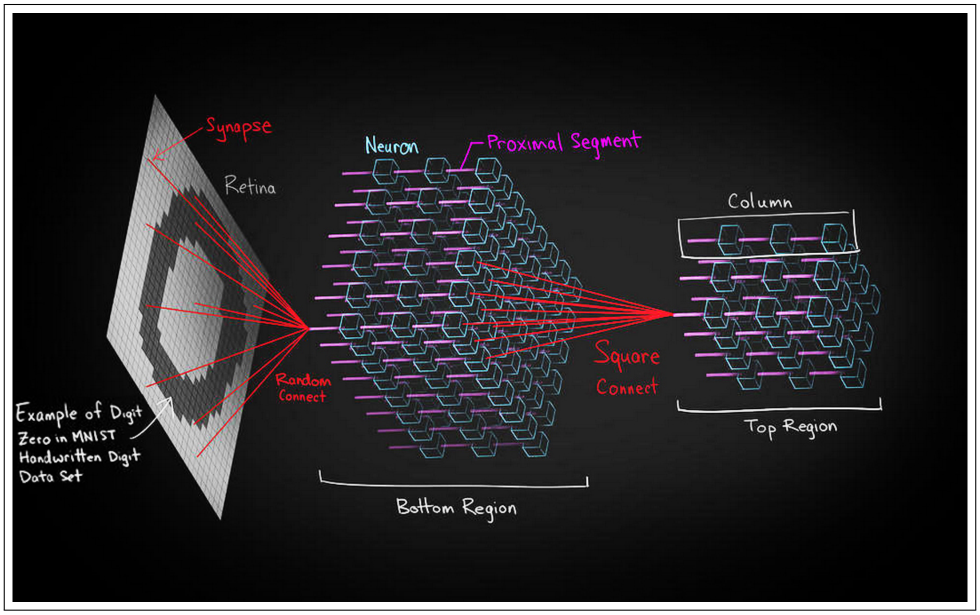

FIG. 1. *Example of brain model able to be built with WalnutiQ to illustrate the terminology.*

gion consists of columns and reads from the retina, and the **top region** also consists of columns and reads from the bottom region. In this implementation, columns have 2 states: active columns are represented by a black grid section and inactive columns are represented by a white grid section.

2.1. Objective function for fitting spatial pooling parameters. The current spatial pooling implementation takes a bitmap B on input and returns a smaller grid G of active columns. Realistic results from spatial pooling should have sparse neural activity and evenly distributed active columns. Consider an input image of a balloon tied with a long piece of string to illustrate why distributed output grids are desirable. If the points were clustered, the active columns representing the string would likely not be included in the output. However, a distribution of the active columns results in active columns representing both the balloon and string.

An objective function $f : \mathbb{R}^3 \to \mathbb{R}$, where the input are parameters to spatial pooling and the output is the score of the output grid of active columns. Define the input parameters to spatial pooling as

$$\vec{x} = (\text{percentMinOverlapScore}, \text{desiredLocalActivity}, \text{desiredPercentOfActiveColumns}),$$

where `percentMinOverlapScore` is the amount of overlap cells read in the input bitmap image, `desiredLocalActivity` is the desired percentage of neural activity, and `desiredPercentOfActiveColumns` is the desired percentage of active columns in the output grid. Let $G(\vec{x})$ be the output grid after performing spatial pooling on the input vector.

Compute the sparsity of the output grid on a single bitmap B with

$$s(G_B(\vec{x})) = -\frac{1}{|e(G_B(\vec{x})) - a(G_B(\vec{x}))|},$$

where $e(G_B(\vec{x}))$ is the expected number of active neurons for the grid and $a(G_B(\vec{x}))$ is the actual number of active neurons. Furthermore, define the distribution of grid points with

$$d(G_B(\vec{x})) = -\sum_{p \in G_B(\vec{x})} \sqrt{|p^2 - c(p)^2|},$$

where p is a point in the output grid and $c(p, G_B(\vec{x}))$ is a function returning the closest other grid point to p using the Euclidean distance.

Define an objective function f_B for a single bitmap image B by

$$f_B(G_B(\vec{x})) = s(G_B(\vec{x})) + d(G_B(\vec{x})).$$

The smaller values of f_B correspond to better output grids for the input bitmap. Furthermore, f_B can be applied to a set of bitmaps $\vec{B}$ with

$$f_{\vec{B}}(G_{\vec{B}}(\vec{x})) = \sum_{B \in \vec{B}} f_B(G_B(\vec{x})).$$

3. QNSTOP. QNSTOP is a class of quasi-Newton methods originally developed for stochastic optimization by Castle et al. [2] and can also be used for deterministic global optimization with minor variations as shown by Easterling et al. [3]. QNSTOP combines ideas from from numerical optimization (secant updates and trust regions) and response surface methodology (ridge analysis). The Fortran QNSTOP implementation presented by Amos et al. [1] is used for the experiments in this paper.

The Latin hypercube sampling mode of the QNSTOP implementation is used to sample points from the feasible set Θ, which is a convex subset of $\mathbb{R}^p$. For each start point, QNSTOP progresses in iteration k and computes the gradient vector $\hat{g}_k$ and Hessian matrix $\hat{H}_k$ of a quadratic model

$$\widehat{m}_k(X - X_k) = \hat{f}_k + \hat{g}_k^T (X - X_k) + \frac{1}{2} (X - X_k)^T \hat{H}_k + (X - X_k)$$

of the objective function f centered at X_k, where $\hat{f}_k$ is generally not $f(X_k)$.

In the unconstrained context, QNSTOP methods progress by

$$X_{k+1} = X_k - \left[\hat{H}_k + \mu_k W_k\right]^{-1} \hat{g}_k$$

where μ_k is the Lagrange multiplier of a trust region subproblem and W_k is a scaling matrix. The iterate is then updated with

$$X_{k+1} = \left(X_k - \left[\hat{H}_k + \mu_k W_k\right]^{-1} \hat{g}_k\right)_\Theta,$$

where $(\cdot)_\Theta$ denotes projection onto Θ.

4. Empirical Results.

4.1. Experiment 1: Optimizing spatial pooling on a single bitmap. Consider the handwritten number "2" shown in Figure 2 as B_2. The objective function $f_{B_2}(G_{B_2}(\vec{x}))$ described in Section 2.1 is used as an input to QNSTOP to obtain the optimal set of parameters $\vec{x}$ to yield the best grid G_{B_2}.

FIG. 2. 66×66 *input bitmap used in Experiment 1.*

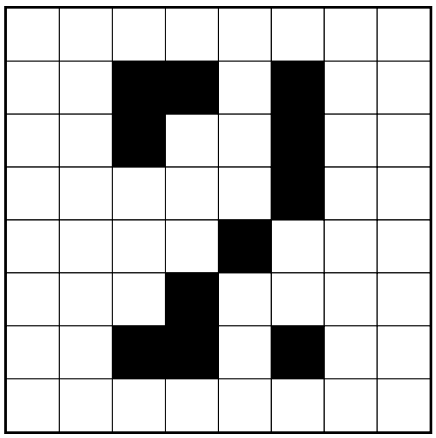

FIG. 3. 8×8 *output grid for the* **unoptimized** *parameter vector for recognizing the handwritten bitmap in Experiment 1.*

QNSTOP uses an input vector, chosen with reasonable values of $(50, 3, 10)$, which result in a score of -1.33 and an output grid $G_{B_2}(50, 3, 10)$ shown in Figure 3. This grid is suboptimal because the active neurons are clustered too closely together.

QNSTOP uses Latin hypercube sampling with 5 starting points including the input vector, a feasible set $\Theta = ([1, 99], [1, 10], [0, 100])$. For each starting point an iteration limit of 20 is used with 20 function evaluations per iteration. The radius of the experimental design region and trust region $\tau = 20$ and is decayed with $g = 15$.

The QNSTOP experiment results in 2105 total function evaluations for all 5 start points. Figure 4 shows QNSTOP's progression for each starting point as different lines. This shows that finding an optimal parameter vector is difficult to maintain.

The global minimum is found to be $(77.8, 1, 7.43)$ with a score of -12.4, which corresponds to the output grid shown in Figure 5. Recall the active columns are represented as black cells. The output shown in Figure 5 has a better score than the output in Figure 3 because the active columns are sparser and distributed. The adjacent active columns in Figure 3 does not efficiently represent the input image "2". Furthermore, the distributed black cells represent the majority of the key parts of the input image.

4.2. Experiment 2: Optimizing the MNIST set of images. Fitting the WalnutiQ spatial pooling model to a single image as in Section 4.1 might result

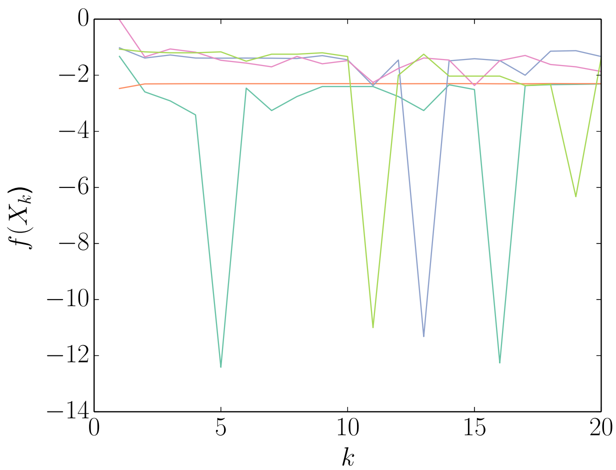

FIG. 4. *QNSTOP progression in each iteration k for each starting point for the single image in Experiment 1.*

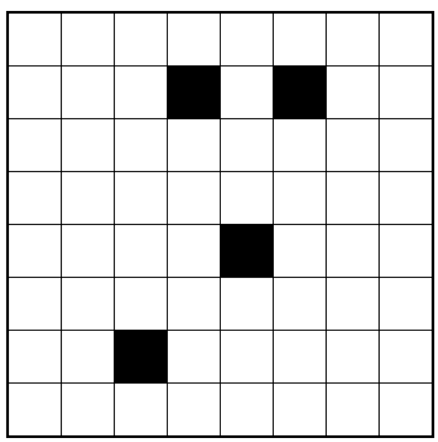

FIG. 5. 8×8 *output grid for the* **optimized** *parameter vector for recognizing the handwritten bitmap in Experiment 1.*

in WalnutiQ parameters well-suited to a specific image, but the parameters might still perform poorly on another image. To fix this, the WalnutiQ model should be evaluated on a set of images by using the objective function $f_{\vec{B}}$ in Section 2.1 to find the parameters that perform best for a set of images.

The MNIST database [5] is a database of 70,000 handwritten digits that have been size-normalized and centered in a fixed-size image and suitable for input into the WalnutiQ spatial pooling model. This experiment fits the WalnutiQ spatial pooling model to 1,000 images in MNIST database, $\vec{B}$.

QNSTOP requires an input vector, chosen with reasonable values of $(50, 3, 10)$ as in Experiment 1, which results in a score of -1.25.

QNSTOP uses Latin hypercube sampling with 5 starting points including the input vector, a feasible set $\Theta = ([1, 99], [1, 10], [0, 100])$. For each starting point an iteration limit of 20 is used with 20 function evaluations per iteration. The radius of the experimental design region and trust region $\tau = 20$ and is decayed with $g = 15$.

The QNSTOP experiment results in 2105 total function evaluations for all 5 start points. Figure 6 shows QNSTOP's progression for each starting point as different lines. This shows the function is smoother that in Experiment 1 and easier for QNSTOP to explore the optimal region.

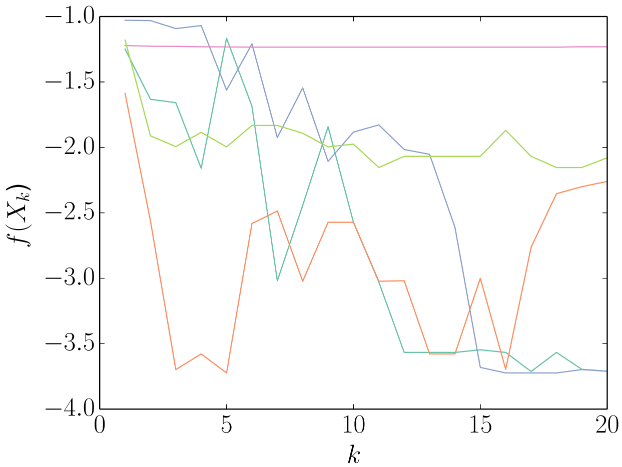

FIG. 6. *QNSTOP progression in each iteration k for each starting point for the MNIST subset used in Experiment 2.*

The QNSTOP experiment results in a global minimum of -3.72 with a parameter vector of $(45.9, 1, 9.76)$. This parameter vector performs the best for 1,000 images, and should also perform well when applied to other images. The handwritten image in Figure 2 is not used in this experiment, but applying this parameter vector to it results in the grid shown in Figure 7, showing a near optimal grid.

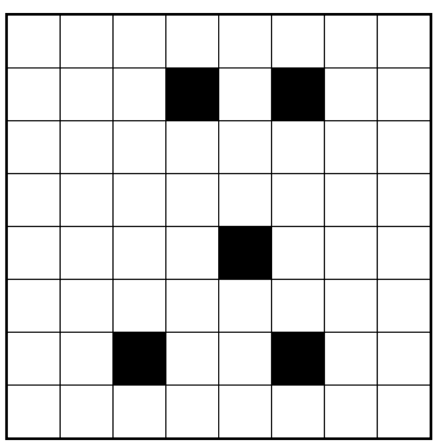

FIG. 7. 8×8 *output grid for the* **optimized** *parameter vector for the MNIST dataset applied to the handwritten bitmap used in Experiment 1.*

5. Conclusion. Experiment 1 shows that optimizing the parameters for spatial pooling on a single digit produces a good representation of the input image. More importantly, Experiment 2 shows that the parameters for producing an optimal sparse and distributed output on 1000 different digits can variety greatly and still produce a good representation of the single image from Experiment 1. Finding parameters to spatial pooling resulting in biologically feasible output grids is key to replicating realistic neural activity with CLA. Future work includes implementing the remaining algorithms in the CLA and running sensory motor vision experiments involving sacadding retinas and biologically inspired prediction algorithms. You can follow our work by starring the open source project at `https://github.com/WalnutiQ/WalnutiQ`.

REFERENCES

[1] Brandon Amos, David Easterling, Layne Watson, Brent Castle, Michael Trosset, and William Thacker, *Fortran 95 implementation of qnstop for global and stochastic optimization*, in SpringSim14 High Performance Computing Symposium, 2014.

[2] Brent Castle, *Quasi-Newton Methods for Stochastic Optimization and Proximity-Based Methods for Disparate Information Fusion*, PhD thesis, School of Informatics and Computing, Indiana University, 2012.

[3] David R Easterling, Layne T Watson, Michael L Madigan, Brent S Castle, and Michael W Trosset, *Parallel deterministic and stochastic global minimization of functions with very many minima*, Computational Optimization and Applications, (2011), pp. 1–24.

[4] J. Hawkins and D. George, *Hierachical temporal memory including htm cortical learning algorithms*, Whitepaper, Numenta, Inc., 2011.

[5] Yann LeCun, Corinna Cortes, and Christopher Burges, *The mnist database of handwritten digits*, AT&T Labs [Online]. Available: `http://yann.lecun.com/exdb/mnist`, (2010).