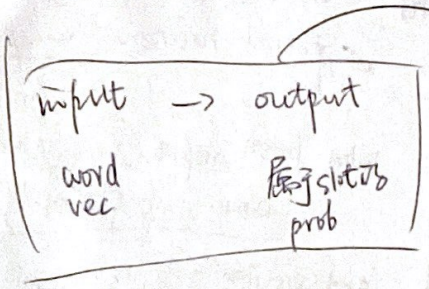
RNN

Slot filling

slot { destination
time of arrival



problem → NN needs memory → RNN

arrive Taipei on November 2nd
~~arrive~~ dest

Leave Taipei on November 2nd
place of departure.

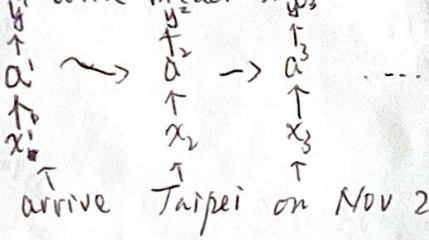
把 output of hidden layer 存入 memory 中.

input: Taipei (1) Taipei (2)

output: 數字1 數字2

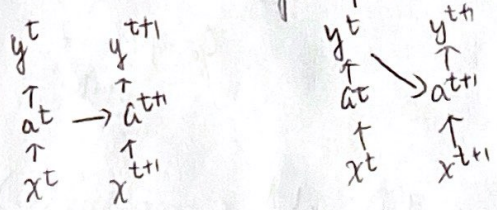
RNN. change the sequence order will change the output

prob of "arrive" in each slot

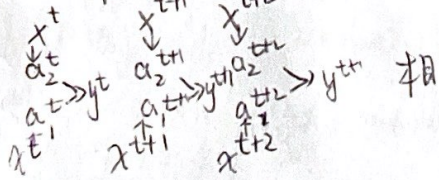


不是3个network. 是同一个network在不同的时间点被使用了3次.

Elman network & Jordan network (good performance)



Bidirectional RNN



相對看到 x^t 之前和之後的才預測

LSTM

RNN 是有 short-term 的每一个 input 进来，memory 的值会被更新

special neuron: 4 inputs → 1 output

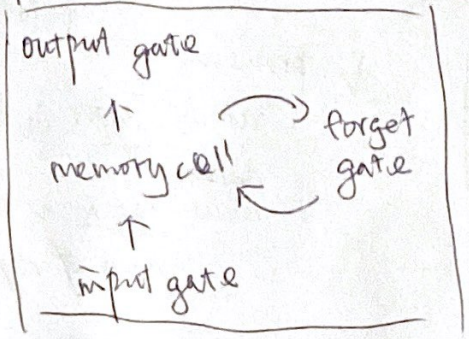
z z_i z_f z_o (z_i z_f z_o 是 $f(z)$)

why both input & output:

∵ input gate 可以控制是否存入 memory.

activation 通常选 sigmoid function. $f \in (0,1)$

for gates



z : input.

z_i : input gate $z_i = Wz$

c : memory

z_f : forget gate

z_o : output gate

new memory: $c' = g(z) f(z_i) + c f(z_f)$

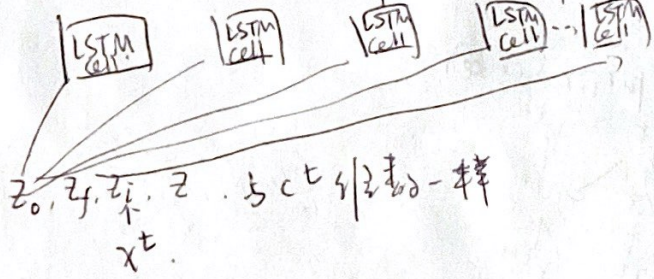
0 代表遗忘，1 代表记得.

activation function

output: $a = h(c') f(z_o)$

c^{t-1} 是当层 neuron 个数的维数 vector

c^{t-1}: 1×5



y^t 与 c^{t-1} 的维数一样

y^{t+1}

$\rightarrow c^t$

z_o, z_f, z_i, z 与 c^t 维数一样

x^t

z_o, z_f, z_i, z

x^{t+1}

实际上 x^t 是存储 input word vector 以及 $h(c^{t-1})$. c^{t-1} (concatenate)

peephole

keras supports: LSTM. GRU. Simple RNN layers.

GRU 为 3-4 gate 但 performance 差不多.

update gate $z_t = f(W_z x_t + U_z h_{t-1} + b_z)$

reset gate $r_t = f(W_r x_t + U_r h_{t-1} + b_r)$

$\hat{h}_t = \phi(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$

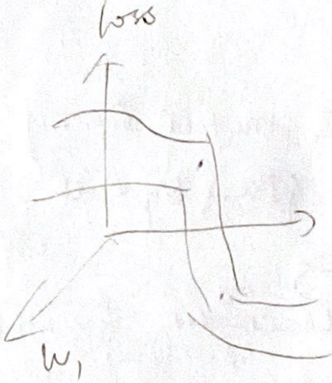
$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t$

BPTT (backpropogation ~~through~~ through time) ∵ RNN 随着时间变化. $y_t = \sigma(W h_t)$

RNN training.

问题

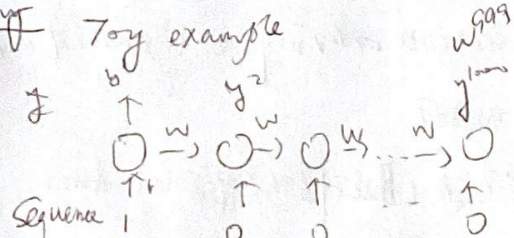
1. The error surface is rough.



如果在悬崖，则 gradient 很大 para update 就会很远

Solution — Clipping.

Why — Toy example



$w=1.01$. $y^{1000}=20000$

$w=0.99$. $y^{1000}=0$

来自 sequence 长度. 同一个 weight 在不同时间上反复被使用. (explode or vanish)

LSTM 可以解决 gradient vanishing 但不能解决 gradient explode

∴ 可以把 learning rate 设得小,

why? RNN memory 会被洗掉 (→ 相当于 forget)

LSTM memory 是相加的 除非 forget gate closed → No gradient vanishing if forget gate is opened
会加 input.

GRU 只有2个 gate. 参数少更 robust. 如果 LSTM overfitting 很严重 可以试 GRU.

GRU 把 input gate 和 forget gate 连动起来. 当 input gate 打开, forget gate 会自动地关闭 forget

Clockwise RNN 和 Structurally constrained Recurrent network (SCRN)

Valina RNN initiated with Identity matrix + ReLU activation performance 很好 (好于 LSTM)

到前已到的都是 output 与 input 同维的 RNN. 如 slot filling

但 RNN 可以作 many to one. 如 sentiment analysis 等 document → embedding layer → RNN → output

many to many: 长到短 ... 语音辨识, CTC ... google: character ... ; 长到长. 如 语音翻译, attention, feedforward network

RNN

* Beyond sequence.

· syntactic parsing (也是 structure learning)

现在可将 parsing tree 转化成 sequence (S(NP NNP)$_{NP}$ (VP VBZ (NP DT NN)$_{NP}$)$_{VP}$)$_S$

LSTM有记忆力，不会忘加右括号.

* sequence to sequence auto-encoder (learning) (有顺序的文法) 考虑 word sequence order的情况下作 document embedding

如果原文 → encoder → decoder → 原文，可以成功的话

的 document ~~embed~~ vector 代表了重要资讯 (且这种方法不需要 label data)

*' skip-thought 是 类似的 output 下一个句子的方法，能更好地理解语义.

*'' 也可以作 hierarchy. word vec → sentence vec → documen vec → sentence vec → word vec

*''' dimension reduction for a sequence with variable length.

audio segments (word-level) → fixed-length vector

y_1 y_t

RNN encoder → RNN decoder 一起 train

x_1 x_t

Attention-based model → reading comprehension

visual QA

speech QA

RNN vs structured learning → integrate

deep. no constriant 用 viterbi algo

bi-directional LSTM + CRF / structured SVM 做 tagging

output as feature