On Gradient

Giulio Galvan

17 marzo 2015

Sommario

1 How to compute gradient

1.1 Backpropagation

First of all we need to define a loss function over the training data, so we define a dataset as

$$D = \{x^{(i)} \in \mathbb{R}^p, y^{(i)} \in \mathbb{R}^q, i \in [1, N]\} \tag{1}$$

and the loss function as

$$L(W) = \frac{1}{N} \sum_{i=1}^{N} L^{(i)}(W) \tag{2}$$

where W is represents all the weights of the net. The network is defined by

$$a_l \triangleq \sum_j w_{lj} \phi_j \tag{3}$$

$$\phi_l \triangleq \sigma(a_l) \tag{4}$$

where w_{lj} is the weight of the connection between neuron j and neuron l and σ is the non linear activation function

So we can compute partial derivatives with respect to a single weight w_{lj}, using simply the chain rule, as

$$\frac{\partial L^{(i)}}{\partial w_{lj}} = \frac{\partial L^{(i)}}{\partial a_l} \cdot \frac{\partial a_l}{\partial w_{lj}} = \delta_l \cdot \phi_j$$

where we put

$$\delta_l \triangleq \frac{\partial L^{(i)}}{\partial a_l} \tag{5}$$

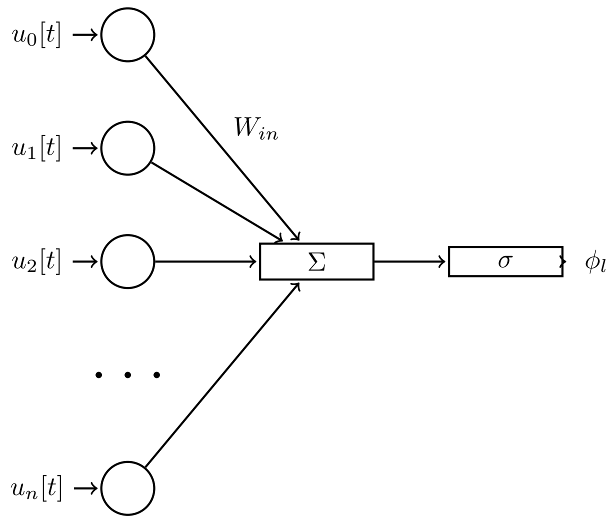

Figura 1: Modello per RNN

So we can easily compute $\delta_u = \frac{\partial L^{(i)}}{\partial a_u}$ for each output unit u once we choose a differentiable loss function; note that we don't need the weights for such a computation.

Let $P(l)$ be the set of parents of neuron l, formally:

$$P(l) = \{k : \exists \text{ a link between } l \text{ and } k \text{ with weight } w_{lk}\} \tag{6}$$

Again, simply using the chain rule, we can write, for each non output unit l:

$$\delta_l = \sum_{k \in P(l)} \frac{\partial L^{(i)}}{\partial a_k} \cdot \frac{\partial a_k}{\partial a_l} = \sum_{k \in P(l)} \delta_k \cdot \frac{\partial a_k}{\partial \phi_l} \cdot \frac{\partial \phi_l}{\partial a_l} = \sum_{k \in P(l)} \delta_k \cdot w_{kl} \cdot \sigma'(a_l) \tag{7}$$

For output units instead we can compute $\delta_u = \frac{\partial L^{(i)}}{\partial a_u}$ directly once we define the loss function.

1.2 Backpropagation matrix notation

Here we rewrite the previously derived equations in matrix notation.

Let us define the weight matrix $W_i \in \mathbb{R}_{(p(i),p(i-1))}$, whose element $W_{i,j}$ is the weight of the arc which links neuron j from level i-1 to neuron i from level i, where $p(i)$ is the neuron number for i^{th} level.

$$\overrightarrow{\phi}_1 \triangleq \overrightarrow{x} \tag{8}$$

$$\overrightarrow{a}_{i+1} \triangleq W_{i+1} \cdot \overrightarrow{\phi}_i \tag{9}$$

$$\overrightarrow{\phi}_{i+1} \triangleq \sigma(\overrightarrow{a}_{i+1}) \tag{10}$$

where $\sigma(\cdot)$ is the non-linear activation function and it's applied element by element. We can rewrite equation 7 in matrix notation as:

$$\frac{\partial L}{\partial W_i} = \frac{\partial L}{\partial \overrightarrow{a}_i} \cdot \frac{\partial \overrightarrow{a}_i}{\partial W_i}^T = \Delta_i \cdot \overrightarrow{\phi}_{i-1}^T \tag{11}$$

where

$$\Delta_i \triangleq \frac{\partial L}{\partial \overrightarrow{a}_i} \tag{12}$$

$$\Delta_i = W_{i+1}^T \cdot \Delta_{i+1} \circ \sigma(\Delta_i) \tag{13}$$

Riferimenti bibliografici

[1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory, 1995.