PR Product: A Substitute for Inner Product in Neural Networks

Zhennan Wang†, Wenbin Zou†, Chen Xu*
Shenzhen University
wangzhennan2017@email.szu.edu.cn, {wzou, xuchen_szu}@szu.edu.cn

Abstract

In this paper, we analyze the inner product of weight vector $\mathbf{w}$ *and data vector* $\mathbf{x}$ *in neural networks from the perspective of vector orthogonal decomposition and prove that the direction gradient of* $\mathbf{w}$ *decreases with the angle between them close to 0 or* π*. We propose the Projection and Rejection Product (PR Product) to make the direction gradient of* $\mathbf{w}$ *independent of the angle and consistently larger than the one in standard inner product while keeping the forward propagation identical. As a reliable substitute for standard inner product, the PR Product can be applied into many existing deep learning modules, so we develop the PR Product version of fully connected layer, convolutional layer and LSTM layer. In static image classification, the experiments on CIFAR10 and CIFAR100 datasets demonstrate that the PR Product can robustly enhance the ability of various state-of-the-art classification networks. On the task of image captioning, even without any bells and whistles, our PR Product version of captioning model can compete or outperform the state-of-the-art models on MS COCO dataset. Code has been made available at:*`https://github.com/wzn0828/PR_Product`.

1. Introduction

Models based on neural networks, especially deep convolutional neural networks (CNN) and recurrent neural networks (RNN), have achieved state-of-the-art results in various computer vision tasks [11, 10, 1]. Most of the optimization algorithms for these models rely on gradient-based learning, so it is necessary to analyze the gradient of inner product between weight vector $\mathbf{w} \in R^d$ and data vector $\mathbf{x} \in R^d$, a basic operation in neural networks. Denoted by $P(\mathbf{w}, \mathbf{x}) = \mathbf{w}^T\mathbf{x}$ the inner product, the gradient of $P(\mathbf{w}, \mathbf{x})$ w.r.t. $\mathbf{w}$ is exactly the data vector $\mathbf{x}$ which can be orthogonally decomposed into the vector projection $\mathbf{P_x}$ on $\mathbf{w}$ and the vector rejection $\mathbf{R_x}$ from $\mathbf{w}$, as shown in Figure 1 (a). The vector projection $\mathbf{P_x}$ is parallel to the weight vector $\mathbf{w}$ and will update the length of $\mathbf{w}$ in next training iteration, called the length gradient. While the vector rejection $\mathbf{R_x}$ is orthogonal to $\mathbf{w}$, it will change the direction of $\mathbf{w}$, called the direction gradient.

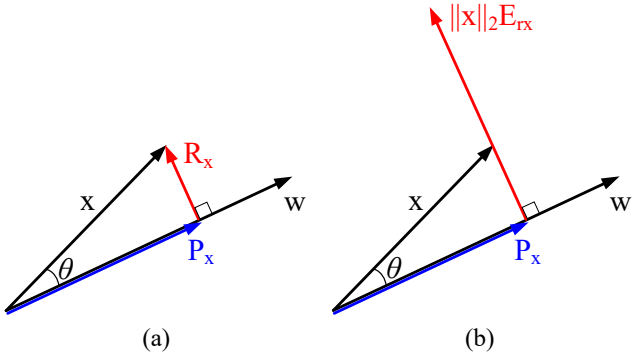

Figure 1. The orthogonal decomposition of the gradient w.r.t. weight vector $\mathbf{w}$ in two-dimensional space. (a) The case of the standard inner product. (b) The case of our proposed PR Product. For the length gradient, both are the vector projection $\mathbf{P_x}$ of $\mathbf{x}$ onto $\mathbf{w}$. However, the direction gradient is changed from the vector rejection $\mathbf{R_x}$ in (a) to $\|\mathbf{x}\|_2\mathbf{E_{rx}}$ in (b), where $\mathbf{E_{rx}}$ represents the unit vector along $\mathbf{R_x}$.

Driven by the orthogonal decomposition of the gradient w.r.t. $\mathbf{w}$, a question arises: Which is the key factor for optimization, length gradient or direction gradient? To answer this question, we optimize three 5-layer fully connected neural networks on Fashion-MNIST [36] with different variants of inner product: standard inner product, a variant without length gradient and a variant without direction gradient. The top-1 accuracy is 88.42%, 88.32% and 38.59%, respectively. From these comparative experiments we can observe that the direction gradient is the key factor for optimization and is far more critical than the length gradient, which might be unsurprising. However, the direction gradient would be very small when $\mathbf{w}$ and $\mathbf{x}$ are nearly parallel, which would hamper the update of the

†The authors are with College of Electronic and Information Engineering, the Shenzhen Key Laboratory of Advanced Machine Learning and Applications, the Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University.
*Corresponding author is with the College of Mathematics and Statistics, Shenzhen University.

direction of weight vector $\mathbf{w}$.

On the other hand, in Euclidean space, the geometric definition of inner product is the product of the Euclidean lengths of the two vectors and the cosine of the angle between them. That is $P(\mathbf{w}, \mathbf{x}) = \mathbf{w}^T\mathbf{x} = \|\mathbf{w}\|_2\|\mathbf{x}\|_2 \cos\theta$, where we denote by $\| * \|_2$ the Euclidean length of vector $*$ and by θ the angle between $\mathbf{w}$ and $\mathbf{x}$ with the range of $[0, 2\pi)$. From this formulation, we can see that the θ is strongly connected with the direction of weight vector $\mathbf{w}$. The gradient of P w.r.t. θ is $\partial P/\partial\theta = -\|\mathbf{w}\|_2\|\mathbf{x}\|_2 \sin\theta$, which becomes small with θ close to 0 or π and thus hinders the optimization. Several recent investigations of backpropagation [4, 43] focus on modifying the gradient of activation function. However, few researches propose variants of backpropagation for the inner product function.

In this paper, we propose the Projection and Rejection Product(abbreviated as PR Product) which changes the backpropagation of standard inner product to eliminate the dependence of the direction gradient of $\mathbf{w}$ and the gradient w.r.t. θ on the value of θ. We firstly prove that the standard inner product of $\mathbf{w}$ and $\mathbf{x}$ only contains the information of vector projection $\mathbf{P}_\mathbf{x}$, which is the main cause of the above dependence. While our proposed PR Product involves the information of both the vector projection $\mathbf{P}_\mathbf{x}$ and the vector rejection $\mathbf{R}_\mathbf{x}$ through rewriting the standard inner product into a different form and suitable components of that form are held fixed during backward pass. We further analyze the gradients of PR Product w.r.t. θ and $\mathbf{w}$. For θ, the absolute value of gradient changes from $\|\mathbf{w}\|_2\|\mathbf{x}\|_2 |\sin\theta|$ to $\|\mathbf{w}\|_2\|\mathbf{x}\|_2$. For $\mathbf{w}$, the length of direction gradient changes from $\|\mathbf{x}\|_2 |\sin\theta|$ to $\|\mathbf{x}\|_2$, as shown in Figure 1.

There are several advantages of using PR Product:(a)The PR Product gets a different backward pass while the forward pass remains exactly the same as the standard inner product; (b) Compared with the behavior of standard inner product, PR increases the proportion of the direction gradient which is the key factor for optimization; (c) As the PR Product maintains the linear property, it can be a reliable substitute for inner product operation in the fully connected layer, convolutional layer and recurrent layer. By reliable, we mean it does not introduce any additional parameters and matches with the original configurations such as activation function, batch normalization and dropout operation.

We showcase the effectiveness of PR Product on image classification and image captioning tasks. For both tasks, we replace all the fully connected layers, convolutional layers and recurrent layers of the backbone models with their PR Product version. Experiments on image classification demonstrate that the PR Product can typically improve the accuracy of the state-of-the-art classification models. Moreover, our analysis on image captioning confirms that the PR Product definitely change the dynamics of neural networks. Without any tricks of improving the performance, like scene graph and ensemble strategy, our PR Product version of captioning model achieves results on par with the state-of-the-art models.

In summary, the main contributions of this paper are:

- We propose the PR Product, a reliable substitute for the standard inner product of weight vector $\mathbf{w}$ and data vector $\mathbf{x}$ in neural networks, which changes the backpropagation while keeping the forward propagation identical;
- We develop the PR-FC, PR-CNN and PR-LSTM, which applies the PR Product into the fully connected layer, convolutional layer and LSTM layer respectively;
- Our experiments on image classification and image captioning suggest that the PR Product is generally effective and can become a basic operation of neural networks.

2. Related Work

Variants of Backpropagation. Several recent investigations have considered variants of standard backpropagation. In particular, [22] presents a surprisingly simple backpropagation mechanism that assigns blame by multiplying errors signals with random weights, instead of the synaptic weights on each neuron, and further downstream. [2] exhaustively considers many Hebbian learning algorithms. The straight-through estimator proposed in [4] heuristically copies the gradient with respect to the stochastic output directly as an estimator of the gradient with respect to the sigmoid argument. [43] proposes Linear Backprop that backpropagates error terms only linearly. Different from these methods, our proposed PR Product changes the gradient of inner product function during backpropagation while maintaining the identical forward propagation.

Image Classification. Deep convolutional neural networks [18, 32, 11, 12, 45, 37, 13] have become the dominant machine learning approaches for image classification. To train very deep networks, shortcut connections have become an essential part of modern networks. For example, Highway Networks [33, 34] present shortcut connections with gating functions, while variants of ResNet [11, 12, 45, 37] use identity shortcut connections. DenseNet [13], a more recent network with several parallel shortcut connections, connects each layer to every other layer in a feed-forward fashion.

Image Captioning. In the early stage of vision to language field, template-based methods [7, 20] generate the caption templates whose slots are filled in by the outputs of object detection, attribute classification and scene recognition, which results in captions that sound unnatural. Recently, inspired by the advances in the NLP field, models based

encoder-decoder architecture [17, 16, 35, 14] have achieved striking advances. These approaches typically use a pre-trained CNN model as the image encoder, combined with an RNN decoder trained to predict the probability distribution over a set of possible words. To better incorporate the image information into the language processing, visual attention for image captioning was first introduced by [38] which allows the decoder to automatically focus on the image subregions that are important for the current time step. Because of remarkable improvement of performance, many extensions of visual attention mechanism [44, 5, 40, 9, 27, 1] have been proposed to push the limits of this framework for caption generation tasks. Except for those extensions to visual attention mechanism, several attempts [31, 26] have been made to adapt reinforcement learning to address the discrepancy between the training and the testing objectives for image captioning. More recently, some methods [41, 15, 25, 39] exploit scene graph to incorporate visual relationship knowledge into captioning models for better descriptive abilities.

3. The Projection and Rejection Product

In this section, we begin by shortly revisiting the standard inner product of weight vector $\mathbf{w}$ and data vector $\mathbf{x}$. Then we formally propose the Projection and Rejection Product (PR Product) which involves the information of both vector projection of $\mathbf{x}$ onto $\mathbf{w}$ and vector rejection of $\mathbf{x}$ from $\mathbf{w}$. Moreover, we analyze the gradient of PR Product. Finally, we develop the PR Product version of fully connected layer, convolutional layer and LSTM layer. In the following, for the simplicity of derivation, we only consider a single data vector $\mathbf{x} \in R^d$ and a single weight vector $\mathbf{w} \in R^d$ except for the last subsection.

3.1. Revisit the Inner Product in Neural Networks

In Euclidean space, the inner product P of the two Euclidean vectors $\mathbf{w}$ and $\mathbf{x}$ is defined by:

$$P(\mathbf{w}, \mathbf{x}) = \mathbf{w}^T\mathbf{x} = \|\mathbf{w}\|_2\|\mathbf{x}\|_2 \cos\theta \tag{1}$$

where $\| * \|_2$ is the Euclidean length of vector $*$, and θ is the angle between $\mathbf{w}$ and $\mathbf{x}$ with the range of $[0, 2\pi)$. From this formulation, we can observe that the angle θ explicitly affects the state of neural networks.

The gradient of P w.r.t. θ. Neither the weight vector $\mathbf{w}$ nor the data vector $\mathbf{x}$ is the function of θ, so it is easy to get:

$$\frac{\partial P}{\partial \theta} = -\|\mathbf{w}\|_2\|\mathbf{x}\|_2 \sin\theta \tag{2}$$

The gradient of P w.r.t. $\mathbf{w}$. From Equation (1) and Figure 1 (a), it is easy to obtain the gradient function of P w.r.t. $\mathbf{w}$:

$$\frac{\partial P}{\partial \mathbf{w}} = \mathbf{x} = \mathbf{P_x} + \mathbf{R_x} \tag{3}$$

Here, $\mathbf{R_x}$ is the direction gradient of $\mathbf{w}$. From Figure 1 (a) and Equation (2), we can see that either the value of the gradient of P w.r.t. θ or the length of $\mathbf{R_x}$ is close to 0 with θ close to 0 or π, which would hamper the optimization of neural networks.

From Figure 1 (a),we can easily get the length of $\mathbf{P_x}$:

$$\|\mathbf{P_x}\|_2 = \|\mathbf{x}\|_2 \left|\cos\theta\right| \tag{4}$$

And the length of $\mathbf{R_x}$ is:

$$\|\mathbf{R_x}\|_2 = \|\mathbf{x}\|_2 \left|\sin\theta\right| \tag{5}$$

So equation (1) can be reformulated as:

$$\begin{aligned} P(\mathbf{w}, \mathbf{x}) &= \begin{cases} -\|\mathbf{w}\|_2\|\mathbf{P_x}\|_2, & \text{if } \pi/2 \le \theta < 3\pi/2; \\ \|\mathbf{w}\|_2\|\mathbf{P_x}\|_2, & \text{otherwise.} \end{cases} \\ &= sign(\cos\theta)\|\mathbf{w}\|_2\|\mathbf{P_x}\|_2 \end{aligned} \tag{6}$$

where sign(*) denotes the sign of *. We can observe that this formulation only contains the information of vector projection of $\mathbf{x}$ on $\mathbf{w}$, $\mathbf{P_x}$. As shown in Figure 1, the vector projection $\mathbf{P_x}$ changes very little when θ is near 0 or π, which may be a block to the optimization of neural networks. Although the length of the rejection vector $\mathbf{R_x}$ is small when θ is close to 0 or π, it varies greatly and thus is able to support the optimization of neural networks. That is our basic motivation for the proposed PR Product.

3.2. The PR Product

In order to take advantage of the vector rejection, the simplest way is to replace the $\|\mathbf{P_x}\|_2$ in Equation (6) with $\|\mathbf{P_x}\|_2 + \|\mathbf{R_x}\|_2$. But the trends of $\|\mathbf{P_x}\|_2$ and $\|\mathbf{R_x}\|_2$ with θ are inconsistent, so we employ $\|\mathbf{x}\|_2 - \|\mathbf{R_x}\|_2$ to involve the information of vector rejection. In addition, we utilize two coefficients to maintain the linear property, which are held fixed during the backward pass. To be more detailed, we derive the PR Product as follows:

$$\begin{aligned} &PR(\mathbf{w}, \mathbf{x}) \\ =&sign(\cos\theta)\|\mathbf{w}\|_2 \left[\frac{\|\mathbf{R_x}\|_2}{\underline{\|\mathbf{x}\|_2}}\|\mathbf{P_x}\|_2 + \frac{\|\mathbf{P_x}\|_2}{\underline{\|\mathbf{x}\|_2}}(\|\mathbf{x}\|_2 - \|\mathbf{R_x}\|_2)\right] \\ =&\|\mathbf{w}\|_2 \left[\underline{|\sin\theta|}\|\mathbf{P_x}\|_2 sign(\cos\theta) + \underline{\cos\theta}(\|\mathbf{x}\|_2 - \|\mathbf{R_x}\|_2)\right] \\ =&\|\mathbf{w}\|_2\|\mathbf{x}\|_2 \left[\underline{|\sin\theta|}\cos\theta + \underline{\cos\theta}(1 - |\sin\theta|)\right] \end{aligned} \tag{7}$$

For clarity, we denote by PR the proposed product function. Note that the $\underline{*}$ denotes detaching * from neural networks. By detaching, we mean * is considered as a constant rather than a variable during backward propagation. Compared with the standard inner product formulation (Equation (6) or (1)), this formulation involves not only the information of vector projection $\mathbf{P_x}$ but also the one of vector rejection $\mathbf{R_x}$ without any additional parameters. We

call this formulation the Projection and Rejection Product or PR Product for brevity.

Although the PR Product does not change the outcome during forward pass, compared with the standard inner product, it changes the gradients during backward pass. In the following, we theoretically derive the gradient of PR w.r.t. θ and $\mathbf{w}$ during backpropagation.

The gradient of PR w.r.t. θ. We just need to calculate the gradients of trigonometric functions except for the detached ones in Equation (7). When θ is in the range of $[0, \pi)$, the gradient of PR w.r.t. θ is:

$$\begin{aligned}\frac{\partial PR}{\partial \theta} &= \|\mathbf{w}\|_2\|\mathbf{x}\|_2 \left(-\sin^2\theta - \cos^2\theta\right) \\ &= -\|\mathbf{w}\|_2\|\mathbf{x}\|_2\end{aligned} \tag{8}$$

When θ is in the range of $[\pi, 2\pi)$, the gradient of PR w.r.t. θ is:

$$\begin{aligned}\frac{\partial PR}{\partial \theta} &= \|\mathbf{w}\|_2\|\mathbf{x}\|_2 \left(\sin^2\theta + \cos^2\theta\right) \\ &= \|\mathbf{w}\|_2\|\mathbf{x}\|_2\end{aligned} \tag{9}$$

We use the following unified form to express the above two cases:

$$\frac{\partial PR}{\partial \theta} = \|\mathbf{w}\|_2\|\mathbf{x}\|_2 sign\left(-\sin\theta\right) \tag{10}$$

Compared with the standard inner product (Equation (2)), the PR Product changes the gradient w.r.t. θ from a smoothing function to a hard one. One advantage of this is the gradient w.r.t. θ does not decrease as θ gets close to 0 or π, providing continuous power for the optimization of neural networks.

The gradient of PR w.r.t. $\mathbf{w}$. Above we discussed the gradient w.r.t. θ, an implicit variable in neural networks. In this part, we explicitly take a look at the differences between the gradients of the standard inner product and our proposed PR Product w.r.t. $\mathbf{w}$.

For the PR Product, we derive the gradient of PR w.r.t. $\mathbf{w}$ from Equation (7) and Equation (10) as follows :

$$\begin{aligned}&\frac{\partial PR}{\partial \mathbf{w}} \\ =&\frac{\mathbf{w}}{\|\mathbf{w}\|_2}\|\mathbf{x}\|_2\cos\theta + \|\mathbf{w}\|_2\|\mathbf{x}\|_2 sign(-\sin\theta)\frac{\partial\theta}{\partial \mathbf{w}} \\ =&\mathbf{P_x} + \|\mathbf{w}\|_2\|\mathbf{x}\|_2 sign(-\sin\theta)\frac{\mathrm{d}\theta}{\mathrm{d}\cos\theta}\frac{\partial\cos\theta}{\partial \mathbf{w}} \\ =&\mathbf{P_x} + \frac{\|\mathbf{w}\|_2\|\mathbf{x}\|_2}{|\sin\theta|}\frac{\partial\left(\frac{\mathbf{w}^T\mathbf{x}}{\|\mathbf{w}\|_2\|\mathbf{x}\|_2}\right)}{\partial \mathbf{w}} \\ =&\mathbf{P_x} + \frac{\|\mathbf{w}\|_2\|\mathbf{x}\|_2}{|\sin\theta|}\frac{(\boldsymbol{I} - \boldsymbol{M_w})\mathbf{x}}{\|\mathbf{w}\|_2\|\mathbf{x}\|_2} \\ =&\mathbf{P_x} + \frac{\mathbf{R_x}}{|\sin\theta|} \\ =&\mathbf{P_x} + \|\mathbf{x}\|_2\frac{\mathbf{R_x}}{\|\mathbf{R_x}\|_2} \\ =&\mathbf{P_x} + \|\mathbf{x}\|_2\mathbf{E_{rx}}, \qquad with \quad \boldsymbol{M_w} = \frac{\mathbf{w}\mathbf{w}^T}{\|\mathbf{w}\|_2^2}\end{aligned} \tag{11}$$

Where $\boldsymbol{M_w}$ is the projection matrix that projects onto the weight vector $\mathbf{w}$, which means $\boldsymbol{M_w}\mathbf{x} = \mathbf{P_x}$, and $\mathbf{E_{rx}}$ is the unit vector along the vector rejection $\mathbf{R_x}$. Similar to Equation (3), the $\mathbf{P_x}$ is the length gradient part and the $\|\mathbf{x}\|_2\mathbf{E_{rx}}$ is the direction gradient part. For the length gradient, the cases in P and PR are identical. For the direction gradient part, however, the one in PR is consistently larger than the one in P, except for the almost impossible cases when θ is equal to $\pi/2$ or $3\pi/2$. So PR increases the proportion of the direction gradient. In addition, the length of direction gradient in PR is independent of the value of θ. Figure 1 shows the comparison of the gradients of the two formulations w.r.t. $\mathbf{w}$.

3.3. PR-X

The PR Product is a reliable substitute for the standard inner product operation, so it can be applied into many existing deep learning modules, such as fully connected layer(FC), convolutional layer(CNN) and LSTM layer. We denote the module X with PR Product by PR-X. In this section, we show the implementation of PR-FC, PR-CNN and PR-LSTM.

PR-FC. To get PR-FC, we just replace the inner product of the data vector $\mathbf{x}$ and each weight vector in the weight matrix with the PR Product. Suppose the weight matrix $\boldsymbol{W}$ contains a set of n column vectors, $\boldsymbol{W} = (\mathbf{w_1}, \mathbf{w_2}, ..., \mathbf{w_n})$, so the output vector of PR-FC can be calculated as follows:

$$\begin{aligned}&PR\text{-}FC(\boldsymbol{W}, \mathbf{x}) \\ =&(PR(\mathbf{w_1}, \mathbf{x}), PR(\mathbf{w_2}, \mathbf{x}), ..., PR(\mathbf{w_n}, \mathbf{x})) + \mathbf{b}\end{aligned} \tag{12}$$

where $\mathbf{b}$ represents an additive bias vector if any.

PR-CNN. To apply the PR Product into CNN, we convert the weight tensor of the convolutional kernel and the data

Model	CIFAR10	CIFAR100
ResNet110	6.23	28.08
PR-ResNet110	**5.97**	**27.88**
PreResNet110	5.99	27.08
PR-PreResNet110	**5.64**	**26.82**
WRN-28-10	4.34	**19.50**
PR-WRN-28-10	**4.03**	19.57
DenseNet-BC-100-12	4.63	22.88
PR-DenseNet-BC-100-12	**4.46**	**22.64**

Table 1. Error rates on CIFAR10 and CIFAR100. The best results are highlighted in bold for the models with the same backbone architectures. All values are reported in percentage. The PR Product version can typically outperform the corresponding backbone models.

tensor in the sliding window into vectors in Euclidean space, and then use the PR Product to calculate the output. Suppose the size of the convolution kernel $\mathbf{w}$ is (k_1, k_2, C_{in}), so the output at position (i, j) is:

$$\begin{aligned} &PR\text{-}CNN(\mathbf{w}, \mathbf{x})_{ij} \\ =&PR(flatten(\mathbf{w}), flatten(\mathbf{x}_{[\mathbf{ij}]})) + b \end{aligned} \tag{13}$$

where $flatten(\mathbf{w})$ and $flatten(\mathbf{x}_{[\mathbf{ij}]}) \in R^{k_1 * k_2 * C_{in}}$, $\mathbf{x}_{[\mathbf{ij}]}$ represents the data tensor in the sliding window corresponding to output position (i,j), and b represents an additive bias if any.

PR-LSTM. To get the PR Product version of LSTM, just replace all the perceptrons in each gate function with the PR-FC. For each element in input sequence, each layer computes the following function:

$$\begin{aligned} \mathbf{i_t} &= \sigma\left(PR\text{-}FC(\boldsymbol{W}_{ii}, \mathbf{x_t}) + PR\text{-}FC(\boldsymbol{W}_{hi}, \mathbf{h_{(t-1)}}) + \mathbf{b_i}\right) \\ \mathbf{f_t} &= \sigma\left(PR\text{-}FC(\boldsymbol{W}_{if}, \mathbf{x_t}) + PR\text{-}FC(\boldsymbol{W}_{hf}, \mathbf{h_{(t-1)}}) + \mathbf{b_f}\right) \\ \mathbf{g_t} &= \tanh\left(PR\text{-}FC(\boldsymbol{W}_{ig}, \mathbf{x_t}) + PR\text{-}FC(\boldsymbol{W}_{hg}, \mathbf{h_{(t-1)}}) + \mathbf{b_g}\right) \\ \mathbf{o_t} &= \sigma\left(PR\text{-}FC(\boldsymbol{W}_{io}, \mathbf{x_t}) + PR\text{-}FC(\boldsymbol{W}_{ho}, \mathbf{h_{(t-1)}}) + \mathbf{b_o}\right) \\ \mathbf{c_t} &= \mathbf{f_t} * \mathbf{c_{(t-1)}} + \mathbf{i_t} * \mathbf{g_t} \\ \mathbf{h_t} &= \mathbf{o_t} * \tanh(\mathbf{c_t}) \end{aligned} \tag{14}$$

where σ is the sigmoid function, and * is the Hadamard product.

In the following, we conduct experiments on image classification to validate the effectiveness of PR-CNN. And then we show the effectiveness of PR-FC and PR-LSTM on image captioning task.

4. Experiments on Image Classification

4.1. Classification Models

We employ various classic networks such as ResNet [11], PreResNet [12], WideResNet [45] and DenseNet-BC [13] as the backbone networks in our experiments. In particular, we consider ResNet with 110 layers denoted by ResNet110, PreResNet with 110 layers denoted by PreResNet110, and WideResNet with 28 layers and a widen factor of 10 denoted by WRN-28-10, as well as DenseNet-BC with 100 layers and a growth rate of 12 denoted by DenseNet-BC-100-12. For ResNet110 and PreResNet110, we use the classic basic block. To get the corresponding PR Product version models, all the fully connected layers and the convolutional layers in the backbone models are replaced with our PR-FC and PR-CNN respectively, and we denote them by PR-X, such as PR-ResNet110, PR-PreResNet110, PR-WRN-28-10 and PR-DenseNet-BC-100-12 respectively.

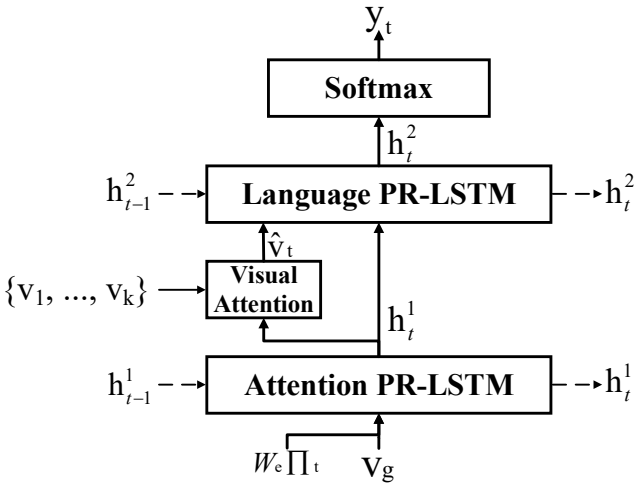

Figure 2. Decoder module used in our captioning model. The input to the Attention PR-LSTM consists of the global image representation $\mathbf{v_g}$ and the embedding of the previously generated word $\boldsymbol{W}_e\boldsymbol{\Pi}_\mathbf{t}$. The input to the Language PR-LSTM consists of the attended image representation $\hat{\mathbf{v}}_\mathbf{t}$ concatenated with the output of the Attention PR-LSTM. The dotted arrows represent the transfer of the hidden states of PR-LSTM layers.

4.2. Dataset and Settings

We conduct our image classification experiments on the CIFAR dataset [19], which consists of 50k and 10k images of 32×32 pixels for the training and test sets respectively. The images are labeled with 10 and 100 categories, namely CIFAR10 and CIFAR100 datasets. We present experiments trained on the training set and evaluated on the test set. We follow the simple data augmentation in [21] for training: 4 pixels are padded on each side and a 32×32 crop is randomly sampled from the padded image or its horizontal flip. For testing, we only evaluate the single view of the original 32×32 image. Note that our focus is on the effectiveness of our proposed PR Product, not on pushing the state-of-the-art results, so we do not use any more data augmentation and training tricks to improve accuracy.

Product	B1	B2	B3	B4	M	RL	C	S
P	76.7	60.8	47.3	36.8	28.1	56.9	116.0	**21.1**
R	76.3	60.4	46.7	36.0	27.7	56.5	113.3	20.6
PR	**76.8**	**61.0**	**47.5**	**37.0**	**28.2**	**57.1**	**116.1**	**21.1**
P*	80.3	**64.9**	50.4	38.6	28.6	58.4	127.2	**22.4**
R*	80.0	64.6	49.8	37.6	28.3	57.8	125.5	22.0
PR*	**80.4**	**64.9**	**50.5**	**38.7**	**28.8**	**58.5**	**128.3**	**22.4**

Table 2. Performance comparison of different products on the test portion of Karpathy splits on MS COCO dataset, where Bn is short for BLEU-n, M is short for METEOR, RL is short for ROUGE-L, C is short for CIDEr, and S is short for SPICE. The top part is for cross-entropy training, and the bottom part is for CIDEr optimization (marked with *). All values are reported in percentage, with the highest value of each entry highlighted in boldface.

4.3. Results and Analysis

For fair comparison, not only are the PR-X models trained from scratch but also the corresponding backbone models, so our results may be slightly different from the ones presented in the original papers due to some hyper-parameters like random number seeds. The strategies and hyper-parameters used to train the respective backbone models, such as the optimization solver, learning rate schedule, parameter initialization method, random seed for initialization, batch size and weight decay, are adopted to train the corresponding PR-X models. The results are shown in Table 1 and some training curves are shown in the supplementary material, from which we can see that the PR-X can typically improve the corresponding backbone models on both CIFAR10 and CIFAR100. On average, it reduces the top-1 error by 0.27% on CIFAR10 and 0.16% on CIFAR100. It is worth emphasizing that the PR-X models don't introduce any additional parameters and keep the same hyper-parameters as the corresponding backbone models.

5. Experiments on Image Captioning

5.1. Captioning Model

We utilize the widely used encoder-decoder framework [1, 27] as our backbone model for image captioning.

Encoder. We use the Bottom-Up model proposed in [1] to generate the regional representations and the global representation of a given image $\boldsymbol{I}$. The Bottom-Up model employs Faster R-CNN [29] in conjunction with the ResNet-101 [11] to generate a variably-sized set of k representations, $\boldsymbol{A} = \{\mathbf{a_1}, ..., \mathbf{a_k}\}$, $\mathbf{a_i} \in R^{2048}$, such that each representation encodes a salient region of the image. We use the global average pooled image representation $\mathbf{a_g} = \frac{1}{k}\sum_i \mathbf{a_i}$ as our global image representation. For modeling convenience, we use a single layer of PR-FC with rectifier activation function to transform the representation vectors into new vectors with dimension d:

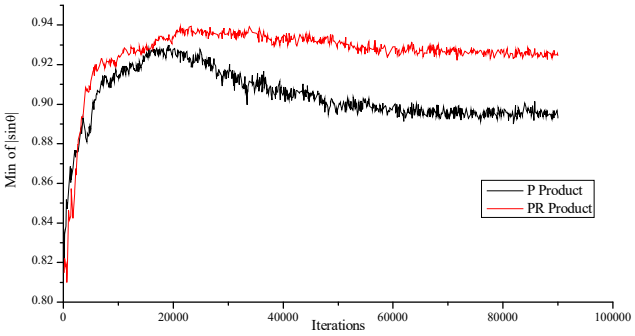

Figure 3. The minimum of $|\sin\theta|$ of the hidden-hidden transfer part in the Attention LSTM.

$$\mathbf{v_i} = ReLU\left(PR\text{-}FC(\boldsymbol{W_a}, \mathbf{a_i})\right), \mathbf{v_i} \in R^d \quad (15)$$

$$\mathbf{v_g} = ReLU\left(PR\text{-}FC(\boldsymbol{W_g}, \mathbf{a_g})\right), \mathbf{v_g} \in R^d \quad (16)$$

where $\boldsymbol{W_a}$ and $\boldsymbol{W_g}$ are the weight parameters. The transformed $\boldsymbol{V} = \{\mathbf{v_1}, ..., \mathbf{v_k}\}$ is our defined regional image representations and $\mathbf{v_g}$ is our defined global image representation.

Decoder. For decoding image representations $\boldsymbol{V}$ and $\boldsymbol{v}_g$ to sentence description, as shown in Figure 2, we utilize an visual attention model with two PR-LSTM layers according to recent methods [1, 28, 41], which are characterized as Attention PR-LSTM and Language PR-LSTM respectively. We initialize the hidden state and memory cell of each PR-LSTM as zero.

Given the output $\mathbf{h_t^1}$ of the Attention PR-LSTM, we generate the attended regional image representation $\hat{\mathbf{v}}_\mathbf{t}$ through the attention model, which is broadly adopted in recent previous work [5, 27, 1]. Here, we use the PR Product version of visual attention model expressed as follows:

$$\begin{aligned}
\mathbf{f_1} &= tanh\left(PR\text{-}FC(\boldsymbol{W_v}, \boldsymbol{V}) + PR\text{-}FC(\boldsymbol{W_{h1}}, \mathbf{h_t^1})\right) \\
\mathbf{f_2} &= PR\text{-}FC(\boldsymbol{W_z}, \mathbf{f_1}) \\
\alpha_\mathbf{t} &= softmax\left(\mathbf{f_2}\right) \\
\hat{\mathbf{v}}_\mathbf{t} &= \sum_{i=1}^{k} \alpha_{t,i}\mathbf{v_i}
\end{aligned} \quad (17)$$

where $\boldsymbol{W_v}$, $\boldsymbol{W_{h1}}$ and $\boldsymbol{W_z}$ are learned parameters, $\mathbf{f_1}$ and $\mathbf{f_2}$ are the outputs of the first layer and the second layer in the attention model respectively. $\alpha_\mathbf{t}$ is the attention weight over k regional image representations, and $\hat{\mathbf{v}}_\mathbf{t}$ is the attended image representation at time step t.

5.2. Dataset and Settings

Dataset. We evaluate our proposed method on the MS COCO dataset [23]. MS COCO dataset contains 123287

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr	SPICE
LSTM-A [42]	73.5	56.6	42.9	32.4	25.5	53.9	99.8	18.5
SCN-LSTM$^{\Sigma}$ [8]	74.1	57.8	44.4	34.1	26.1	-	104.1	-
Adaptive [27]	74.2	58.0	43.9	33.2	26.6	-	108.5	-
SCST:Att2all$^{\Sigma}$ [31]	-	-	-	32.2	26.7	54.8	104.7	-
Up-Down [1]	77.2	-	-	36.2	27.0	56.4	113.5	20.3
Stack-Cap [9]	76.2	60.4	46.4	35.2	26.5	-	109.1	-
ARNet [6]	74.0	57.6	44.0	33.5	26.1	54.6	103.4	19.0
NBT [28]	75.5	-	-	34.7	27.1	-	107.2	20.1
GCN-LSTM$_{sem}$ [41]	**77.3**	-	-	36.8	27.9	57.0	**116.3**	20.9
Ours:PR	76.8	**61.0**	**47.5**	**37.0**	**28.2**	**57.1**	116.1	**21.1**
EmbeddingReward* [30]	71.3	53.9	40.3	30.4	25.1	52.5	93.7	-
LSTM-A* [42]	78.6	-	-	35.5	27.3	56.8	118.3	20.8
SCST:Att2all$^{\Sigma *}$ [31]	-	-	-	35.4	27.1	56.6	117.5	-
Up-Down* [1]	79.8	-	-	36.3	27.7	56.9	120.1	21.4
Stack-Cap* [9]	78.6	62.5	47.9	36.1	27.4	56.9	120.4	20.9
GCN-LSTM$^{*}_{sem}$ [41]	80.5	-	-	38.2	28.5	58.3	127.6	22.0
CAVP* [25]	-	-	-	38.6	28.3	58.5	126.3	21.6
SGAE* [39]	**80.8**	-	-	38.4	28.4	**58.6**	127.8	22.1
Ours:PR*	80.4	**64.9**	**50.5**	**38.7**	**28.8**	58.5	**128.3**	**22.4**

Table 3. Performance compared with the state-of-the-art methods on the Karpathy test split of MS COCO. $^{\Sigma}$ indicates ensemble. The top part is for cross-entropy training, and the bottom part is for REINFORCE-based optimization (marked with *). All values are reported in percentage, with the highest value of each entry highlighted in boldface.

images labeled with at least 5 captions. There are 82783 training images and 40504 validation images, and it provides 40775 images as the test set for online evaluation as well. For offline evaluation, we use a set of 5000 images for validation, a set of 5000 images for test and the remains for training, as given in [16]. We truncate captions longer than 16 words and then build a vocabulary of words that occur at least 5 times in the training set, resulting in 9487 words.

Implementation Details. In the captioning model, we set the number of hidden units in each LSTM or PR-LSTM to 512, the embedding dimension of a word to 512, and the embedding dimension of image representation to 512. All of our models are trained according to the following recipe. We train all models under the cross-entropy loss using ADAM optimizer with an initial learning rate of 5×10^{-4} and a momentum parameter of 0.9. We anneal the learning rate using cosine decay schedule and increase the probability of feeding back a sample of the word posterior by 0.05 every 5 epochs until we reach a feedback probability 0.25 [3]. We then run REINFORCE training to optimize the CIDEr metric using ADAM with a learning rate 5×10^{-5} with cosine decay schedule and a momentum parameter of 0.9. During CIDEr optimization mode and testing mode, we use a beam size of 5. Note that in all our model variants, the untransformed image representations $\boldsymbol{A}$ and $\mathbf{a_g}$ from the Encoder are fixed and not fine-tuned. As our focus is on the effectiveness of our proposed PR Product, so we just exploit the widely used backbone model and settings, without any additional tricks of improving the performance, like scene graph and ensemble strategy.

5.3. Performance Comparison and Experimental Analysis

The effectiveness of PR Product. To test the effectiveness of PR Product, we first compare the performance of models using the following different substitutes for inner product on Karpathy's split of MS COCO dataset:

- **P Product**: This is just the standard inner product. In Euclidean geometry, it is also called projection product, so we abbreviate it as P Product.
- **R Product**: Contrary to P Product, R Product only involves the information of vector rejection of $\mathbf{x}$ from $\mathbf{w}$. To keep the same range and sign as P Product, we formulate the R Product as follows:

$$R(\mathbf{w}, \mathbf{x}) = sign(\cos\theta)\|\mathbf{w}\|_2(\|\mathbf{x}\|_2 - \|\mathbf{R_x}\|_2) \quad (18)$$

- **PR Product**: This is the proposed PR Product. Evidently, the PR Product is the combination of the P Product and R Product with the relationship as follows:

$$PR(\mathbf{w}, \mathbf{x}) = \underline{|\sin\theta|}P(\mathbf{w}, \mathbf{x}) + \underline{|\cos\theta|}R(\mathbf{w}, \mathbf{x}) \quad (19)$$

For fair comparison, results are reported for models trained with cross-entropy loss and models optimized for

	BLEU-1		BLEU-2		BLEU-3		BLEU-4		METEOR		ROUGE-L		CIDEr	
	C5	C40	C5	C40	C5	C40	C5	C40	C5	C40	C5	C40	C5	C40
SCN-LSTM$^{\Sigma}$ [8]	74.0	91.7	57.5	83.9	43.6	73.9	33.1	63.1	25.7	34.8	54.3	69.6	100.3	101.3
Adaptive$^{\Sigma}$ [27]	74.8	92.0	58.4	84.5	44.4	74.4	33.6	63.7	26.4	35.9	55.0	70.5	104.2	105.9
SCST:Att2all$^{*\Sigma}$ [31]	78.1	93.7	61.9	86.0	47.0	75.9	35.2	64.5	27.0	35.5	56.3	70.7	114.7	116.7
Up-Down$^{*\Sigma}$ [1]	**80.2**	**95.2**	64.1	**88.8**	49.1	**79.4**	36.9	**68.5**	27.6	36.7	57.1	72.4	117.9	120.5
LSTM-A [42]	78.7	93.7	62.7	86.7	47.6	76.5	35.6	65.2	27.0	35.4	56.4	70.5	116.0	118.0
PG-BCMR* [26]	75.4	91.8	59.1	84.1	44.5	73.8	33.2	62.4	25.7	34.0	55.0	69.5	101.3	103.2
MAT [24]	73.4	91.1	56.8	83.1	42.7	72.7	32.0	61.7	25.8	34.8	54.0	69.1	102.9	106.4
Stack-Cap* [9]	77.8	93.2	61.6	86.1	46.8	76.0	34.9	64.6	27.0	35.6	56.2	70.6	114.8	118.3
Ours:PR*	79.9	94.5	**64.3**	88.2	**49.6**	79.0	**37.7**	68.3	**28.4**	**37.5**	**58.0**	**73.0**	**122.3**	**124.1**

Table 4. Performance compared with the state-of-the-art methods on the online MS COCO test server. $^{\Sigma}$ indicates ensemble, and * indicates fine-tuned by REINFORCE-based optimization. The top part is for the ensemble models, and the bottom part is for the singles. All values are reported in percentage, with the highest value of each entry highlighted in boldface.

CIDEr score on Karpathy's split of MS COCO dataset, as shown in Table 2. Although the R Product does not perform as well as the P Product or PR Product, the results show that the vector rejection of data vector from weight vector can be used to optimize neural networks. Compared with the P Product and R Product, the PR Product achieves performance improvement across all metrics regardless of cross-entropy training or CIDEr optimization, which experimentally proves the cooperation of vector projection and vector rejection is beneficial to the optimization of neural networks. To intuitively illustrate the advantage of the PR Product, we show some examples of image captioning in supplementary material.

To better understand how the PR Product affects neural networks, we plot the minimum of $|\sin\theta|$ to investigate the dynamics of neural networks to some extent. Figure 3 shows the statistic of the hidden-hidden transfer part in the Attention LSTM, and plots for more layers can be found in the supplementary material. For most of the layers, the minimum of $|\sin\theta|$ in PR Product version is larger than the one in P Product, which means the weight vector and data vector in PR Product are more orthogonal. We argue this is the reason for PR Product to take effect.

Comparison with State-of-the-Art Methods. To further verify the effectiveness of our proposed method, we also compare the PR Product version of our captioning model with some state-of-the-art methods on Karpathy's split of MS COCO dataset. Results are reported in Table 3, of which the top part is for cross-entropy loss and the bottom part is for CIDEr optimization.

Among those methods, SCN-LSTM [8] and SCST:Att2all [31] use the ensemble strategy. GCN-LSTM [41], CAVP [25] and SGAE [39] exploit information of visual scene graphs. Even though we do not use any of the above means of improving performance, our PR Product version of captioning model achieves the best performance in most of the metrics, regardless of cross-entropy training or CIDEr optimization. In addition, we also report our results on the official MS COCO evaluation server in Table 4. As the scene graph models can greatly improve the performance, for fair comparison, we only report the results of methods without scene graph models. It is noteworthy that we just use the same model as reported in Table 3, without retraining on the whole training and validation images of MS COCO dataset. We can see that our single model achieves competitive performance compared with the state-of-the-art models, even though some models exploit ensemble strategy.

6. Conclusion

In this paper, we propose a reliable substitute for the inner product of weight vector $\mathbf{w}$ and data vector $\mathbf{x}$, the PR Product, which involves the information of both the vector projection $\mathbf{P}_{\mathbf{x}}$ and the vector rejection $\mathbf{R}_{\mathbf{x}}$. The length of the direction gradient of PR Product w.r.t. $\mathbf{w}$ is consistently larger than the one in standard inner product. In particular, we show the PR Product version of the fully connected layer, convolutional layer and LSTM layer. Applying these PR Product version modules to image classification and image captioning, the results demonstrate the robust effectiveness of our proposed PR Product. As the basic operation in neural networks, we will apply the PR Product to other tasks like object detection.

Acknowledgement. This work was supported in part by the NSFC Project under Grants 61771321 and 61872429, in part by the Guangdong Key Research Platform of Universities under Grants 2018WCXTD015, in part by the Science and Technology Program of Shenzhen under Grants KQJSCX20170327151357330, JCYJ20170818091621856, and JSGG20170822153717702, and in part by the Interdisciplinary Innovation Team of Shenzhen University.

References

[1] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 3, page 6, 2018.

[2] Pierre Baldi and Peter Sadowski. A theory of local learning, the learning channel, and the optimality of backpropagation. *Neural Networks*, 83:51–74, 2016.

[3] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

[4] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[5] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6298–6306. IEEE, 2017.

[6] Xinpeng Chen, Lin Ma, Wenhao Jiang, Jian Yao, and Wei Liu. Regularizing rnns for caption generation by reconstructing the past with the present. *arXiv preprint arXiv:1803.11439*, 2018.

[7] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In *European conference on computer vision*, pages 15–29. Springer, 2010.

[8] Zhe Gan, Chuang Gan, Xiaodong He, Yunchen Pu, Kenneth Tran, Jianfeng Gao, Lawrence Carin, and Li Deng. Semantic compositional networks for visual captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, 2017.

[9] Jiuxiang Gu, Jianfei Cai, Gang Wang, and Tsuhan Chen. Stack-captioning: Coarse-to-fine learning for image captioning. *arXiv preprint arXiv:1709.03376*, 2017.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[13] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[14] Wenhao Jiang, Lin Ma, Yu-Gang Jiang, Wei Liu, and Tong Zhang. Recurrent fusion network for image captioning. *arXiv preprint arXiv:1807.09986*, 2018.

[15] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1219–1228, 2018.

[16] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[17] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. Multimodal neural language models. In *International Conference on Machine Learning*, pages 595–603, 2014.

[18] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

[19] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[20] Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Babytalk: Understanding and generating simple image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2891–2903, 2013.

[21] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial Intelligence and Statistics*, pages 562–570, 2015.

[22] Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7:13276, 2016.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[24] Chang Liu, Fuchun Sun, Changhu Wang, Feng Wang, and Alan Yuille. Mat: A multimodal attentive translator for image captioning. *arXiv preprint arXiv:1702.05658*, 2017.

[25] Daqing Liu, Zheng-Jun Zha, Hanwang Zhang, Yongdong Zhang, and Feng Wu. Context-aware visual policy network for sequence-level image captioning. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 1416–1424. ACM, 2018.

[26] Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. Improved image captioning via policy gradient optimization of spider. In *Proc. IEEE Int. Conf. Comp. Vis*, volume 3, page 3, 2017.

[27] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 6, page 2, 2017.

[28] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Neural baby talk. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7219–7228, 2018.

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[30] Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. Deep reinforcement learning-based image captioning with embedding reward. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 290–298, 2017.

[31] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 3, 2017.

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[33] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[34] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *Advances in neural information processing systems*, pages 2377–2385, 2015.

[35] Qi Wu, Chunhua Shen, Lingqiao Liu, Anthony Dick, and Anton van den Hengel. What value do explicit high level concepts have in vision to language problems? In *Proceedings of computer vision and pattern recognition*, pages 203–212, 2016.

[36] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[37] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1492–1500, 2017.

[38] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[39] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding scene graphs for image captioning, 2018.

[40] Zhilin Yang, Ye Yuan, Yuexin Wu, William W Cohen, and Ruslan R Salakhutdinov. Review networks for caption generation. In *Advances in Neural Information Processing Systems*, pages 2361–2369, 2016.

[41] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 684–699, 2018.

[42] Ting Yao, Yingwei Pan, Yehao Li, Zhaofan Qiu, and Tao Mei. Boosting image captioning with attributes. In *IEEE International Conference on Computer Vision*, pages 22–29, 2017.

[43] Mehrdad Yazdani. Linear backprop in non-linear networks. In *Advances in Neural Information Processing Systems Workshop*, 2018.

[44] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of computer vision and pattern recognition*, pages 4651–4659, 2016.

[45] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.