Text Recognition
梁鼎
Pipeline
RNN
LSTM
GRNN
RNN structure
CTC
Forward-Backward algorithms
Results

Text Recognition

梁鼎

September 16, 2015

Contents I

1 Pipeline

2 RNN
- LSTM
- GRNN
- RNN structure

3 CTC
- Forward-Backward algorithms

4 Results

Task

Figure: Left are correct ones and incorrect examples are listed on the right.

Pipeline

RNN

LSTM

GRNN

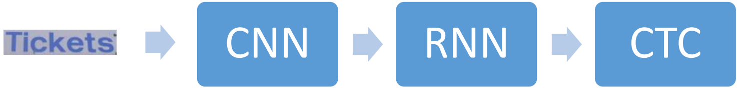
RNN structure

CTC

Forward-Backward algorithms

Results

An end to end pipeline



RNN (Recurrent Neural Network)

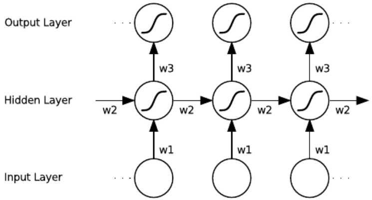


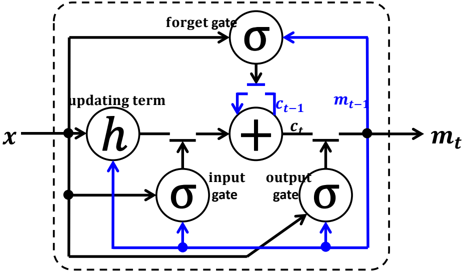
Figure: A normal RNN

RNN is used to model the relation between frames. Both input and output can be sequences or static.

1. one to many: image caption
2. many to one: video classification
3. many to many: text recognition, machine translation

LSTM (Long Short Term Memory)

Let's recall LSTM (Hochreiter and Schmidhuber, 1997):
Advantage: avoid vanishing
Disadvantage: complex



$$
\begin{aligned}
i_t &= \sigma(W_{ix}x_t + W_{im}m_{t-1}) & (4)\\
f_t &= \sigma(W_{fx}x_t + W_{fm}m_{t-1}) & (5)\\
o_t &= \sigma(W_{ox}x_t + W_{om}m_{t-1}) & (6)\\
c_t &= f_t \odot c_{t-1} + i_t \odot h(W_{cx}x_t + W_{cm}m_{t-1}) & (7)\\
m_t &= o_t \odot c_t & (8)
\end{aligned}
$$

GRNN (Gated Recurrent Neural Network)

GRNN (Cho et al. 2014)
Main ideas:

1. keep around memories to capture long distance dependencies
2. allow error messages to flow at different strengths depending on the inputs

GRUs (Gated Recurrent Units)

- Standard RNN computes hidden layer at next time step directly

$$h_t = f\left(W^{(hh)}h_{t-1} + W^{(hx)}x_t\right)$$

- GRU first computes an **update gate** based on current input vector and hidden state

$$z_t = \sigma\left(W^z x_t + U^z h_{t-1}\right)$$

- Compute **reset gate** similarly but with different weights

$$r_t = \sigma\left(W^r x_t + U^r h_{t-1}\right)$$

GRUs (Gated Recurrent Units)

- **New memory content**:

$$\tilde{h}_t = \tanh\left(Wx_t + r_t \circ Uh_{t-1}\right)$$

  If reset gate unit is 0, then this ignores previous memory and only stores the new vector information

- **Final memory** at time step combines current and previous time steps:

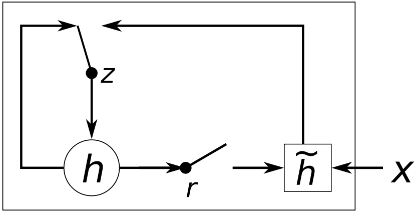
$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$$

GRU intuition

$$
\begin{aligned}
z_t &= \sigma\left(W^z x_t + U^z h_{t-1}\right) \\
r_t &= \sigma\left(W^r x_t + U^r h_{t-1}\right) \\
\tilde{h}_t &= \tanh\left(W x_t + r_t \circ U h_{t-1}\right) \\
h_t &= z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t
\end{aligned}
$$

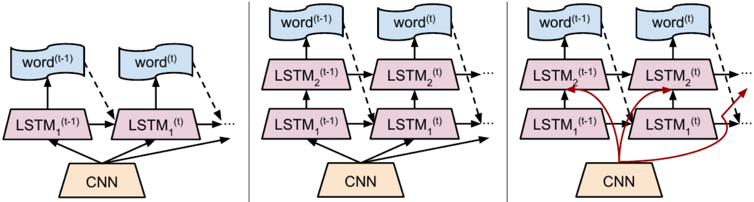
- If reset is close to 0, ignoring previous hidden state allows model to drop information that is irrelevant in the future
- Update gate z controls how much of past state should matter now.
  - If z close to 1, then we can copy information in that unit through many time steps! **Less vanishing gradient**!
- Units with short-term dependencies often have reset gates very active.

GRU intuition



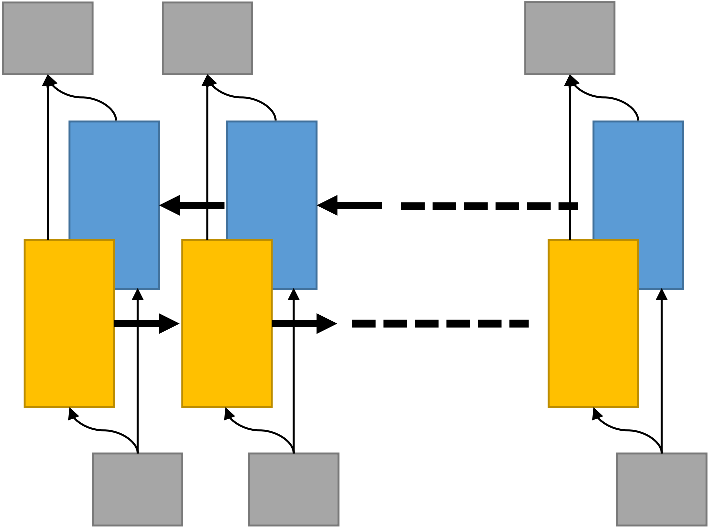
$$z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right)$$
$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right)$$
$$\tilde{h}_t = \tanh\left(Wx_t + r_t \circ Uh_{t-1}\right)$$
$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$$

LRCN

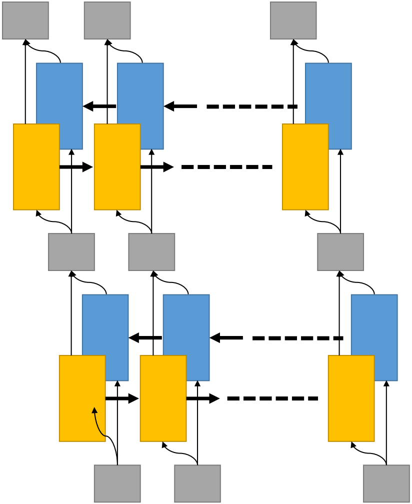


Single Layer **LRCN**$_{1u}$ | Two Layers, Unfactored **LRCN**$_{2u}$ | Two Layers, Factored **LRCN**$_{2f}$

Bidirectional RNN



Stack RNN



CTC (Connectionist Temporal Classification)

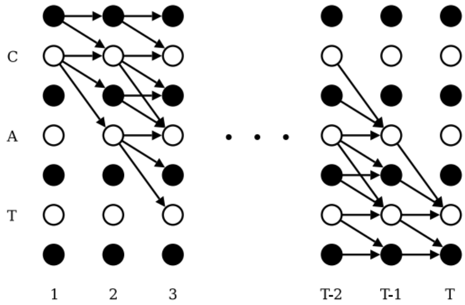
target: $F(a-ab-) = F(-aa--abb) = aab$

$$
\begin{aligned}
p(\pi|x) &= \prod_{t=1}^{T} y_{\pi_t}^{t} \\
p(l|x) &= \sum_{\pi \in F^{-1}(l)} p(\pi|x)
\end{aligned}
$$

maximize all the training samples probobilities

$$
Loss = -\ln \prod p(l|x) = -\sum \ln p(l|x)
$$

Forward-Backward algorithms



$$
\begin{aligned}
\alpha(t,u) &= \sum_{\pi \in V(t,u)} \prod_{i=1}^{t} y^{i}_{\pi_i} \\
p(l|x) &= \alpha(T, U') + \alpha(T, U'-1)
\end{aligned}
$$

Forward-Backward algorithms

β is calculated the same way as α, but in a reversed direction.

$$\alpha(t,u)\beta(t,u) = \sum_{\pi\in X(t,u)} \prod_{i=1}^{T} y^t_{\pi_t} = \sum_{\pi\in X(t,u)} p(\pi|x)$$

$$p(l|x) = \sum_{u=1}^{|l'|} \alpha(t,u)\beta(t,u)$$

results

- English text recognition
  - images are hard for recognition
  - CNN+stack LSTM+CTC: 90% accuracy, state-of-art 92.3%
- Chinese long weibo text recognition
  - too much labels compared to English tasks
  - CNN+CTC: 98% accuracy
  - CNN+RNN+CTC: hard to converge to an acceptable state (GRNN is much better, though both LSTM and GRNN suck)

Text Recognition

梁鼎

Pipeline

RNN

LSTM

GRNN

RNN structure

CTC

Forward-Backward algorithms

Results

Thanks!