Implementation of Speech Recognition with LSTM and CTC

1

0.1. Framewise Classification with LSTM Network

Our project consists of two main components-framewise classification and phoneme decoding. In the first part, we train a classifier that can predict each frame separated by 10ms (overlap 5ms). For one sentence, there are about 500 to 700 frames. We use Mel-Frequency Cepstrum Coefficients (MFCC) to abstract feature from the acoustic voice data. For every frame, there are 13 features. Recurrent Neural Network is a good model to train from input data sequence to output data sequence, but as the frames comes chronologically, recurrent neural network may forget the information from the previous frame. And in this project, we implement a special recurrent neural network, Long Short-Term Memory, which can solve this problem by introducing a different architecture of hidden state that can have longer memory.

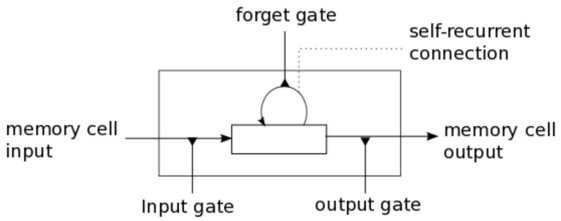

Figure 1. Construct of LSTM Model

From Figure 1. Structure of an LSTM network, there are three gates in the hidden state, **Input Gate**, **Output Gate** and **Forget Gate**.

- **Forward Step**
  Input gate: The input gate defines how much of the newly computed state for the current input you want to let through at time t:

$$i_t = sigmoid(W_{x_i}x_t + W_{h_i}h_{t-1} + b_i)$$

  Forget Gate: The forget gate defines how much of the previous state you want to let through at time t:

$$f_t = sigmoid(W_{x_f}x_t + W_{h_f}h_{t-1} + b_f)$$

  While sometimes, we also want the memory cell forget some of the previous information, so we set a memory upper bound. When the number of memory cells a hidden state tries to remember reaches this upper bound, any older state memory will be forgotten.

$$f_t = 1 - i_t$$

Output Gate: The output gate defines how much of the internal state you want to expose to the external network (higher layers and the next time step) at time t:

$$o_t = sigmoid(W_{x_o} x_t + W_{h_o} h_{t-1} + b_o)$$

x_t is the input at t time. h_{t-1} is the output at t-1 time. c_{int} the candidate value for the states of the memory cells at time t:

$$c_{int} = tanh(W_{x_c} x_t + W_{h_c} h_{t-1} + b_{c_{int}})$$

c_t is the memory state at time t:

$$s_t = f_t s_{t-1} + i_t c_{int}$$

h_t is the output of this network at time t:

$$h_t = o_t tanh(s_t)$$

- Softmax Layer
  We use the output from LSTM and using softmax to calculate y, which is the input of CTC.

$$y = softmax(V h_t)$$

Update V:

$$\partial V = dy \times h_t$$

Derivative of h backing to the LSTM:

$$\partial h_t = dy \cdot V$$

- Back Propagation
  From $h_t = o_t tanh(s_t)$, we can calculate:

$$\partial s_t = o_t \partial h_t + \partial s_{t-1} \quad and \quad \partial o_t = \partial s_t \partial h_t$$

From $s_t = f_t s_{t-1} + i_t c_{int}$, we can calculate:

$$\partial f_t = s_{t-1} \partial s_t \quad and \quad \partial i_t = c_{int} \partial s$$

$$\partial i_t = (c_{int} - s_{t-1}) \partial s \quad and \quad \partial c_{int} = i_t \partial s$$

From these sigmoid functions, we can calculate:

$$\partial W_c = (1 - c_{int}^2) \partial c \times x_t \quad and \quad \partial H_c = (1 - c_{int}^2) \partial c \times h_{t-1}$$

$$\partial W_o = (1 - o_t) o_t \partial o \times x_t \quad and \quad \partial H_o = (1 - o_t) o_t \partial o \times h_{t-1}$$

$$\partial W_i = (1 - i_t) i_t \partial i \times x_t \quad and \quad \partial H_i = (1 - i_t) i_t \partial i \times h_{t-1}$$

Forget:

$$\partial W_f = (c_{int}^2 - 1) \times x_t \quad and \quad \partial H_f = (c_{int}^2 - 1) \times h_{t-1} \; or$$

Remember:

$$\partial W_f = (1 - f_t) f_t \partial f \times x_t \quad and \quad \partial H_f = (1 - f_t) f_t \partial f \times h_{t-1}$$

0.2. Decoding with CTC Network

LSTM has the exact same limitation as RNN does. They both cannot handle the mapping of two sequences of different length. Some other measures mentioned above like HMM have been used to address this issue. CTC is introduced to cope with this problem by incorporating another layer at the end of the LSTM network and in a way making the whole thing trainable by backpropagation.

For every output from LSTM and softmax network, it contains the probability of each distinct phoneme given a frame of speech at each time step. The objective is to maximum the path which is defined as a sequence that can be converted to the correct frame sequence just by removing repetition of phonemes and blanks.
First we should define conversions from a path to an output sequence to be aaaabbb=a_bbbb=aaaaab=
a_b_=ab while it is not equal to a_ab or ab_b.
Second we define the objective function as following:

$$O = -\sum_{(\mathbf{X},\mathbf{Z}\in S)} \ln(p(\mathbf{z}|\mathbf{x}))$$

z is the true output sequence, **x** is the input of speech data as frames of MFCC features.
Third we find that it is impossible to calculate all the path which belongs to the correct path. So we should use forward and backward algorithm to calculate the $p(\mathbf{z}|\mathbf{x})$.
We define that the true label sequence is **l** and we add blanks to the beginning and the end and inserted between every pairs of labels. The length of **l'** is 2—**l**—+1. We define $y^t_{l'_s}$ which means the possible of l'_s at t frame.
Now calculate forward:
Initialisation:

$$\alpha_1(1) = y^1_{l'_1}$$
$$\alpha_1(2) = y^1_{l'_2}$$
$$\alpha_1(s) = y^1_{l'_s}, \forall s > 2$$

and recursion:

$$\alpha_t(s) = \bar{\alpha}_t(s) y^t_{l'_s} \;\; if\ I'_s = blank\ or\ I'_{s-2} = I'_s\ or$$
$$\alpha_t(s) = (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) y^t_{l'_s} \;\; otherwise$$
$$where\ \bar{\alpha}_t(s) = \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$$

Now calculate backward:
Initialisation:

$$\beta_T(|l'|) = y^T_{l'_{|l'|}}$$
$$\beta_T(|l'|-1) = y^T_{l'_{|l'|-1}}$$

$$\beta_T(s) = y^1_{l'_s}, \forall s < |l'| - 1$$

and recursion:

$$\beta_T(s) = \bar{\beta}_t(s) y^t_{l'_s} \;\; if \; I'_s = blank \; or \; I'_{s+2} = I'_s or$$
$$[\beta_T(s) = (\bar{\beta}_t(s) + \beta_{t+1}(s+2)) y^t_{l'_s} \;\; otherwise$$
$$where \; \bar{\beta}_t(s) = \beta_{t+1}(s) + \beta_{t+1}(s+1)$$

Then normalize α and β

$$p(\mathbf{l}|\mathbf{x}) = \sum_{s=1}^{|l'|} \frac{\alpha_t(s)\beta_t s}{y^t_{l'_s}}$$

$$\frac{\partial O}{\partial h^t_k} = y^t_k - \frac{1}{y^t_k Z_t} \sum_{s \in lab(\mathbf{Z},k)} \hat{\alpha}_t(s)\hat{\beta}_t(s)$$

$$where \; Z_t = \sum_{s=1}^{|l'|} \frac{\hat{\alpha}_t(s)\hat{\beta}_t(s)}{y^t_{|l'_s|}} \; and \; lab(\mathbf{l}, k) = \{s : l'_s = k\}$$