Deep Predictive Coding Network with Local Recurrent Processing for Object Recognition

Kuan Han1,3, Haiguang Wen1,3, Yizhen Zhang1,3, Di Fu1,3, Eugenio Culurciello1,2, Zhongming Liu1,2,3,∗

1School of Electrical and Computer Engineering, Purdue University
2Weldon School of Biomedical Engineering, Purdue University
3Purdue Institute for Integrative Neuroscience, Purdue University

Abstract

Inspired by “predictive coding” - a theory in neuroscience, we develop a bi-directional and dynamic neural network with local recurrent processing, namely predictive coding network (PCN). Unlike feedforward-only convolutional neural networks, PCN includes both feedback connections, which carry top-down predictions, and feedforward connections, which carry bottom-up errors of prediction. Feedback and feedforward connections enable adjacent layers to interact locally and recurrently to refine representations towards minimization of layer-wise prediction errors. When unfolded over time, the recurrent processing gives rise to an increasingly deeper hierarchy of non-linear transformation, allowing a shallow network to dynamically extend itself into an arbitrarily deep network. We train and test PCN for image classification with SVHN, CIFAR and ImageNet datasets. Despite notably fewer layers and parameters, PCN achieves competitive performance compared to classical and state-of-the-art models. Further analysis shows that the internal representations in PCN converge over time and yield increasingly better accuracy in object recognition. Errors of top-down prediction also reveal visual saliency or bottom-up attention.

1 Introduction

Modern computer vision is mostly based on feedforward convolutional neural networks (CNNs) [18, 33, 50]. To achieve better performance, CNN models tend to use an increasing number of layers [19, 24, 50, 59], while sometimes adding "short-cuts" to bypass layers [19, 56]. What motivates such design choices is the notion that models should learn a deep hierarchy of representations to perform complex tasks in vision [50, 59]. This notion generally agrees with the brain's hierarchical organization [31, 62, 67, 16]: visual areas are connected in series to enable a cascade of neural processing [60]. If one layer in a model is analogous to one area in the visual cortex, the state-of-the-art CNNs are considerably deeper (with 50 to 1000 layers) [20, 13] than the visual cortex (with 10 to 20 areas). As we look to the brain for more inspiration, it is noteworthy that biological neural networks support robust and efficient intelligence for a wide range of tasks without any need to grow their depth or width [37].

What distinguishes the brain from CNNs is the presence of abundant feedback connections that link a feedforward series of brain areas in reverse order [11]. Given both feedforward and feedback connections, information passes not only bottom-up but also top-down, and interacts with one another to update the internal states over time. The interplay between feedforward and feedback connections

∗Correspondence to: Zhongming Liu <zmliu@purdue.edu>

has been thought to subserve the so-called "predictive coding" - a neuroscience theory that becomes popular. It says that feedback connections from a higher layer carry the prediction of its lower-layer representation, while feedforward connections in turn carry the error of prediction upward to the higher layer. Repeating such bi-directional interactions across layers renders the visual system a dynamic and recurrent neural network. Such a notion can also apply to artificial neural networks. As recurrent processing unfolds in time, a static network architecture is used over and over to apply increasingly more non-linear operations to the input, as if the input were computed through more and more layers stacked onto an increasingly deeper feedforward network. In other words, running computation through a bi-directional network for a longer time may give rise to an effectively deeper network to approximate a complex and nonlinear transformation from pixels to concepts, which is potentially how brain solves invariant object recognition without the need to grow its depth.

Inspired by the theory of predictive coding, we propose a bi-directional and dynamical network, namely Deep Predictive Coding Network (PCN), to run a cascade of local recurrent processing for object recognition. PCN combines predictive coding and local recurrent processing into an iterative inference algorithm. When tested for image classification with benchmark datasets (CIFAR-10, CIFAR-100, SVHN and ImageNet), PCN uses notably fewer layers and parameters to achieve competitive performance relative to classical or state-of-the-art models. Further behavioral analysis of PCN sheds light on its computational mechanism and potential use for mapping visual saliency or bottom-up attention.

2 Related Work

Predictive Coding

In the brain, connections between cortical areas are mostly reciprocal. Rao and Ballard suggest that bi-directional connections subserve "predictive coding": feedback connections from a higher cortical area carry neural predictions to the lower cortical area, while the feedforward connections carry the unpredictable information (or error of prediction) to the higher area to correct the neuronal states throughout the hierarchy. With supporting evidence from empirical studies, this mechanism enables iterative inference for perception and unsupervised learning, incorporates modern neural networks for classification and video prediction, and likely represents a unified theory of the brain.

Predictive Coding Network with Global Recurrent Processing

Driven by the predictive coding theory, a bi-directional and recurrent neural network has been proposed for object recognition. PCN combines predictive coding and local recurrent processing into an iterative inference algorithm. When tested for image classification with benchmark datasets, PCN uses notably fewer layers and parameters to achieve competitive performance relative to classical or state-of-the-art models. Further behavioral analysis of PCN sheds light on its computational mechanism and potential use for mapping visual saliency or bottom-up attention.

3 Predictive Coding Network

Herein, we design a bi-directional (feedforward and feedback) neural network that runs local recurrent processing between neighboring layers, and we refer to this network as Predictive Coding Network (PCN). As illustrated in Fig. 1, PCN is a stack of recurrent blocks, each running dynamic and recurrent processing within itself through feedforward and feedback connections. Feedback connections are...
used to predict lower-layer representations. In turn, feedforward connections send the error of prediction to update the higher-layer representations. After repeating this processing for multiple cycles within a given block, the lower-layer representation is merged to the higher-layer representation through a bypass connection. The merged representation is further sent as the input to the next recurrent block to start another series of recurrent processing in a higher level. After the local recurrent processing continues through all recurrent blocks in series, the emerging top-level representations are used for image classification.

In the following mathematical descriptions, we use italic letters as symbols for scalars, bold lowercase letters for column vectors and bold uppercase letters for matrices. We use $T$ to denote the number of recurrent cycles, $r_l(t)$ to denote the representation of layer $l$ at time $t$, $W_{l-1,l}$ to denote the feedforward weights from layer $l - 1$ to layer $l$, $W_{l,l-1}$ to denote the feedback weights from layer $l$ to layer $l - 1$ and $W_{l-1,l}^{bp}$ to denote the weights of bypass connections.

### 3.1 Local Recurrent Processing in PCN

Within each recurrent block (e.g. between layer $l - 1$ and layer $l$), the local recurrent processing serves to reduce the error of prediction. As in Eq. (1), the higher-layer representation $r_l(t)$ generates a prediction, $p_{l-1}(t)$, of the lower-layer representation, $r_{l-1}$, through feedback connections, $W_{l,l-1}$, yielding an error of prediction $e_{l-1}(t)$ as Eq. (2).

$$p_{l-1}(t) = (W_{l,l-1})^T r_l(t)$$  \hspace{1cm} (1)

$$e_{l-1}(t) = r_{l-1} - p_{l-1}(t)$$  \hspace{1cm} (2)

The objective of recurrent processing is to reduce the sum of the squared prediction error (Eq. (3)) by updating the higher-layer representation, $r_l(t)$, with an gradient descent algorithm [55]. In each cycle of recurrent processing, $r_l(t)$ is updated along the direction opposite to the gradient (Eq. (4)) with an...
We implement PCN with some architectural features common to modern CNNs. Specifically, Predictive Coding Network with local recurrent processing. The input image

\[
L_{l-1}(t) = \frac{1}{2} \| r_{l-1} - p_{l-1}(t) \|^2
\]

\[
\frac{\partial L_{l-1}(t)}{\partial r_l(t)} = -W_{l,l-1}e_{l-1}(t)
\]

\[
r_l(t+1) = r_l(t) - \alpha_l \frac{\partial L_{l-1}(t)}{\partial r_l(t)} = r_l(t) + \alpha_l W_{l,l-1}e_{l-1}(t)
\]

\[
r_l(t+1) = r_l(t) + \alpha_l (W_{l,l-1,l})^T e_{l-1}(t)
\]

3.2 Network Architecture

We implement PCN with some architectural features common to modern CNNs. Specifically, feedforward and feedback connections are implemented as regular convolutions and transposed convolutions [10], respectively, with a kernel size of 3. Bypass convolutions are implemented as 1 x 1 convolution. Batch normalization (BN) [26] is applied to the input to each recurrent block. During the recurrent processing between layer \( l-1 \) and layer \( l \), rectified linear units (ReLU) [41] are applied to the initial output \( r_l(0) \) at \( t = 0 \) and the prediction error at each time step \( e_{l-1}(t) \), as expressed by Eq. (7) and Eq. (8), respectively. ReLU renders the local recurrent processing an increasingly non-linear operation as the processing continues over time.

\[
r_l(0) = ReLU \left( (W_{l,l-1})^T r_{l-1} \right)
\]

\[
e_{l-1}(t) = ReLU \left( r_{l-1} - p_{l-1}(t) \right)
\]

Besides, a \( 2 \times 2 \) max-pooling with a stride of 2 is optionally applied to the output from every 2 (or 3) blocks. On top of the highest recurrent block is a classifier, including a global average pooling, a fully-connected layer, followed by softmax.

For comparison, we also design the feedforward-only counterpart of PCN and refer to it as the "plain" model. It includes the same feedforward and bypass connections and uses the same classification layer as in PCN, but excludes any feedback connection or recurrent processing. The architecture of the plain model is similar to that of Inception CNN [59].

Algorithm 1 Predictive Coding Network with local recurrent processing.

Input: The input image \( r_0 \);

1: for \( l = 1 \) to \( L \) do
2: \( r_{l-1}^{BN} = \text{BatchNorm}(r_{l-1}) \);
3: \( r_l(0) = \text{ReLU} \left( FFConv \left( r_{l-1}^{BN} \right) \right) \);
4: for \( t = 1 \) to \( T \) do
5: \( p_{l-1}(t) = FBConv \left( r_l(t-1) \right) \);
6: \( e_{l-1}(t) = \text{ReLU} \left( r_{l-1} - p_{l-1}(t) \right) \);
7: \( r_l(t) = r_l(t-1) + \alpha_l FFConv \left( e_{l-1}(t) \right) \);
8: end for
9: \( r_l = r_l(T) + BPConv \left( r_{l-1}^{BN} \right) \);
10: end for
11: return \( r_L \), for classification;
12: \( \triangleright \) FFConv represents the feedforward convolution, FBConv represents the feedback convolution and BPConv represents the bypass convolution.

Our implementation is described in Algorithm 1. Note that the update rate \( \alpha_l \) used for local recurrent processing is a learnable and non-negative parameter separately defined for each filter in each recurrent block. The number of cycles of recurrent processing - an important parameter in PCN, is varied to be \( T = 1, \ldots, 5 \). For both PCN and its CNN counterpart (i.e. \( T=0 \)), we design multiple
architectures (labeled as A through E) suitable for different benchmark datasets (SVHN, CIFAR and ImageNet), as summarized in Table 1. For example, PCN-A-5 stands for a PCN with architecture A and 5 cycles of local recurrent processing.

Table 1: Architectures of PCN. Each column shows a model. We use PcConv to represent a predictive coding layer, with its parameters denoted as “PcConv<kernel size>-<number of channels in feedback convolutions>-<number of channels in feedforward convolutions>”. The first layer in PCN-E is a regular convolution with a kernel size of 7, a padding of 3, a stride of 2 and 64 output channels. "*" indicates the layer applying maxpooling to its output. Feature maps in one grid have the same size.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>SVHN</th>
<th>CIFAR</th>
<th>ImageNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCN Configuration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dataset</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVHN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CIFAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ImageNet</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#Layers</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image Size</td>
<td>32 × 32</td>
<td>224 × 224</td>
<td></td>
</tr>
<tr>
<td>Layers</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Classification</td>
<td>global average pooling, FC-10/100/1000, softmax</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#Params</td>
<td>0.15M</td>
<td>0.61M</td>
<td>4.91M</td>
</tr>
</tbody>
</table>

4 Experiments

We train PCN with local recurrent processing and its corresponding plain model for object recognition with the following datasets, and compare their performance with classical or state-of-the-art models.

4.1 Datasets

CIFAR The CIFAR-10 and CIFAR-100 datasets [32] consist of 32×32 color images drawn from 10 and 100 categories, respectively. Both datasets contain 50,000 training images and 10,000 testing images. For preprocessing, all images are normalized by channel means and standard derivations. For data augmentation, we use a standard scheme (flip/translation) as suggested by previous works [19,23,38,45,24].

SVHN The Street View House Numbers (SVHN) dataset [42] consists of 32×32 color images. There are 73,257 images in the training set, 26,032 images in the test set and 531,131 images for extra training. Following the common practice [23,38], we train the model with the training and extra sets and tested the model with the test set. No data augmentation is introduced and we use the same pre-processing scheme as in CIFAR.

ImageNet The ILSVRC-2012 dataset [7] consists of 1.28 million training images and 50k validation images, drawn from 1000 categories. Following [19,20,59], we use the standard data augmentation scheme for the training set: a 224×224 crop is randomly sampled from either the original image or its horizontal flip. For testing, we apply a single crop or ten crops with size 224×224 on the validation set and the top-1 or top-5 classification error is reported.

4.2 Training

Both PCN and (plain) CNN models are trained with stochastic gradient descent (SGD). On CIFAR and SVHN datasets, we use 128 batch-size and 0.01 initial learning rate for 300 and 40 epochs, respectively. The learning rate is divided by 10 at 50%, 75% and 87.5% of the total number of training epochs. Besides, we use a Nesterov momentum of 0.9 and a weight decay of 1e-3, which is determined by a 45k/5k split on the CIFAR-100 training set. On ImageNet, we follow the most common practice [19,66] and use a initial learning rate of 0.01, a momentum of 0.9, a weight decay
### Table 2: Error rates (%) on CIFAR datasets. #L and #P are the number of layers and parameters, respectively.

<table>
<thead>
<tr>
<th>Type</th>
<th>Model</th>
<th>#L</th>
<th>#P</th>
<th>C-10</th>
<th>C-100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feed-Forward</td>
<td>HighwayNet [56]</td>
<td>19</td>
<td>2.3M</td>
<td>7.72</td>
<td>32.39</td>
</tr>
<tr>
<td></td>
<td>FractalNet [13]</td>
<td>21</td>
<td>38.6M</td>
<td>5.22</td>
<td>23.30</td>
</tr>
<tr>
<td></td>
<td>ResNet [27]</td>
<td>110</td>
<td>1.7M</td>
<td>0.91</td>
<td>27.22</td>
</tr>
<tr>
<td></td>
<td>ResNet [60] (Pre-act)</td>
<td>164</td>
<td>1.9M</td>
<td>5.23</td>
<td>24.58</td>
</tr>
<tr>
<td></td>
<td>WRN [69]</td>
<td>28</td>
<td>36.5M</td>
<td>4.00</td>
<td>19.25</td>
</tr>
<tr>
<td></td>
<td>DenseNet [35]</td>
<td>100</td>
<td>0.8M</td>
<td>4.91</td>
<td>22.17</td>
</tr>
<tr>
<td></td>
<td>BC [23]</td>
<td>190</td>
<td>25.6M</td>
<td>3.46</td>
<td>17.18</td>
</tr>
<tr>
<td>Recurrent</td>
<td>RCNN [66]</td>
<td>160</td>
<td>1.68M</td>
<td>7.09</td>
<td>31.75</td>
</tr>
<tr>
<td>Models</td>
<td>DasNet [55]</td>
<td>-</td>
<td>-</td>
<td>9.22</td>
<td>33.78</td>
</tr>
<tr>
<td></td>
<td>CliqueNet [56]</td>
<td>18</td>
<td>10.14M</td>
<td>5.10</td>
<td>22.14</td>
</tr>
<tr>
<td></td>
<td>- [63]</td>
<td>30</td>
<td>10.02M</td>
<td>5.06</td>
<td>21.83</td>
</tr>
<tr>
<td></td>
<td>PCN with Global Recurrent Processing [63]</td>
<td>9</td>
<td>4.91M</td>
<td>5.70</td>
<td>24.01</td>
</tr>
<tr>
<td></td>
<td>PCN-C-1</td>
<td>9</td>
<td>4.91M</td>
<td>5.70</td>
<td>24.01</td>
</tr>
<tr>
<td></td>
<td>PCN-C-2</td>
<td>9</td>
<td>4.91M</td>
<td>5.70</td>
<td>24.01</td>
</tr>
<tr>
<td></td>
<td>PCN-C-5</td>
<td>9</td>
<td>4.91M</td>
<td>5.70</td>
<td>24.01</td>
</tr>
<tr>
<td></td>
<td>PCN-D-1</td>
<td>9</td>
<td>9.90M</td>
<td>5.73</td>
<td>23.78</td>
</tr>
<tr>
<td></td>
<td>PCN-D-2</td>
<td>9</td>
<td>9.90M</td>
<td>5.73</td>
<td>23.78</td>
</tr>
<tr>
<td></td>
<td>PCN-D-5</td>
<td>9</td>
<td>9.90M</td>
<td>5.73</td>
<td>23.78</td>
</tr>
<tr>
<td>Plain Models</td>
<td>Plain-C</td>
<td>9</td>
<td>2.59M</td>
<td>5.68</td>
<td>25.85</td>
</tr>
<tr>
<td></td>
<td>Plain-D</td>
<td>9</td>
<td>5.21M</td>
<td>5.61</td>
<td>25.31</td>
</tr>
</tbody>
</table>

### Table 3: Error rates (%) on SVHN. #L and #P are the number of layers and parameters, respectively.

<table>
<thead>
<tr>
<th>Model</th>
<th>#L</th>
<th>#P</th>
<th>SVHN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxOut [45]</td>
<td>-</td>
<td>-</td>
<td>2.47</td>
</tr>
<tr>
<td>NIN [28]</td>
<td>-</td>
<td>-</td>
<td>2.35</td>
</tr>
<tr>
<td>DropConnect [61]</td>
<td>-</td>
<td>-</td>
<td>1.94</td>
</tr>
<tr>
<td>DSN [35]</td>
<td>-</td>
<td>-</td>
<td>1.92</td>
</tr>
<tr>
<td>RCNN [58]</td>
<td>6</td>
<td>2.6M</td>
<td>1.77</td>
</tr>
<tr>
<td>FitNet [43]</td>
<td>13</td>
<td>1.5M</td>
<td>2.42</td>
</tr>
<tr>
<td>WRN [69]</td>
<td>16</td>
<td>11M</td>
<td>1.54</td>
</tr>
<tr>
<td>PCN (Global)</td>
<td>7</td>
<td>0.14M</td>
<td>2.42</td>
</tr>
<tr>
<td>PCN-A [63]</td>
<td>7</td>
<td>0.15M</td>
<td>2.29</td>
</tr>
<tr>
<td>PCN-A-2</td>
<td>7</td>
<td>0.15M</td>
<td>2.22</td>
</tr>
<tr>
<td>PCN-A-5</td>
<td>7</td>
<td>0.15M</td>
<td>2.07</td>
</tr>
<tr>
<td>PCN-B [63]</td>
<td>7</td>
<td>0.01M</td>
<td>1.99</td>
</tr>
<tr>
<td>PCN-B-2</td>
<td>7</td>
<td>0.61M</td>
<td>1.97</td>
</tr>
<tr>
<td>PCN-B-5</td>
<td>7</td>
<td>0.61M</td>
<td>1.96</td>
</tr>
<tr>
<td>Plain-A</td>
<td>7</td>
<td>0.88M</td>
<td>2.85</td>
</tr>
<tr>
<td>Plain-B</td>
<td>7</td>
<td>0.32M</td>
<td>2.43</td>
</tr>
</tbody>
</table>

of 1e-4, 100 epochs with the learning rate dropped by 0.1 at epochs 30, 60 & 90. The batch size is 256 for PCN-E-0/1/2, 128 for PCN-E-3/4, 115 for PCN-E-5 due to limited computational resource.

### 4.3 Evaluating the Behavior of PCN

To understand how the model works, we further examine how local recurrent processing changes the internal representations of PCN. For this purpose, we focus on testing PCN-D-5 with CIFAR-100.

**Converging representation?** Since local recurrent processing is governed by predictive coding, it is anticipated that the error of prediction tends to decrease over time. To confirm this expectation, the L2 norm of the prediction error is calculated for each layer and each cycle of recurrent processing and averaged across all testing examples in CIFAR-100. This analysis reveals the temporal behavior of recurrently refined internal representations.

**What does the prediction error mean?** Since the error of prediction drives the recurrent update of internal representations, we examine the spatial distribution of the error signal (after the final cycle) first for each layer and then average the error distributions across all layers by rescaling them to the same size. The resulting error distribution is used as a spatial pattern in the input space, and it is applied to the input image as a weighted mask to visualize its selectivity.

**Does predictive coding help image classification?** The goal of predictive coding is to reduce the error of top-down prediction, seemingly independent of the objective of categorization that the PCN model is trained for. As recurrent processing progressively updates layer-wise representation, does each update also subserve the purpose of categorization? As in [27], the loss of categorization, as a nonlinear function of layer-wise representation $L(r_t(t))$, shows its Taylor’s expansion as Eq. (9),

$$L(r_{t}(t+1)) - L(r_{t}(t)) = \Delta r_{t}(t) \cdot \frac{\partial L(r_{t}(t))}{\partial r_{t}(t)} + O(\Delta r_{t}(t)^2)$$  

If each update tends to reduce the categorization loss, it should satisfy $\Delta r_{t}(t) \cdot \frac{\partial L(r_{t}(t))}{\partial r_{t}(t)} < 0$, or the "cosine distance" between $\Delta r_{t}(t)$ and $\frac{\partial L(r_{t}(t))}{\partial r_{t}(t)}$ should be negative. Here $O(\cdot)$ is ignored as suggested by [27], since the incremental part becomes minor because of the convergent representation in PCN. We test this by calculating the cosine distance for each cycle of recurrent processing in each layer, and then averaging the results across all testing images in CIFAR-100.
Table 4: Error rates (%) on ImageNet.

<table>
<thead>
<tr>
<th>Model</th>
<th>#Layers</th>
<th>#Params</th>
<th>Single-Crop</th>
<th>10-crop</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>top-1</td>
<td>top-5</td>
</tr>
<tr>
<td>ResNet-34</td>
<td>34</td>
<td>21.80M</td>
<td>26.69</td>
<td>7.14</td>
</tr>
<tr>
<td>ResNet-50</td>
<td>50</td>
<td>25.56M</td>
<td>23.87</td>
<td>6.24</td>
</tr>
<tr>
<td>PCN-E-5</td>
<td>13</td>
<td>17.26M</td>
<td>25.31</td>
<td>7.79</td>
</tr>
<tr>
<td>PCN-E-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plain-E</td>
<td>13</td>
<td>9.34M</td>
<td>31.15</td>
<td>11.27</td>
</tr>
</tbody>
</table>

Figure 2: PCN shows better categorization performance given more cycles of recurrent processing, for CIFAR-10, CIFAR-100 and ImageNet. The red dash line represents the accuracy of the plain model.

4.4 Experimental results

**Classification performance** On CIFAR and SVHN datasets, PCN always outperforms its corresponding CNN counterpart (Table 2 and 3). On CIFAR-100, PCN-D-5 reduces the error rate by 3.54% relative to Plain-D. The performance of PCN is also better than those of the ResNet family [19, 20], although PCN is much shallower with only 9 layers whereas ResNet may use as many as 1001 layers. Although PCN under-performs WRN [69] and DenseNet-190 [24] by 2-4%, it uses a much shallower architecture with many fewer parameters. On SVHN, PCN shows competitive performance despite fewer layers and parameters. On ImageNet, PCN also performs better than its plain counterpart. With 5 cycles of local recurrent processing (i.e., PCN-E-5), PCN slightly under-performs ResNet-50 but outperforms ResNet-34, while using fewer layers and parameters than both of them. Therefore, the classification performance of PCN compares favorably with other state-of-the-art models especially in terms of the performance-to-layer ratio.

**Recurrent Cycles** The classification performance generally improves as the number of cycles of local recurrent processing increases for both CIFAR and ImageNet datasets, and especially for ImageNet (Fig. 2). It is worth noting that this gain in performance is achieved without increasing the number of layers or parameters, but by simply running computation for longer time on the same network.

**Local vs. Global Recurrent Processing** The PCN with local recurrent processing (proposed in this paper) performs better than the PCN with global recurrent processing (proposed in our earlier paper [63]. Table 2 shows that PCN with local recurrent processing reduces the error rate by 5% on CIFAR-100 compared to PCN with the same architecture but global recurrent processing. In addition, local recurrent processing also requires less computational resource for both training and inference.

4.5 Behavioral Analysis

To understand how/why PCN works, we further examine how local recurrent processing changes its internal representations and whether the change helps categorization. Behavioral analysis reveals some intriguing findings.

As expected for predictive coding, local recurrent processing progressively reduces the error of top-down prediction for all layers, except the top layer on which image classification is based (Fig.
Figure 3: Behavioral analysis of PCN. (a) Errors of prediction tend to converge over repeated cycles of recurrent processing. The norm of the prediction error is shown for each layer and each cycle of local recurrent processing. (b) Errors of prediction reveal visual saliency. Given an input image (left), the spatial distribution of the error averaged across layers (middle) shows visual saliency or bottom-up attention, highlighting the part of the input with defining features (right). (3) The averaged cosine loss between $\Delta r_l(t)$ and $\frac{\partial L(r_l(t))}{\partial r_l(t)}$.

This finding implies that the internal representations converge to a stable state by local recurrent processing. Interestingly, the spatial distribution of the prediction error (averaged across all layers) highlights the apparently most salient part of the input image, and/or the most discriminative visual information for object recognition (Fig. 3b). The recurrent update of layer-wise representation tends to align along the negative gradient of the categorization loss with respective to the corresponding representation (Fig. 3c). From a different perspective, this finding lends support to an intriguing implication that predictive coding facilitates object recognition, which is somewhat surprising because predictive coding, as a computational principle, herein only explicitly reduces the error of top-down prediction without having any explicit role that favors inference or learning towards object categorization.

5 Discussion and Conclusion

In this study, we advocate further synergy between neuroscience and artificial intelligence. Complementary to engineering innovation in computing and optimization, the brain must possess additional mechanisms to enable generalizable, continuous, and efficient learning and inference. Here, we highlight the fact that the brain runs recurrent processing with lateral and feedback connections under
predictive coding, instead of feedforward-only processing. While our focus is on object recognition, the PCN architecture can potentially be generalized to other computer vision tasks (e.g. object detection, semantic segmentation and image caption) [13, 22, 40], or subserve new computational models that can encode [62, 16, 29, 49] or decode [46, 17, 48, 57] brain activities. PCN with local recurrent processing outperforms its counterpart with global recurrent processing, which is not surprising because global feedback pathways might be necessary for top-down attention [4, 3], but may not be necessary for core object recognition [9] itself. By modeling different mechanisms, we support the notion that local recurrent processing is necessary for the initial feedforward process for object recognition.

This study leads us to rethink about the models for classification beyond feedforward-only networks. One interesting idea is to evaluate the equivalence between ResNets and recurrent neural networks (RNN). Deep residual networks with shared weights can be strictly reformulated as a shallow RNN [37]. Regular ResNets can be reformulated as time-variant RNNs [37], and their representations are iteratively refined along the stacked residual blocks [27]. Similarly, DenseNet has been shown as a generalized form of higher order recurrent neural network (HORN) [6]. The results in this study are in line with such notions: a dynamic and bi-directional network can refine its representations across time, leading to convergent representations to support object recognition.

On the other hand, PCN is not contradictory to existing feedforward models, because the PCN block itself is integrated with an Inception-type CNN module. We expect that other network modules are applicable to further improve PCN performance, including cutout regularization [8], dropout layer [21], residual learning [19] and dense connectivities [24]. Although not explicitly trained, error signals of PCN can be used to predict saliency in images, suggesting that other computer vision tasks [13, 22, 40] could benefit from the diverse feature representations (e.g. error, prediction and state signals) in PCN.

The PCN with local recurrent processing described herein has the following advantages over feedforward CNNs or other dynamic or recurrent models, including a similar PCN with global recurrent processing [63]. 1) It can achieve competitive performance in image classification with a shallow network and fewer parameters. 2) Its internal representations converge as recurrent processing proceeds over time, suggesting a self-organized mechanism towards stability [12]. 3) It reveals visual saliency or bottom-up attention while performing object recognition. However, its disadvantages are 1) the longer time of computation than plain networks (with the same number of layers) and 2) the sequentially executed recurrent processing, both of which should be improved or addressed in future studies.

Acknowledgement

The research was supported by NIH R01MH104402 and the College of Engineering at Purdue University.

References


