A Biologically Inspired Image Classifier: Adaptive Feature Detection

Jeffrey C. Ames

Konstantinos P. Michmizos

Abstract—Today’s artificial neural networks use computational models and algorithms inspired by the knowledge of the brain in the ‘90s. Powerful as they are, artificial networks are impressive but their domain specificity and reliance on vast numbers of labeled examples are obvious limitations. About a decade ago, spiking neural networks (SNNs) emerged as a new formalism that takes advantage of the spike timing and are particularly versatile when depicting spatio-temporal representations. The challenge now is to design rules for SNNs that can help them interact with their environment just like humans do. Specifically for visual classification tasks, we need to design a set of simple features that can describe any input, seen and unseen, by adapting to the environment. Herein, we propose an adaptive mechanism for deducing feature detectors from input data. Our proposed method adapts online to new instances of existing categories pooled from the MNIST database of handwritten numbers. The extracted features are comparable to those found in biological neural networks for certain classes of inputs. We anticipate that our proposed model will be embedded in our ongoing effort to design an SNN for image classification.

I. INTRODUCTION

Understanding intelligence is a fundamental endeavor for several different disciplines. Translating our understanding of intelligence to machines is a fundamental problem in computing. The original work of McCulloch & Pitts in 1943[1] introduced the notion of binary state (“active” vs. “not-active”) models of neurons that became the computational units in the first generation of artificial neural networks (ANNs), namely multilayer perceptrons[2][3], Hopfield networks[4], and Boltzmann machines[5]. Subsequent neuronal models introduced continuous inputs and outputs connected through an “activation function” such as the sigmoid and the linear saturated functions[6]. In addition to being more biologically realistic, the resulting second-generation ANNs allowed for efficient analog computations[7] and gave rise to artificial intelligence systems that can solve complex tasks considered out of reach just a few years ago; for a review see [8].

Impelled by the experimental evidence accumulating over a decade of neurophysiological studies on the importance of spike timing in encoding information[9][10] and the related spike-timing dependent plasticity[11], a third generation of neural networks has recently emerged. The Spiking Neural Networks (SNNs)[12] employ spiking neurons as their computational units that regard the presence and timing of individual spikes as the means of communication and neural computation. SNNs compare with traditional ANNs where analog values are considered, representing the rate at which spikes are fired; these models can do everything the older models could, and often with many fewer neurons[13], and they perform particularly well in spatio-temporal pattern tasks. The challenge is that there are no known design rules to devise an SNN for a given task; the modeler is on his own.

Replicating human vision is one of the dominant applications for ANNs. Historically, neural networks have been successful in a wide range of object recognition tasks, varying from face detection[14] to reading bank checks[15]. Over the last few years, ANNs, empowered with deep learning techniques, have particularly advanced object recognition. For example, GoogLeNet, which is a deep convolutional neural network based on the Hebbian principle and scale invariance, set the new standards for classification and detection of still images[16]. While ANNs approaches are beginning to rival human performance in certain tasks, the gap between human and machine vision capabilities remains large. The most important divides between biological and artificial systems include the size of required training datasets as well as the variability and the noise that they can handle.

Unlike most computer vision test benchmarks that employ the so-called “closed-set” inputs, the classes and the conditions that a system will encounter in real life cannot be known in advance. In fact, a large fraction of real-world images that a machine encounters carry an intrinsically ambiguous label. Even “open-set” benchmarks such as the MNIST hand-written digit recognition dataset[17] are typically tested after training the network with all possible digit labels that it might encounter; for a recent example with 99% accuracy on MNIST see [18]. This concern is also echoed in large-scale object datasets, such as the ImageNet challenge[19], where large variability is expected. Current approaches to solve the problem of large variability aim to create enormous number of categories and simplify the object classification problem to a simple mapping. Interestingly enough, even robust deep network classifiers break up easily in the presence of noise in the input[20].

To address these concerns, we are currently developing an SNN that can classify highly varying, ambiguous and noisy inputs. To allow for this variability to be efficiently processed, it is critical for the SNN to learn from experience. Our first line of attack is modeling the simple-cell receptive field, where pixel intensities in the image, in analogy to photons in the retina, are translated to basic types of neural representations. It is widely known that humans break up an image into relevant features (e.g., horizontal or vertical lines, corners, curves, etc.) for subsequent classification. It
has been noted that the cat striate cortex appears to make use of receptive field patterns much like Gabor filters[21]. We may ask how these patterns could develop – are they hard-wired via genetics or developed by interacting with the environment? Blakemore and Van Sluyters showed that development of feature detectors in kittens is strongly affected by their environment[22], which suggests a similar role for development in humans.

In this paper, we describe the development of a simple-cell receptive field that adapts to its environment and therefore can natively create new basis functions as needed. Specifically, we use the notion of biologically-realistic grating-like filters initially proposed by Olshausen and Field [23][24]. To this end, we make use of a related model, given in [25], that develops a set of basis functions by observing a set of inputs over time. The input is taken from the MNIST database. The basis functions are initially random, and are iteratively modified to give better approximations to the input.

II. METHODS

We work towards building a classifier for handwritten digits from 0 to 9, using the existing MNIST data set[17], which includes both images (as 28 × 28 arrays of 8-bit grayscale values) and labels (indicating the correct classification).

Algorithm 1 Find efficient basis functions

function PROBABILISTICCLOSESTBASIS(tile, φ)
projections = [φᵢ · tile for φᵢ in φ]
x = index of random projection weighted by value
return φₓ, φₓ · tile
end function

function FINDBASES(images)
φ = [2 * random() - 0.5] * 14*14] * 64
for t in 1..numTrials do
    image = pickRandomImage(images)
    tile = pickRandomTile(image)
    for r in 1..numRounds do
        φᵢ, ω = probabilisticClosestBasis(tile, φ)
        φᵢ += ω(φᵢ - φᵢ)
        tile -= ωφᵢ
    end for
end for
return φ
end function

For the first step, we use a method based on [25] to learn a set of basis functions based on this input set. The method develops a set of basis functions by observing a set of inputs over time. The basis functions are initially random, and are iteratively modified to give better approximations to the input. We select a random image, break it into discrete tiles, and use a random tile to modify our basis functions. In each round, we pick the basis function which best approximates the image (i.e., it has the greatest projection of the image). Allowing for multiple rounds for the same image results in more than one basis function being used to approximate the image. In the algorithm (Algorithm 1), φ is the set of basis functions and η is the learning rate.

We use 64 basis functions, 4000 trials, and 64 rounds per trial, with a learning rate of 0.01. This number of basis functions is likely more than necessary in this space, but since orthogonality is not required, we are free to oversupply the algorithm. We choose 4000 trials so that approximately 100 samples of each digit can be presented, from each of which one tile (of size 14 × 14) is selected.

Algorithm 2 Reconstruct an input image

function CLOSESTBASIS(tile, φ)
projections = [φᵢ · tile for φᵢ in φ]
x = index of max projection
return φₓ, φₓ · tile
end function

function RECONSTRUCT(tile, φ)
recTile = 0
for r in 1..numRounds do
    φᵢ, ω = closestBasis(tile, φ)
    recTile += ωφᵢ
    tile -= ωφᵢ
    φ = φ \ {φᵢ}
end for
return recTile
end function

We also make use of a reconstruction algorithm (Algorithm 2), given a tile to reconstruct and a list of basis functions. It is similar to the training method in that it uses projections of the input against each basis to select the most similar basis.

III. RESULTS

Training the system as described yields feature detectors as shown in Figure 1. We can then utilize these to reconstruct input digits, using the reconstruction algorithm given earlier: we match basis functions to the input based on their projection, repeating the process with the remainder until all bases have been used. The resulting reconstructions for a sample set of inputs are shown in Figure 2.

To measure the accuracy of reconstruction, we quantify the error over trials as the root mean squared (RMS) error based on the difference between the original image tile and the reconstructed image tile. Figure 3 shows the RMS error over the first 60 trials of a typical run.

Furthermore, the algorithm can adapt to new inputs. In this case, we train it on one instance of a “7” for 60 trials before introducing a new “7”. We repeat this again after 60 trials of the second instance, and the RMS error shows how the features adapt over time to the newer stimuli (Figure 4).

We also test how varying the number of rounds affects the ability of the algorithm to effectively learn feature detectors (Figure 5). For each value of the number of rounds, we run the algorithm for 4000 trials and then average the
reconstruction error in the last 1000 trials. The error bars denote the standard error.

IV. DISCUSSION

In this paper, we presented a biologically plausible mechanism that learned to represent incoming visual information, even given no prior knowledge of the stimulus. We showed that the system adapted to novel stimuli after a brief period of adjustment.

The derived features allowed for a faithful reconstruction of the input, with a few caveats. Specifically, the image
appeared slightly blurred, and there was high-frequency noise around the edges of the digits. Increasing the number of available basis functions yielded better reconstructions.

We also showed how this system can adapt to new inputs—initially with higher than normal errors in image reconstruction, but quickly settling back to the baseline. We further demonstrated that a greater number of basis functions offers a real benefit to the system in terms of learning and representing input images more effectively.

We propose this as a possible first layer of a neuro-inspired approach to classification, e.g., as in combination with a network of spiking neurons. Using derived feature detectors confers two benefits to such a system. First, the weights from the inputs to the initial layers do not need to be hard-coded using existing distributions (e.g., Gabor filters). And second, this lends a robustness to variance in the inputs that a static system would lack, whether due to changes in the stimuli over time or due to unexpected, previously unseen stimuli.

An SNN-based system has many advantages compared to the traditional ANNs. Notably, SNNs use temporal and spatial information, so they can be used for “real” dynamic environments. As explained in Section I, spiking neurons can transmit and receive information through spikes’ time. This can result in fast and efficient implementations, such as the ones in IBM’s TrueNorth[26]. What is more, compared with the classical ANNs, SNNs communicate through spikes and therefore have better robustness to noise. Interestingly enough, spikes can be modeled relatively easily by digital circuits[27]. Since SNNs can compute any analog function that a second generation ANN can and usually with fewer neurons, a neuro-chip based on SNNs will have smaller size and consume less power.

V. CONCLUSION

Despite the abundance of fast accumulating knowledge from neuroscience, research mainly focuses on the molecular, spiking, and synaptic mechanisms, rather than the computing essence of these neural processes. Here, we show a biologically plausible method that allows neural networks to adapt feature detectors depending on their sensed environment. We show how our method adapts to new input stimuli to update these features, in the hopes that such methods will further help computational scientists in their quest for more intelligent algorithms.

REFERENCES