Layer-wise Gradient Decoupling

Dennis Alexander Mertens Velasquez (i6206990)

1 Introduction

Suppose we have n training examples Z_0, Z_T, where $Z_0 = [z_0^0, \ldots, z_0^{n-1}]$ denotes our inputs and $Z_T = [z_T^0, \ldots, z_T^{n-1}]$ our targets, and a feed-forward differentiable circuit of depth T like in figure 1.

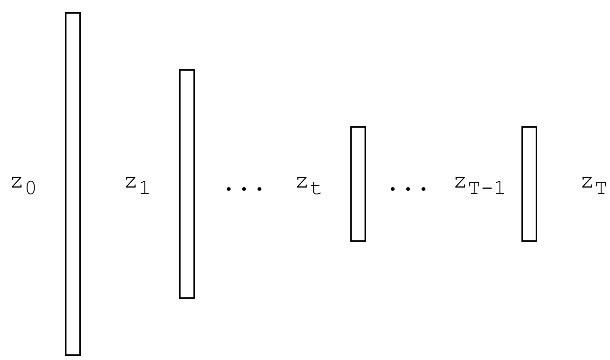

Figure 1: A feed-forward differentiable circuit of depth T; z_0 denotes the input, $z_1, \ldots, z_{T-1}$ denote the hidden activations, and z_T denotes the output. Each layer $t \in \{0, \ldots, T-1\}$ has parameters θ_t and activation function f.

Usually, when applying backpropagation (BP) and gradient descent (GD), we optimize the parameters $\Theta = [\theta_0, \ldots, \theta_{T-1}]$ of every layer by nudging them according to their gradients.[1] In such a setting, the gradients of θ_t depend on the gradients and activations of layers $t+1, \ldots, T-1$. This sequential dependence is at the heart of numerous complications during training. I propose a relaxation of the standard formulation of such circuits that promises to mitigate some difficulties and completely remove others.

2 Motivation

...

[1]To be clear: BP is responsible for propagating the errors through the circuit, and GD is responsible for adjusting the parameters accordingly.

3 Approach

Let z_t^i denote the target activation of the tth layer given the ith training example, and $\hat{z}_t^i$ its estimate. For $t \in \{0, T\}$, the targets are given. For $t \in \{1, \ldots, T-1\}$, assume auxiliary trainable parameters $Z_t = [z_t^0, \ldots, z_t^{n-1}]$ and define the hidden activations as

$$\hat{z}_{t+1}^i = f(z_t^i, \theta_t), \forall i \in \{0, \ldots, n-1\}. \tag{1}$$

Usually, any activation $\hat{z}_{t+1}^i$ is defined in terms of the preceding $\hat{z}_t^i$, hence the sequential dependence. In contrast, the dependence is broken by introducing an auxiliary z_t^i. Naturally, if we train such a system, it will not generalize because only the last layer will learn to map z_{T-1}^i to z_T^i whilst every other part of the circuit will settle for arbitrary configurations. To fix this, we must enforce that

$$\hat{z}_t^i \approx z_t^i, \forall t \in \{1, \ldots, T-1\}, \forall i \in \{0, \ldots, n-1\}. \tag{2}$$

During training, some loss function $\mathcal{L}(Z_T, \hat{Z}_T) = \sum_{i=0}^{n-1} L_y(z_T^i, \hat{z}_T^i)$ is always required, where L_y measures the output error.[2] By extending the loss to include terms that regulate the hidden activations, the condition in equation 2 is enforced. That is,

$$\mathcal{L}(Z_T, \hat{Z}_T) = \sum_{i=0}^{n-1} L_y(z_T^i, \hat{z}_T^i) + \sum_{t=1}^{T-1} \sum_{i=0}^{n-1} L_h(z_t^i, \hat{z}_t^i), \tag{3}$$

where L_h measures the hidden error.

I hypothesize that by enforcing the condition in equation 2, we will obtain a circuit that settles at a configuration similar to that of the same circuit trained with standard BP and GD.[3] Except that the proposed approach should be completely immune to the difficulties listed in section 2.

4 Evidence

...

5 Questions

...

[2]Usually, loss functions are augmented with a regularization term, but it is irrelevant in this context, and thus not included.

[3]Note that I have to define what I mean by *similar*, as it could be interpreted in terms of the distance between the two circuits in parameters space or the discrepancy in performance. Such a definition will be provided in due time.