Recurrent Neural Network (RNN)

Recurrent Neural Network

Sequence — series + time
length — Input / Output

Architecture

One to One — Normal NN

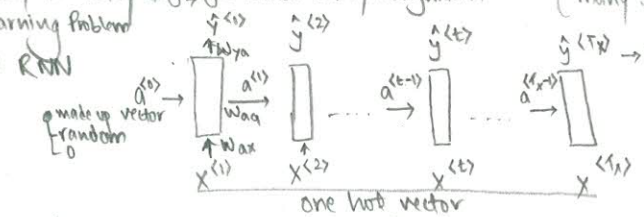
RNN:
- One to many — Music generation
- many to one — Sentiment Analysis
- many to many — $T_x = T_y$ — Named Entity Recognition
  - $T_x \neq T_y$ — Machine Translation

attention based

Many to many → $T_x = T_y$ - Named Entity Recognition

Supervised Learning Problem

Unidirectional RNN



$\hat{y}^{\langle T_x\rangle}$ → binary classification of each word

Standard NN
- Input & output - different size
  - max-padding
- feature/parameter sharing

Forward Propagation

vocabulary / Dictionary - Representative words
- Top occurring n words
- Unknown

sequence of word / sentence

$a^{\langle 0\rangle} = \vec{0}$

$a^{\langle t\rangle} = g_1(W_{aa}a^{\langle t-1\rangle} + W_{ax}x^{\langle t\rangle} + b_a)$ — tanh / ReLU

$\hat{y}^{\langle t\rangle} = g_2(W_{ya}a^{\langle t\rangle} + b_y)$ — Sigmoid / Softmax

weight W_{ax}: input / output

$\rightarrow W_a\left[a^{\langle t-1\rangle}, x^{\langle t\rangle}\right] + b_a \quad W_a = [W_{aa} \mid W_{ax}]$ stacked horizontally

$\rightarrow W_y a^{\langle t\rangle} + b_y$

Backward Propagation through time

$L(\hat{y}, y) = \sum_t L^{\langle t\rangle}(\hat{y}^{\langle t\rangle}, y^{\langle t\rangle})$

$= -y^{\langle t\rangle}\log\hat{y}^{\langle t\rangle} - y(1-y^{\langle t\rangle})\log(1-\hat{y}^{\langle t\rangle})$

Gradient Problem (Layer)
- Vanishing
  - Backprop error → memory
  - local influences
  - language model: word — singular / plural — long range dependencies
- Exploding
  - gradient↑ → numerical overflow (NaN)
    - gradient clipping

Language Sequence — Model — speech recognition

sentence → sequence of words

$P(y^{\langle 1\rangle}, y^{\langle 2\rangle} \cdots y^{\langle T_y\rangle})$ - language model

Training Set

Tokenize - vocabulary - map — One hot vector / Indices
- EOS - end of sentence
- Punctuation
- Uncommon words

Sequence → One word at a time $x^{\langle t\rangle} \rightarrow y^{\langle t-1\rangle}$

Sample Novel Sequence

level — word / character

choose start

predict softmax → random sampling....

till — EOS / no of words

challenge - Ununknown words

Language model and sequence generation

Language model

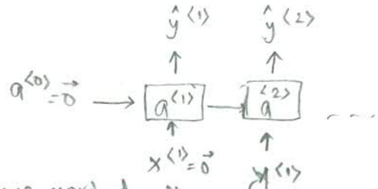
Speech recognition
Machine translation

probability (sentence)
next "
(sequence of words)

$y^{\langle i \rangle}$

Training set - large corpus of text
tokenize, <EOS> ⟶ vocabulary
L end of sentence
& unknown words

conditional prob



Sampling novel sequences
Sample a sequence (sentence) from trained RNN
$\hat{y}^{\langle 1 \rangle}$ - probability (i^{th} word) → $dist^n$
softmax $dist^n$ random sample
- reject unknown words

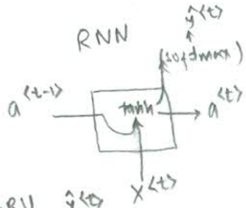
predicts one word at a time, going from left to right.

$L(\hat{y}^{\langle t \rangle}, y^{\langle t \rangle}) = -\sum_i y_i^{\langle t \rangle} \log \hat{y}_i^{\langle t \rangle}$

continue till <EOS> / # of words.

word level language model vocabulary - words
character " " " " - letters... ← longer sequences / complex computer extensive

RNN



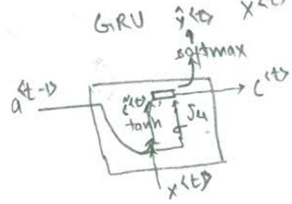
Vanishing ↑ gradients
Exploding ↓ → NaN

capturing long term dependencies → pronoun - [singular / plural
memorize

RNN - local influences back propagate earlier - error - weakness

Exploding gradient [NaN / numerical overflow
gradient clipping - rescale gradient vector
- max value

GRU



Gated Recurrent Unit (GRU) (2014) [long range connection - capture
vanishing gradient

memory cell → $c^{\langle t \rangle}$ $a^{\langle t \rangle}$

Candidate value → $\tilde{c}^{\langle t \rangle} = \tanh(W_c[\Gamma_r * c^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_c)$

Update Gate → $\Gamma_u = \sigma(W_u[c^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_u)$ — mostly 0 or 1
gamma update
Gated fence

vanishing gradient ← ↑-ve
↳ $c^{\langle t \rangle}$ is maintained

$c^{\langle t \rangle} = \Gamma_u * \tilde{c}^{\langle t \rangle} + (1 - \Gamma_u) * c^{\langle t-1 \rangle}$

Relevance Gate $\Gamma_r = \sigma(W_r[c^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_r)$ $a^{\langle t \rangle} = c^{\langle t \rangle}$

LSTM (Long short term Memory unit - general) (1997)

$\tilde{c}^{\langle t \rangle} = \tanh(W_c[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_c)$

Update Gate $\Gamma_u = \sigma(W_u[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_u)$

Forget Gate $\Gamma_f = \sigma(W_f[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_f)$

Output Gate $\Gamma_o = \sigma(W_o[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_o)$

$c^{\langle t \rangle} = \Gamma_u * \tilde{c}^{\langle t \rangle} + \Gamma_f * c^{\langle t-1 \rangle}$

$a^{\langle t \rangle} = \Gamma_o * \tanh c^{\langle t \rangle}$

peephole connection → add $c^{\langle t-1 \rangle}$ term while calculating gate values.

Bidirectional RNN
BRNN

- Unidirectional / forward] activation
+ backward (from future) - backward connection

acyclic graph forward backward
$\hat{y}^{\langle t \rangle} = g(W_y[\overrightarrow{a}^{\langle t \rangle}, \overleftarrow{a}^{\langle t \rangle}] + b_y)$

Properties
- entire sequence of data
- real time speech recognition

Deep RNNs - complex function - stack multiple layers
computationally expensive to train

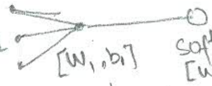
Natural Language Processing & Word Embeddings

NLP - word representation

- One hot encoding [Vocabulary, Index] · words are independent
  - ↓ featurised representation – similar words – [distance – Euclidean, inner product]
- Word Embeddings ← featurised representation
  - words are embedded in high dimension space
  - Get [Train, Pretrained – Transfer learning]
    - ↓data – [Named Entity Recognition, text summarization, co-reference resolution, Parsing]
    - ↑ – [language model, machine translation]
  - features – parallel in n dimension: $e_w - e_{w_1} \parallel e_{f_1} - e_{f_2}$
  - $\arg\max_w \left(\text{similarity}\left(e_w, e_{f_1} - e_{f_2} + e_{w_1}\right)\right)$
  - cosine: $\text{sim}(u,v) = \frac{u^T v}{\|u\|_2 \|v\|_2}$
  - Analogy reasoning → similarity
  - Visualize: nD → 2D (non-linear transformation) } t-sne algorithm
  - Embedding Matrix ← learn from gradient descent
    - ↳ to convert one hot encoding → embedded representation [multiplication, look-up]

Learn Word Embeddings – E

1) 2003 Supervised learning problem – Context → Target pair
   - Context: given n word – [before, after, nearby – skip gram model]
   - Target: ∈ X
   - One hot encoding (O_j) × e_j embedded vector → [W_1, b_1] → softmax [W_2, b_2] (parameters)



2) 2013 Word2Vec Algorithm
   - (i) Skip gram model: Context → Target Pair
     - Context: random; Target: random [± n word window]
     - ↳ Heuristic balance [common, uncommon] words
     - $O_c \rightarrow E \rightarrow e_c \rightarrow$ softmax $\hat{y}$ (E: Embedding matrix)
     - $P(t|c) = \frac{e^{\theta_t^T e_c}}{\sum e^{\theta_t^T e_c}}$ [θ_t → parameter – output t, can includ bias]
     - Loss function $L(\hat{y}, y) = \sum y_i \log \hat{y}_i$
     - [slow → Hieardial Softmax – tree: computional complexity increases by log not linearly.]
   - (ii) CBoW model: Continuous backwards Model – surrounding context

3) 2013 Negative Sampling
   - Context → Target Pair → Target (0, 1)
     - select, select → 1
     - K time: select, random → 0 – negative sampling
     - positive to negative ratio
     - random – [observed frequency – stop words, 1/vocab size, Emphorical $\frac{f(w_i)^{3/4}}{\sum_j f(w_j)^{3/4}}$]
   - Basic logistic regression model: $P(y=1|c,t) = \sigma(\theta_t^T e_c)$ — softmax func, sigmoid

4) 2014 GloVe Algorithm (Global Vector)
   - X_{ij} = # i appear in context j (i ↑ t, j ↑ c) – symmetric – nearby
   - minimize $\sum_i \sum_j f(X_{ij}) \left(\theta_i^T e_j + b_i + b_j - \log X_{ij} \right)^2$ ($\theta_i^T e_j$: symmetric)
     - weighting term [frequent, infrequent] words
     - $X_{ij} = 0$ ← log
     - starts with uniformly random
     - Optimize → GD
     - $e_w^{final} = \frac{e_w + \theta_w}{2}$
   - cannot guarantee that individual component of feature representation → interpretable
     - $\theta_i^T e_j = (A\theta_i)^T (A^{-T} e_j)$

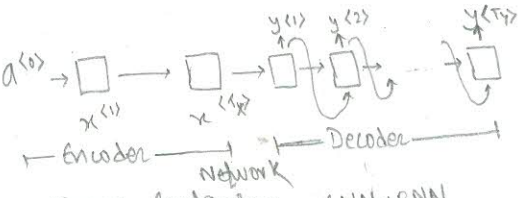
Application

- Sentiment classification
  - Use word embedding: $O \rightarrow E \rightarrow e \rightarrow$ sum/average $\rightarrow \hat{y}$ softmax
    - ignore word order
  - Use RNN many to one architecture
- Debiasing Word Embedding – 2016
  - bias – [gender, ethenicity, sexual orientation] → [college admission, job, loan, criminal justice]
  - 1) Identify bias direction: difference – [average 2D, Singular value decomposition (SVD)]
  - 2) Neutralization
    - word not definational
    - intrensic bias
  - 3) Equalize pair
    - difference – pairwise

Sequence Model

Sequence to sequence Model

2014



2014/2015 Image Captioning CNN+RNN

Pick up most likely sentence

Language Model + condition → Machine Translation
└ Encoder

output $P(y^{\langle 1\rangle}, \dots, y^{\langle T_y\rangle} | x)$

- random sample
- most likely translation $\arg\max_{y^{\langle 1\rangle}, \dots, y^{\langle T_y\rangle}} P(y^{\langle 1\rangle}, \dots, y^{\langle T_y\rangle} | x)$

Search Algorithm
- Greedy Search
  - choose one word at a time
    - assume independent
      - joint probability
- Exact search algorithm
  - no. of possible output ↑
- Approx search algorithm
  - BFS (Breadth First Search)
  - DFS (Depth First Search)

Beam Search Algorithm

beam width - n words - keep track of (n x no. of words) in vocabulary at each evaluation
- instantiate n copy of network
- if EOS - end of sentence

-1 word a Greedy Search

Length Normalization

Beam search → ① $\arg\max_y \prod_{t=1}^{T_y} P(y^{\langle t\rangle} | x, y^{\langle 1\rangle}, \dots, y^{\langle t-1\rangle})$ product → numerical underflows

① $\arg\max_y \sum_{t=1}^{T_y} \log P(y^{\langle t\rangle} | x, y^{\langle 1\rangle}, \dots, y^{\langle t-1\rangle})$ sum → numerical stable

shorter sentence is preferred

② Divide by T_y^{α} softer approach, $\alpha = 0.7$ → heuristic

log - monotonically increasing func

Error Analysis

Error
- Beam Search: $P(y^* | x) > P(\hat{y} | x)$
- RNN: $P(y^* | x) \leq P(\hat{y} | x)$

y^* - Human
$\hat{y}$ - Algorithm

Bleu Score

Bilingual evaluation understudy - take note of senior actor)

Evaluate machine translation → multiple equally good translation (human)

└ good - type of words generated - appear atleast one of human generated references

2002

$$\text{Precision} = \frac{\text{present in reference}}{\text{\# words in machine translated output}}$$

repeated words → clip numerator → max # of words present in reference

Modified Precision

words in Isolation - Unigram
- Bigram
- n-gram

$$P_n = \frac{\sum_{n\text{-gram} \in \hat{y}} \text{Count}_{clip}(n\text{-gram})}{\sum_{n\text{-gram} \in \hat{y}} \text{Count}(n\text{-gram})}$$

Combined Bleu Score

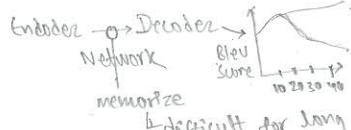
$$= B.P. * \exp\left(\frac{1}{N} \sum_{n=1}^{N} P_n\right)$$

└ brevity penalty - penalize short sentence

$$B.P. = \begin{cases} 1 & \text{if (machine translation} > \text{reference output) length} \\ \exp(1 - MT\ len/ref\ len) & \text{otherwise} \end{cases}$$

Attention Model

2014



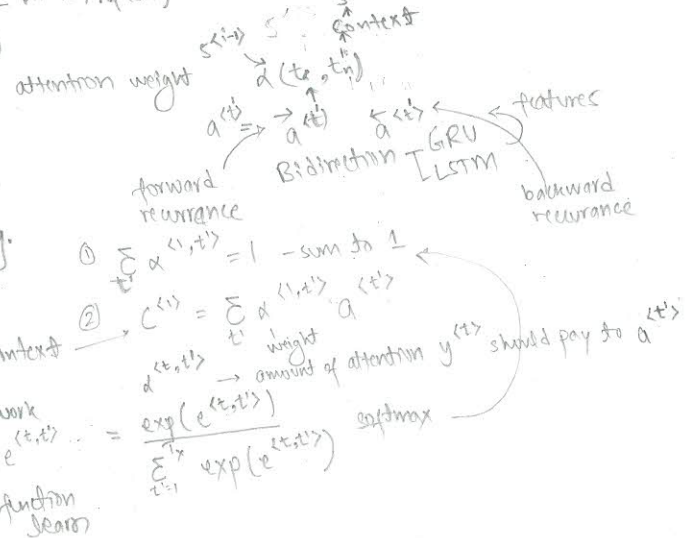
└ difficult for long sentence

Run time - Quadratic Cost

└ input & output sentence is not long.

Application - Image captioning

Visualization of attention weights



① $\sum_{t'} \alpha^{\langle t,t'\rangle} = 1$ - sum to 1

② context $C^{\langle t\rangle} = \sum_{t'} \alpha^{\langle t,t'\rangle} a^{\langle t'\rangle}$ weight

$\alpha^{\langle t,t'\rangle}$ → amount of attention $y^{\langle t\rangle}$ should pay to $a^{\langle t'\rangle}$

$$e^{\langle t,t'\rangle} = \frac{\exp(e^{\langle t,t'\rangle})}{\sum_{t'=1}^{T_x} \exp(e^{\langle t,t'\rangle})}$$

Speech Recognition

audio clip ⟶ transcript

Δ in air pressure against time

ear → intensity for different frequency over time

Preprocessing ⟶ Spectrogram / filter bank output

Phonemes / Hand engineered } features

Neural Network ⟵ large dataset

- Attention Model
- CTC cost - 2006
  - Connectionist temporal classification

input size ↑ (duration × hertz) >> # output

- collapse repeated characters not seperated by blank

Trigger Word Detection

Key word → wake up device

audio clip → target level y

- compute spectogram feature
- right after trigger word – imbalance training set
  - repeat 1 for few period