Approaches to Practical Applications – Recurrent Neural Networks and More

Outline

- Fields where deep learning is active
- The difficulties of deep learning
- The approaches to maximizing deep learning possibilities and abilities
- Summary

Active Fields of Deep Learning

- Huge investment among Google, Facebook, Microsoft, and IBM
- Focus fields
  - Image recognition
  - Natural language processing, NLP

Image Recognition

- Most frequently incorporated with deep learning
- Started by Prof. Hinton
  - Lowest error rates ever in an image recognition competition
- Google utilizes deep learning to:
  - Auto-generate thumbnails for YouTube
  - Auto-tag and search photos in Google Photos

Image Recognition

- Mainly applied to image tagging or categorizing and robotics
- Deep learning is more suited to image processing
  - An error rate of MNIST image classification is recorded at 0.21 percent with a deep learning algorithm (http://cs.nyu.edu/~wanli/dropc/)
  - Better than human (http://arxiv.org/pdf/0710.2231v1.pdf)

Why?

- In deep neural networks, many layers are stacked and features are extracted from training data step by step at each layer
  - Image data is featured as a layered structure
  - When you look at images, you will unconsciously catch brief features first and then look into a more detailed feature
- More improvement needed
  - Understand images and their contents

Deep Structure for Image

- Local receptive fields substituted with kernels of convolutional layers were introduced to avoid networks becoming too dense
- Downsampling methods such as max-pooling were invented to avoid the overreaction of networks towards a gap of image location
- Still we can't build omnipotent models
  - No Free Lunch Theorem (NFLT) for optimization

Natural Language Processing

- Become the most active going forward
  - Not going so good like image recognition
- Google utilizes deep learning to:
  - Voice search, voice recognition
  - Google translation
- IBM Watson (Watson API)
  - Cognitive computing system that understands and learns natural language
  - Supports human decision-making
  - Extracts keywords and entities from tons of documents
  - Has functions to label documents

Feed-forward Neural Networks for NLP

- Fundamental problem of NLP
  - "to predict the next word given a specific word or words"
- Simple to describe, too many difficulties:
  - The length of each sentence is not fixed but variable, and the number of words is astronomical
  - There can be unforeseen problems such as misspelled words, acronyms, and so on
  - Sentences are sequential data, and so contain temporal information

NN Problem

- The number of neurons in each layer including the input layer needs to be fixed in advance
- The networks need to be the same size for all the sample data
- NLP: length of the input data is not fixed and can vary a lot
  - N-gram method

N-gram

- Word w and history h

$$P\left(w_1^n\right)=P\left(w_1\right)P\left(w_2 \mid w_3\right)P\left(w_3 \mid w_1^2\right)\ldots P\left(w_n \mid w_1^{n-1}\right)$$

$$=\prod_{k=1}^{n} P\left(w_k \mid w_1^{k-1}\right)$$

  - Problem: we have no way of calculating the exact probability of a word following a long sequence of preceding words $P\left(w_n \mid w_{n-1}\right)$

2 Approaches to Solve

- Original N-gram model
- Neural networks model based on N-gram

Original N-gram model

- Approximate the history with the last N words
  - Instead of computing the probability of a word given its whole history

$$P\left(w_1^n\right) = \prod_{k=1}^{n} P\left(w_k \mid w_1^{k-1}\right)$$

$$\approx \prod_{k=1}^{n} P\left(w_k \mid w_{k-1}\right)$$

$$P\left(w_n \mid w_1^{n-1}\right) \approx P\left(w_n \mid w_{n-N+1}^{n-1}\right)$$

$$P\left(w_1^n\right) \approx \prod_{k=1}^{n} P\left(w_k \mid w_{k-N+1}^{k-1}\right)$$

  - these approximations with N-gram are based on the probabilistic model called the Markov model

Smart Computing 智慧計算實驗室 SAIC

Estimate N-gram Probabilities

- Maximum likelihood estimation (MLE)

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{\sum_w C(w_{n-1}w)} \qquad P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

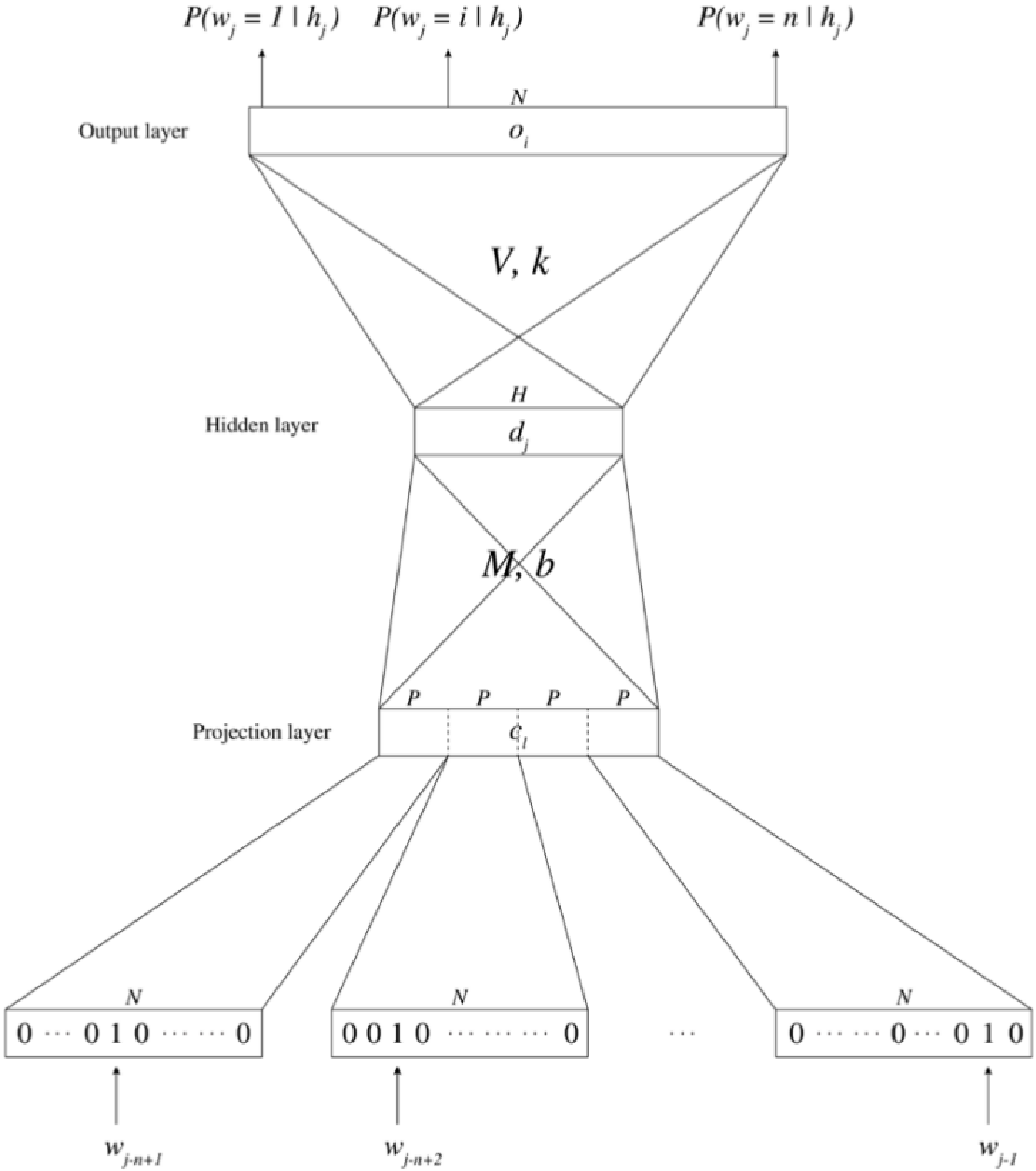
- Generalize MLE for N-gram

$$P(w_n \mid w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n-N+1}^{n-1})}$$

Neural Network Language Model

- Neural Network Language Model (NNLM)
  - neural network models predict the conditional probability of a word wj given a specific history, hj
  - http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

Smart Computing 智慧計算實驗室 SAIC



Smart Computing 智慧計算實驗室 SAIC

NNLM

- 1-of-N coding
  - N is the size of the vocabulary, and each word in the vocabulary is an N-dimensional vector where only the index of the word is set to 1 and all the other indices to 0
- The inputs of NLMM are the indices of the n − 1 previous words $h_j = w_{j-n+1}^{j-1}$
- Each word is mapped to the projection layer, for continuous space representation

Smart Computing 智慧計算實驗室 SAIC

NNLM

- This linear projection (activation) from a discrete to a continuous space is basically a look-up table with N × P entries, where P denotes the feature dimension
- The activation

$$d_j = h\left(\sum_{l=1}^{(n-1).P} m_{jl} c_l + b_j\right)$$

NNLM

- Output units

$$o_i = \sum_j v_{ij} d_j + k_i$$

- Probability of a word i

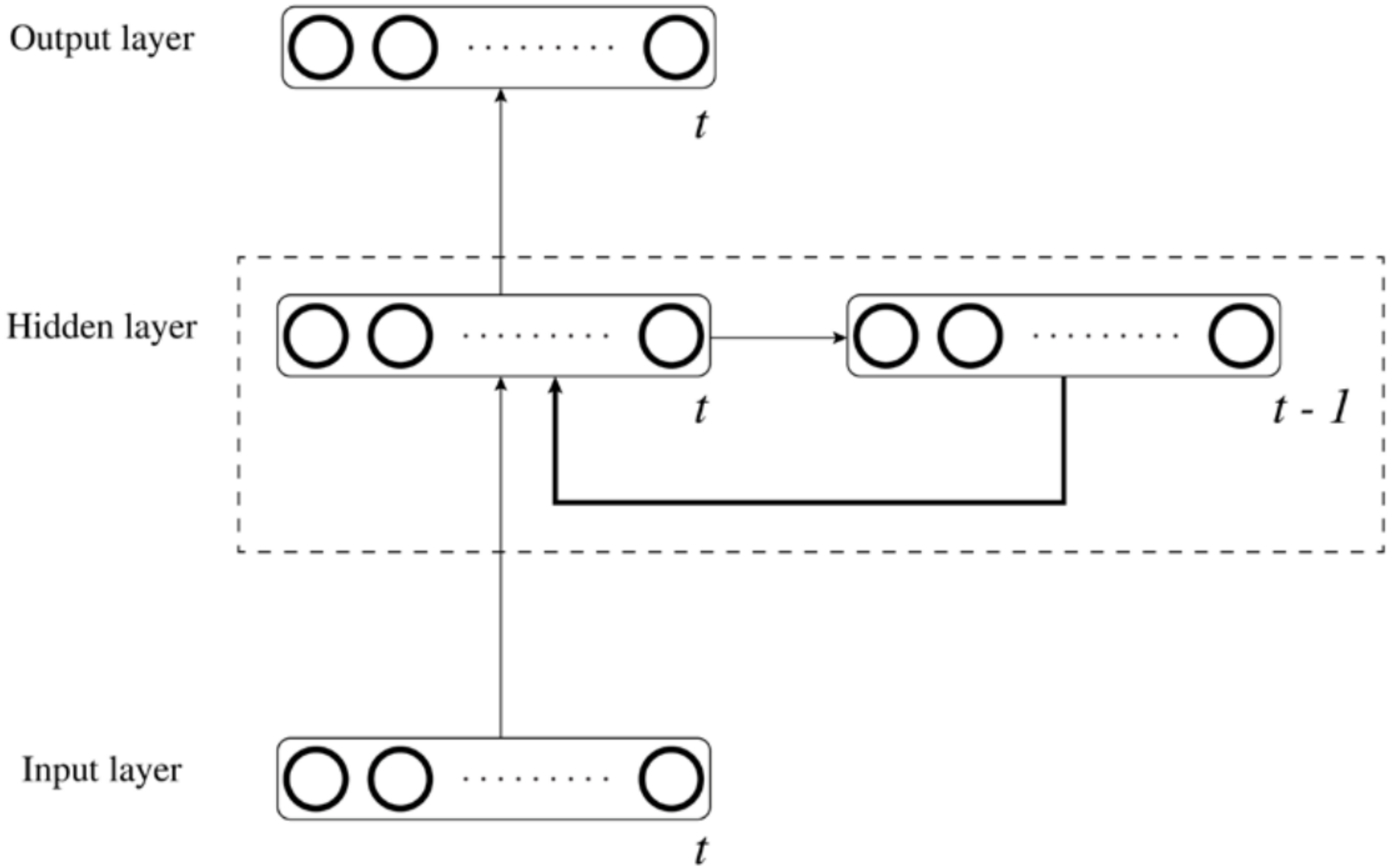
$$P\left(w_j = i \mid h_j\right) = \frac{\exp\left(o_i\right)}{\sum_{l=1}^{N} \exp\left(o_i\right)}$$

Deep Learning for NLP

- Problem of NNLM: still have a context?
  - Common to all the other fields that have time sequential data
  - Precipitation, stock prices, yearly crop of potatoes, movies, …
- How would it be possible to let neural networks be trained with time sequential data?
  - Preserve the context of data: recurrent neural network (RNN)

Recurrent Neural Networks

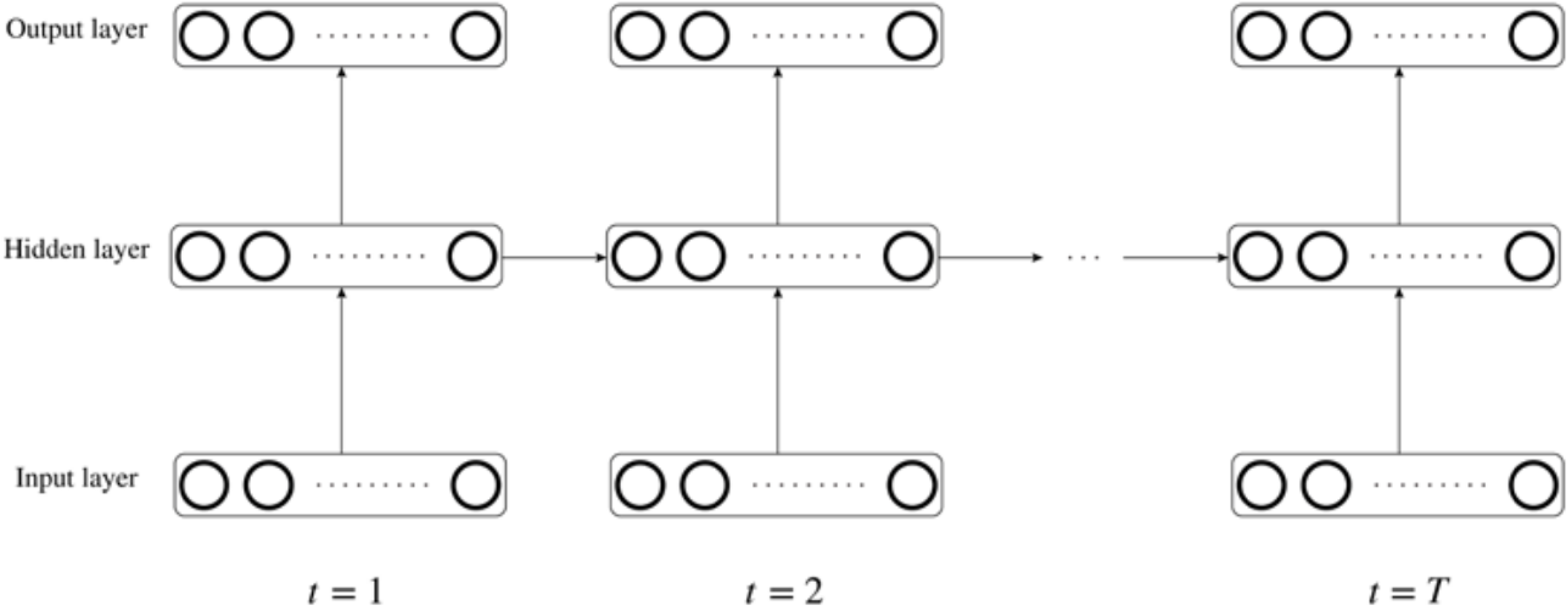
- Graphical model



Recurrent Neural Networks

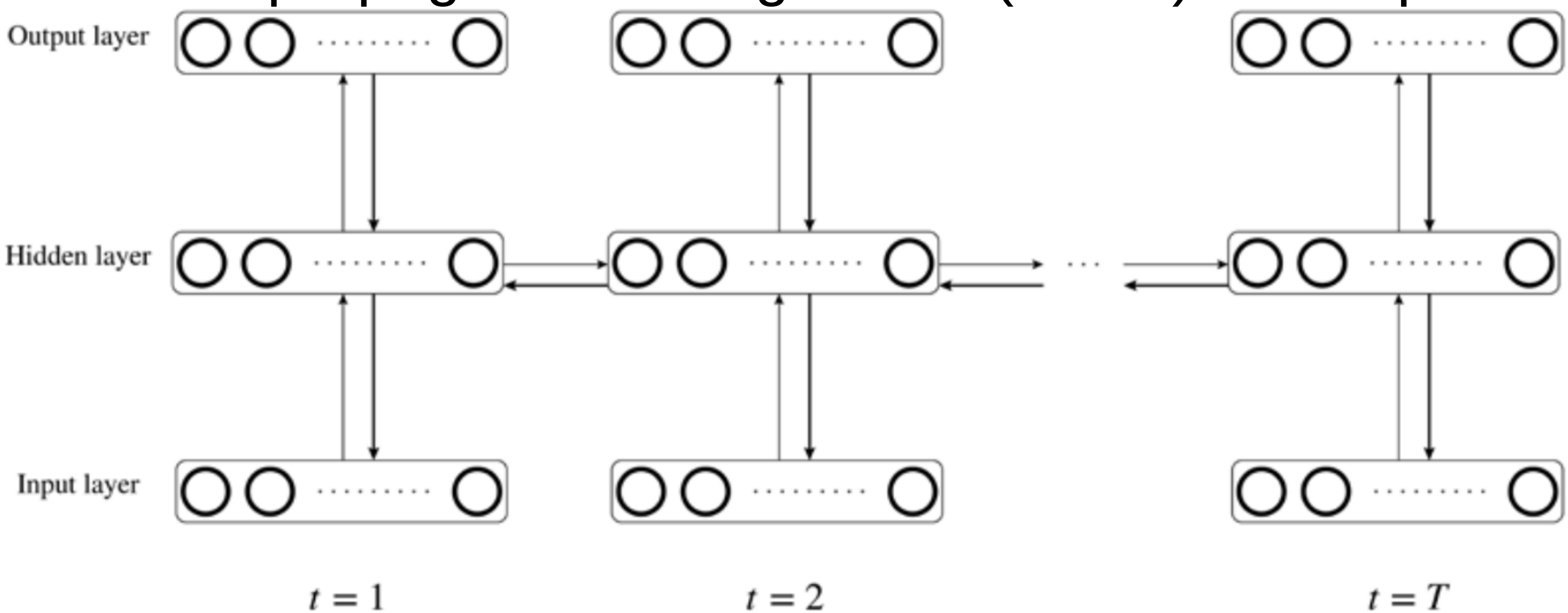
- RNN has connections between hidden layers with respect to time
  - The input at time t is activated in the hidden layer at time t , preserved in the hidden layer, and then propagated to the hidden layer at time t + 1 with the input at time t + 1
  - Enables the networks to contain the states of past data and reflect them

Unfold RNN



Training RNN

- Train RNN model using the backpropagation algorithm
  - Backpropagation through Time (BPTT) technique

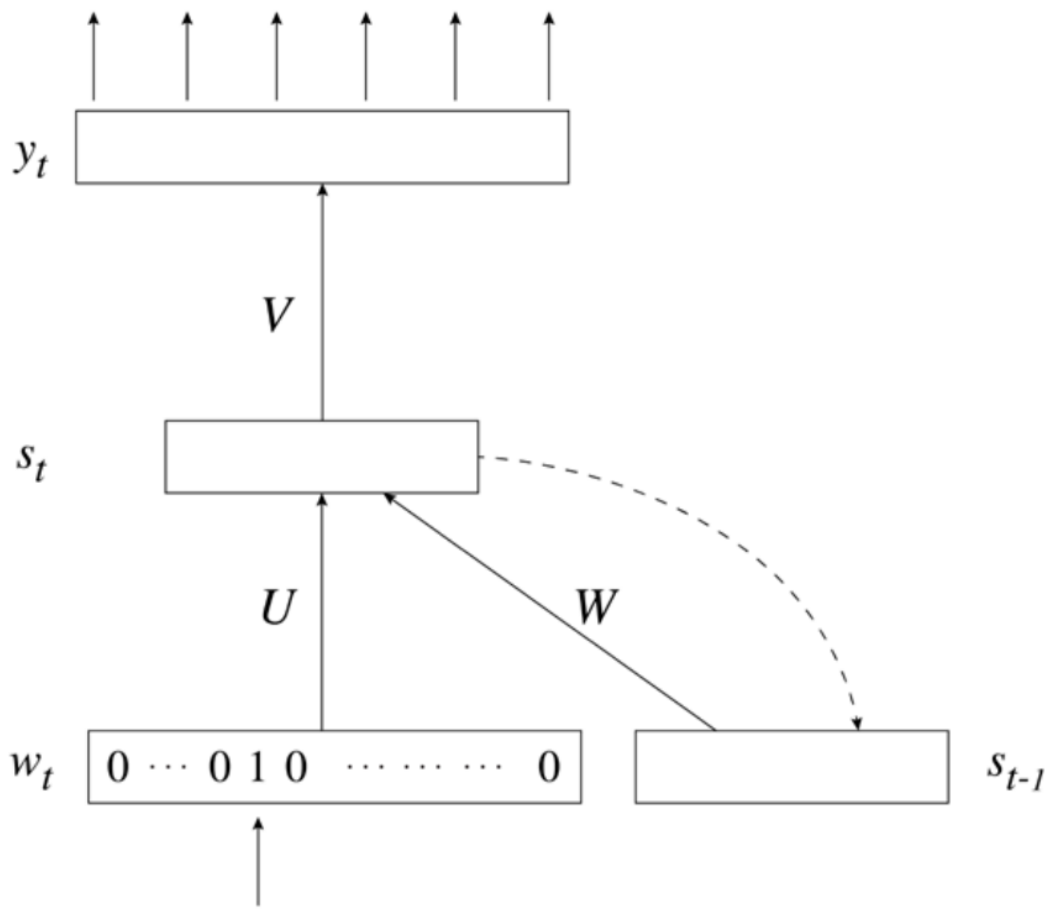


Training RNN

- Theoretically, the network at each time step should consider the whole sequence up to then
- Practically, time windows with a certain length are often applied to the model to:
  - Make the calculation less complicated
  - Prevent the vanishing gradient problem
  - The exploding gradient problem

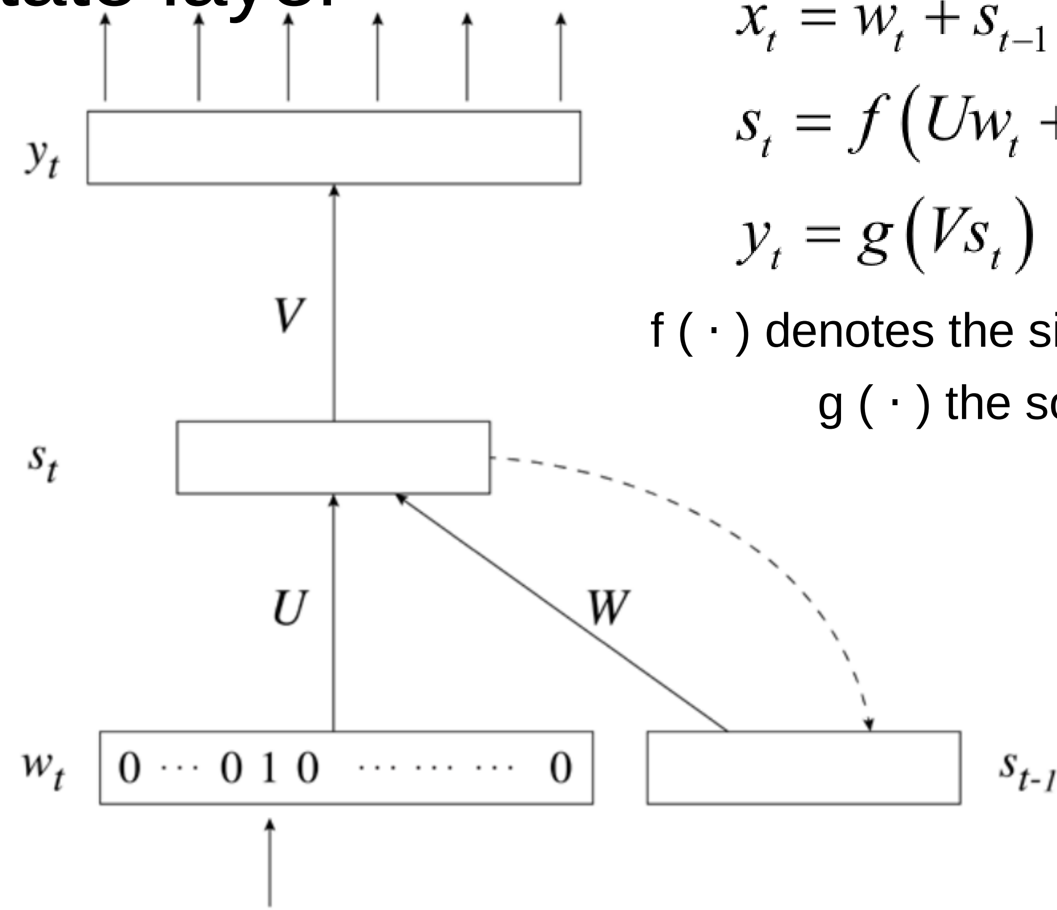
Recurrent Neural Network Language Model (RNNLM)

- Introduced by Mikolov et al. (http://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf)



RNNLM

- Input layer x, hidden layer s, output layer y
- Hidden layer is also often called the context layer or the state layer



$$x_t = w_t + s_{t-1}$$

$$s_t = f\left(Uw_t + Ws_{t-1}\right)$$

$$y_t = g\left(Vs_t\right)$$

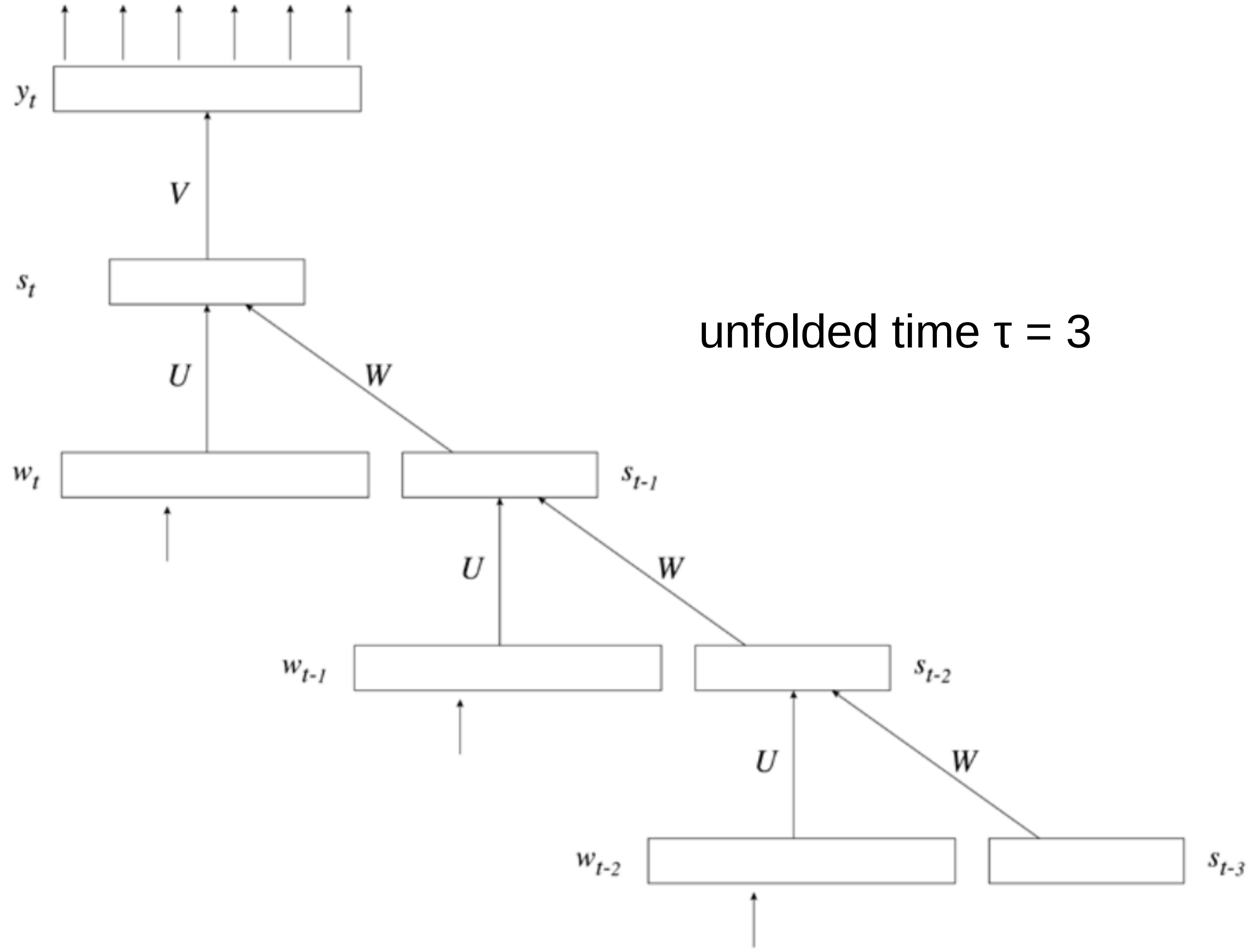
f (·) denotes the sigmoid function

g (·) the softmax function

Smart Computing 智慧計算實驗室 SAIC

Training RNNLM: Truncated BPTT

- Often truncate the time length
- Algorithm: truncated BPTT



unfolded time τ = 3

Training RNNLM

- d_t is the label vector of the output, the error:

$$\delta_t^{out} = d_t - y_t \qquad \delta_t^{hidden} = d\left(\left(\delta_t^{out}\right)^T V, t\right)$$

- Unfolding time t:

$$d(x,t) = x s_t (1 - s_t)$$

$$\delta_{t-\tau-1}^{hidden} = d\left(\left(\delta_{t-\tau}^{out}\right)^T V, t-\tau-1\right)$$

Training RNNLM

- Learning rate α, U maps each word to latent space

$$V_{t+1} = V_t + s_t \left(\delta_t^{out}\right)^T \alpha$$

$$U_{t+1} = U_t + \sum_{\tau=0}^{T} w_{t-\tau} \left(\delta_{t-\tau}^{hidden}\right)^T \alpha$$

$$W_{t+1} = W_t + \sum_{\tau=0}^{T} w_{t-\tau-1} \left(\delta_{t-\tau}^{hidden}\right)^T \alpha$$

- Mapped word vectors contain the meaning of the words
  - "king" – "man" + "woman" would return "queen"

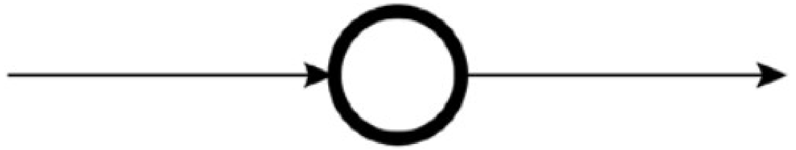
Smart Computing 智慧計算實驗室 SAIC

Long Short Term Memory Networks, LSTM

- Training with the standard RNN requires the truncated BPTT
  - Can BPTT really train the model enough to reflect the whole context?
- Solve the long-term dependency problem?
  - Long short term memory (LSTM) network

LSTM

- How we can store and tell past information in the network?
  - Hidden layer unit



  - Memorize the past information within the neuron

    - The neuron added here has linear activation and its value is often set to 1 => constant error carousel (CEC)
    - The error stays in the neuron like a carousel and won't vanish
    - CEC works as a storage cell and stores past inputs

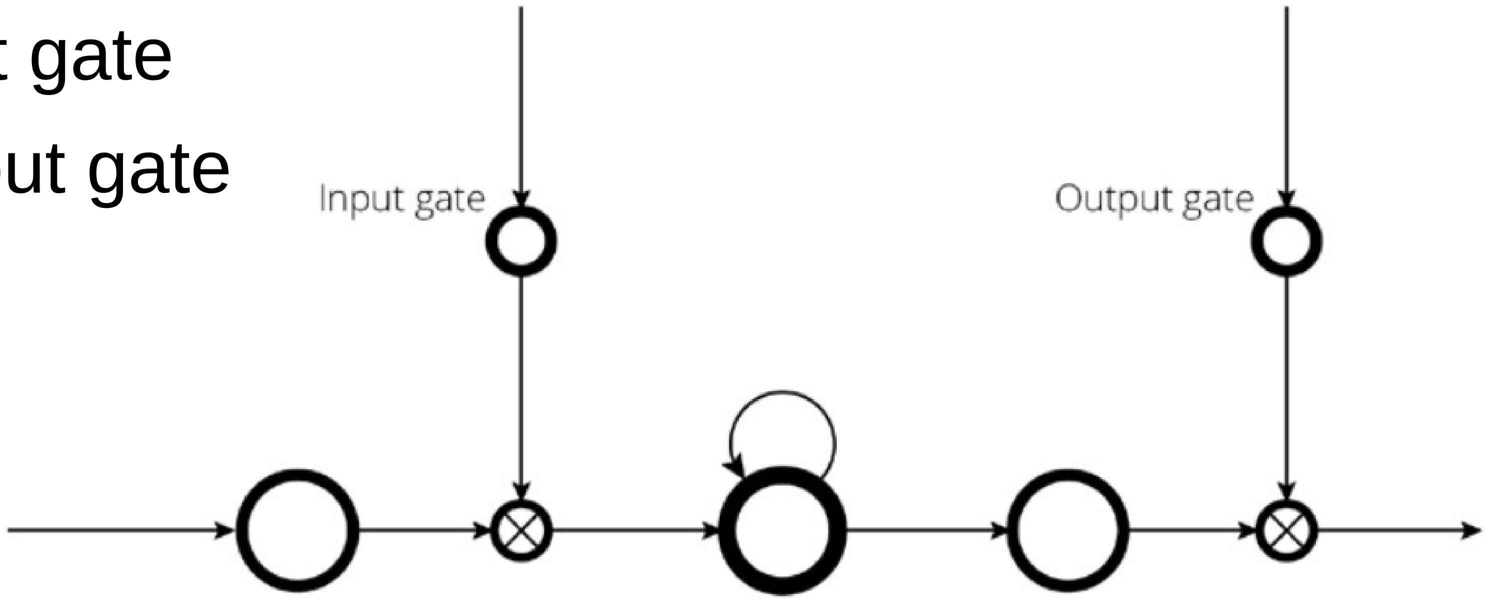
Smart Computing 智慧計算實驗室 SAIC

Constant Error Carousel, CEC

- Solves the gradient vanishing problem
- Another problems: all data propagated through is stocked in the neuron, it probably stores noise data as well
  - Input weight conflicts
  - Output weight conflicts
  - Require: controls the propagation of inputs and outputs

CEC

- Solution: putting units that act like "gates" before and behind the CEC
  - Input gate
  - Output gate

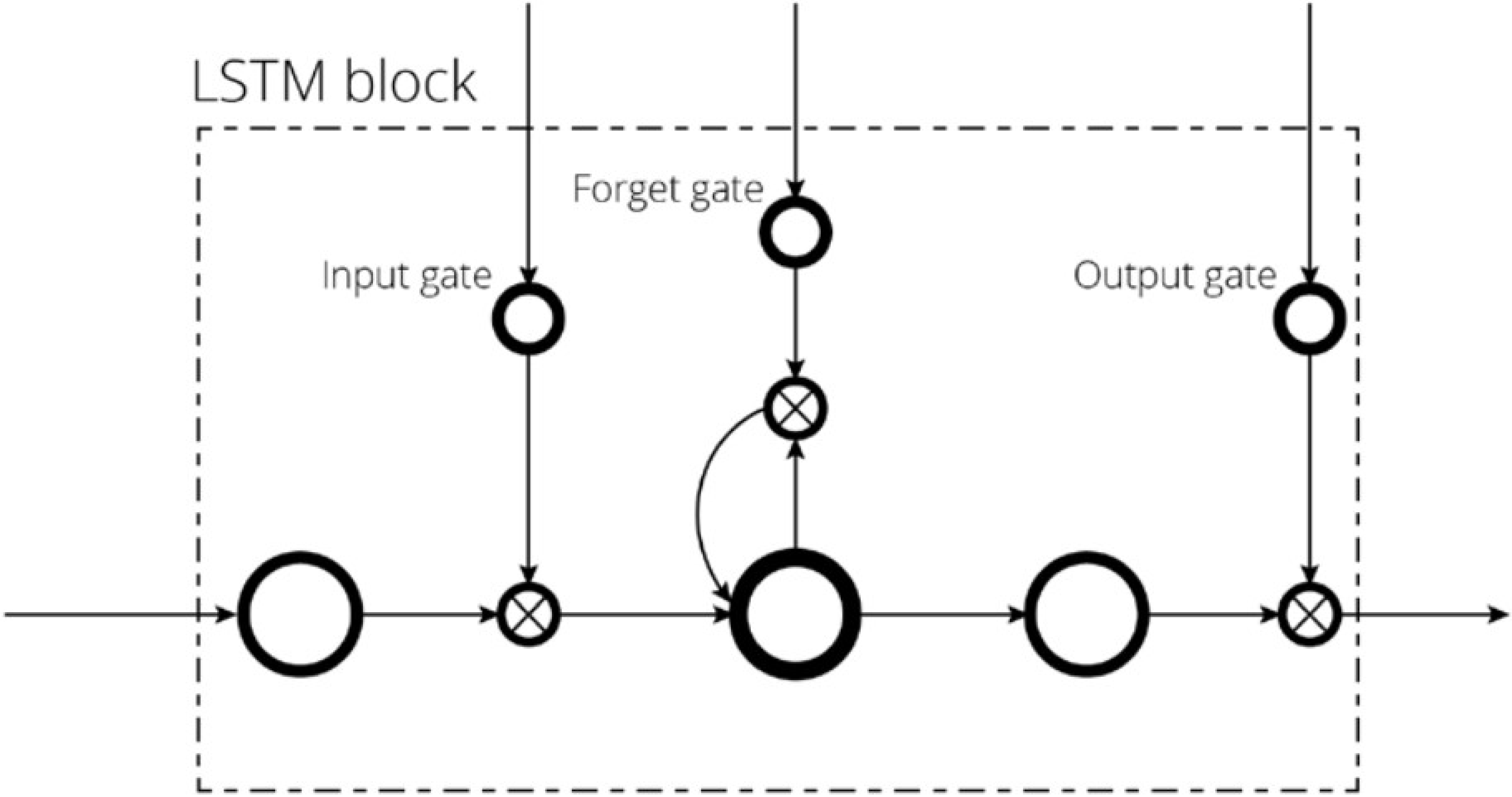


Ideally, the gate should return the discrete value of 0 or 1 corresponding to the input, 0 when the gate is closed and 1 when open, because it is a gate, but programmatically, the gate is set to return the value in the range of 0 to 1 so that it can be well trained with BPTT.

CEC

- Memories stored in the CEC can't be refreshed easily in a few steps
  - Forget gate
  - The value preserved in the CEC is overridden with a new memory when the value of the gate takes a 0 or close to it
- LSTM memory block
  - With input, output, forget gates
  - Block, not a single neuron

LSTM Memory Block

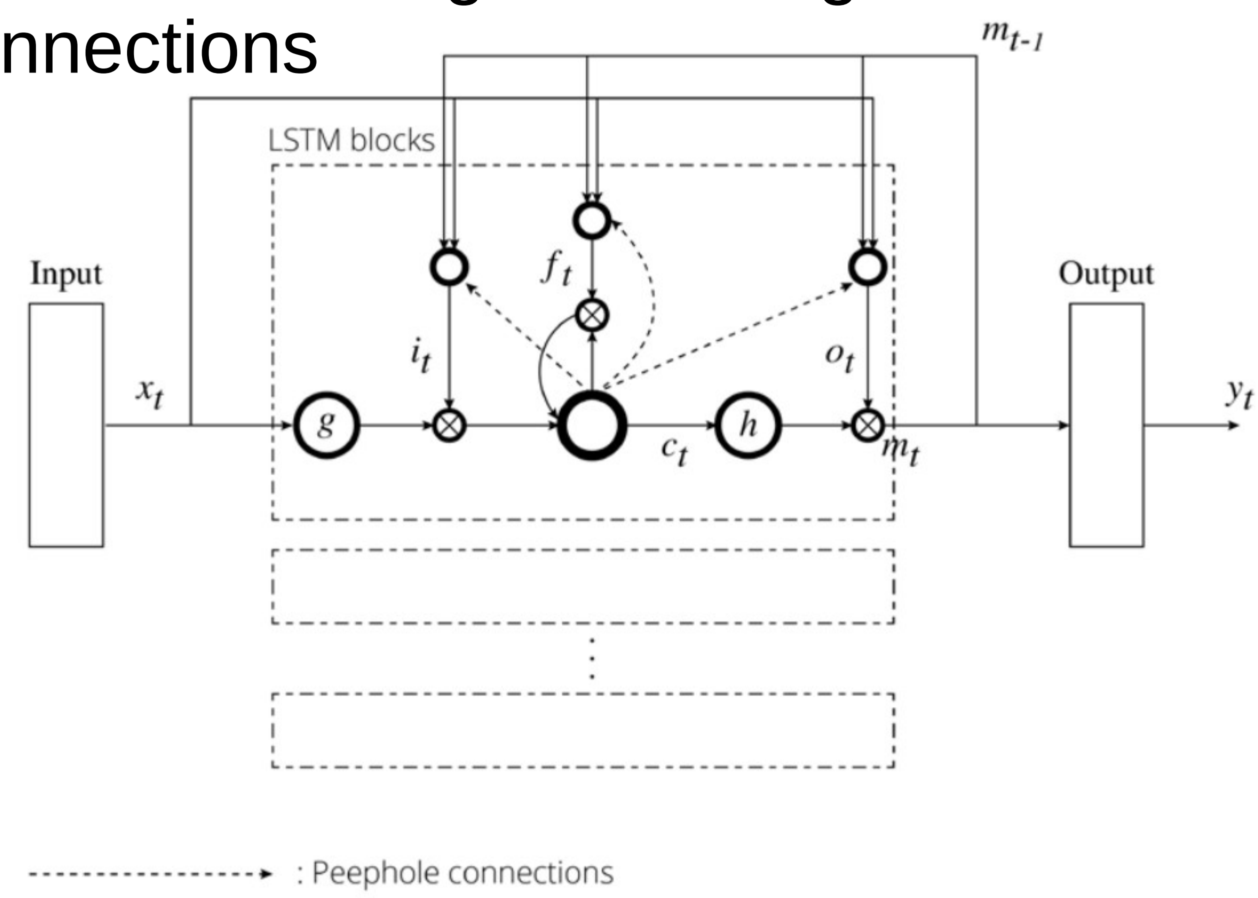


LSTM Memory Block

- Each gate receives connections from the input units and the outputs of all the units in LSTM
- No direct connection from the CEC
  - Unable to see the true hidden state of the network: output of a block depends on the output gate
  - If the output gate is closed, none of the gates can access the CEC and it is devoid of essential information => performance problem!

LSTM Memory Block

- Peephole connections
  - Standard weighted connections, no errors are backpropagated from the gates through the peephole connections



$$i_t = \sigma\left(W_{ix}x_t + W_{im}m_{t-1} + W_{ic}c_{t-1} + b_i\right)$$

$$f_t = \sigma\left(W_{fx}x_t + W_{fm}m_{t-1} + W_{fc}c_{t-1} + b_f\right)$$

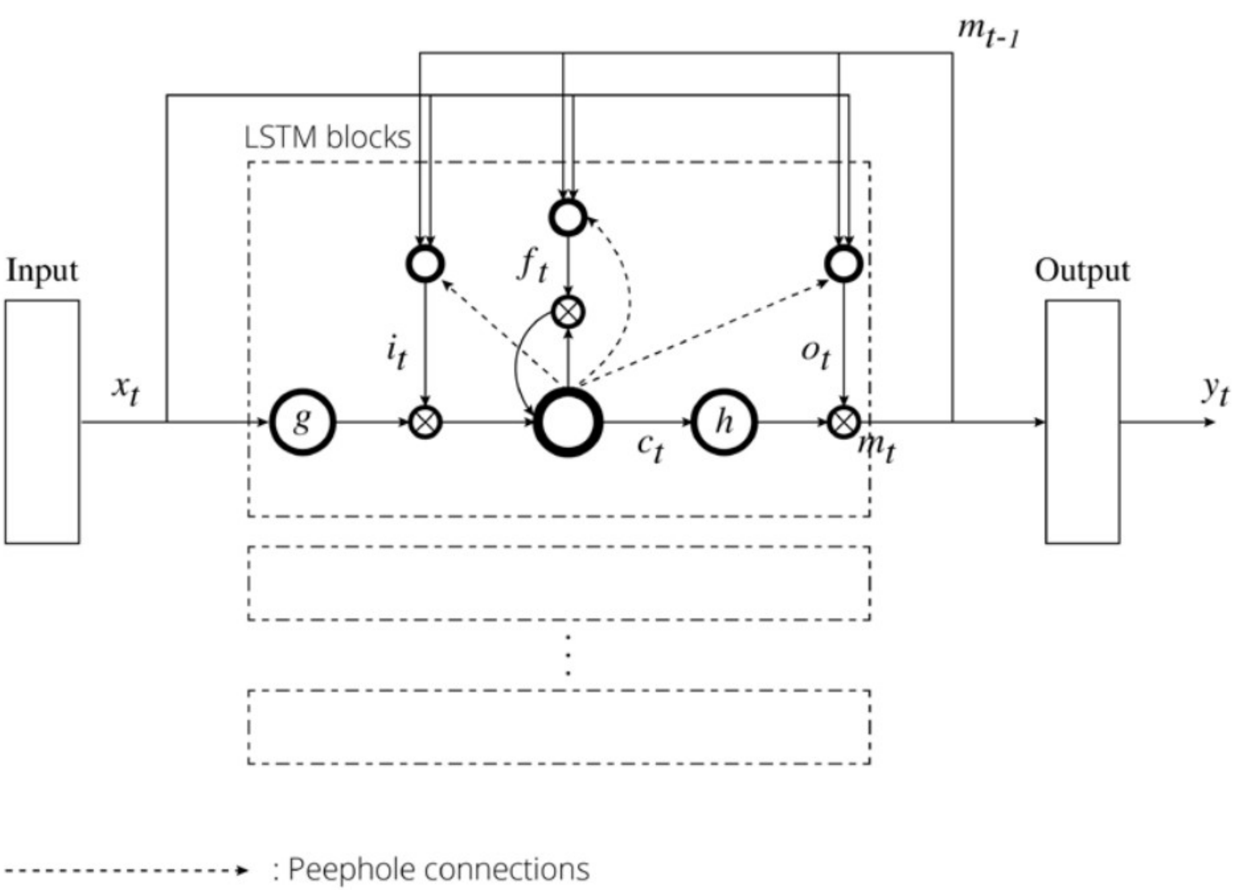
$$c_t = f_t \odot c_{t-1} + i_t \odot g\left(W_{cx}x_t + W_{cm}m_{t-1} + b_c\right)$$

$$o_t = \sigma\left(W_{ox}x_t + W_{om}m_{t-1} + W_{oc}c_t + b_o\right)$$

$$m_t = o_t \odot h\left(c_t\right)$$

$$y_t = s\left(W_{ym}m_t + b_y\right)$$

σ denotes the sigmoid function, and s (·) the softmax function. $\odot$ is the element-wise product of the vectors



LSTM References

- Sequence to Sequence Learning with Neural Networks (Sutskever et al., http://arxiv.org/pdf/1409.3215v3.pdf)
- Grid Long Short-Term Memory (Kalchbrenner et al., http://arxiv.org/pdf/1507.01526v1.pdf)
- Show, Attend and Tell: Neural Image Caption Generation with Visual Attention (Xu et al., http://arxiv.org/pdf/1502.03044v2.pdf)

Outline

- Fields where deep learning is active
- The difficulties of deep learning
- The approaches to maximizing deep learning possibilities and abilities
- Summary

Difficulties of Deep Learning

- How much deep learning is utilized in other fields?
  - Few
- Why?
  - Too many model parameters (hyper parameters)
  - Go through more trial and error to get high precision
- Great performance is supported by steady parameter-tuning

Difficulties of Deep Learning

- Deep learning often fails to train and classify data from simple problems
  - Weights can't be well optimized
  - Data quantities
- Deep learning is still far from the true AI

Outline

- Fields where deep learning is active
- The difficulties of deep learning
- The approaches to maximizing deep learning possibilities and abilities
- Summary

Approaches to Apply

- 3 categories
  - Field-oriented approach
  - Breakdown-oriented approach
  - Output-oriented approach: explores new ways of how we express the output with deep learning

Field-oriented Approach

- This approach doesn't require new techniques or algorithms
- Medicine
  - Tumors or cancers are detected on scanned images => image recognition
- Automobiles
  - Surroundings of running cars are image sequences and text
  - Self-driving cars

Field-oriented Approach

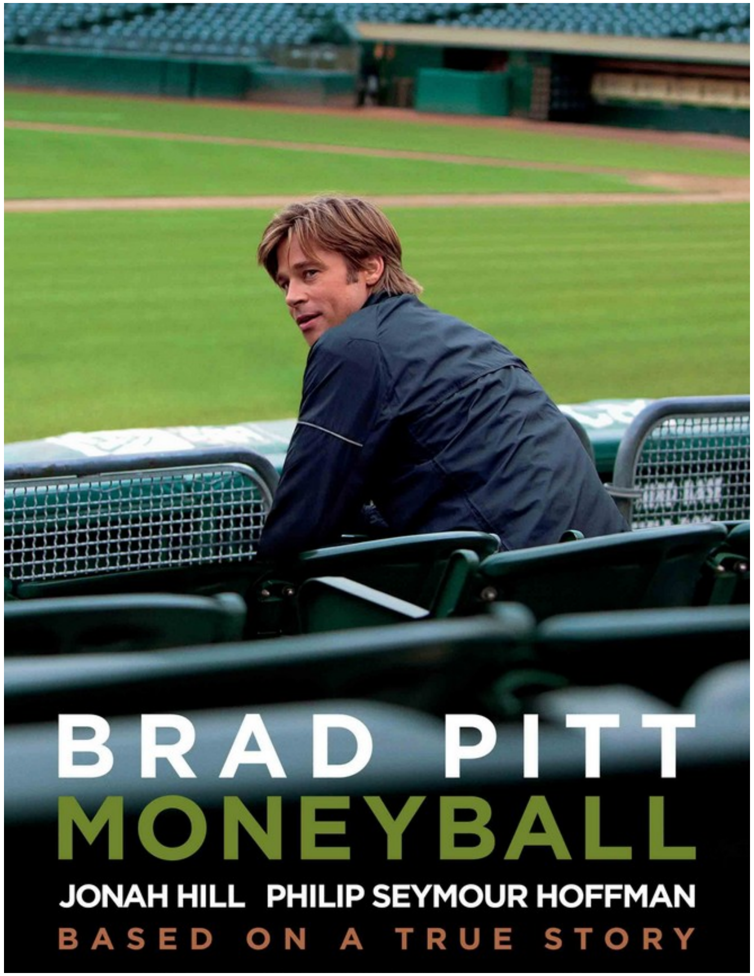
– George Hotz, the first person to hack the iPhone, built a self-driving car in his garage (http://www.bloomberg.com/features/2015-george-hotz-self-driving-car/)

Field-oriented Approach

- Advert technologies
  - Use user-behavior-based indicators, such as page view (PV), click through rate (CTR), and conversion rate (CVR), to estimate the effect of an ad
  - Use deep learning to analyze the actual content of an ad and autogenerate ads going forward
- Profession or practice
  - Doctor, lawyer, patent attorney, and accountant
  - With NLP's precision and accuracy gets higher

Field-oriented Approach

- Sports
  - Movie: Moneyball
  - increased the win percentage of the team by adopting a regression model in baseball



Smart Computing 智慧計算實驗室 SAIC

Breakdown-oriented Approach

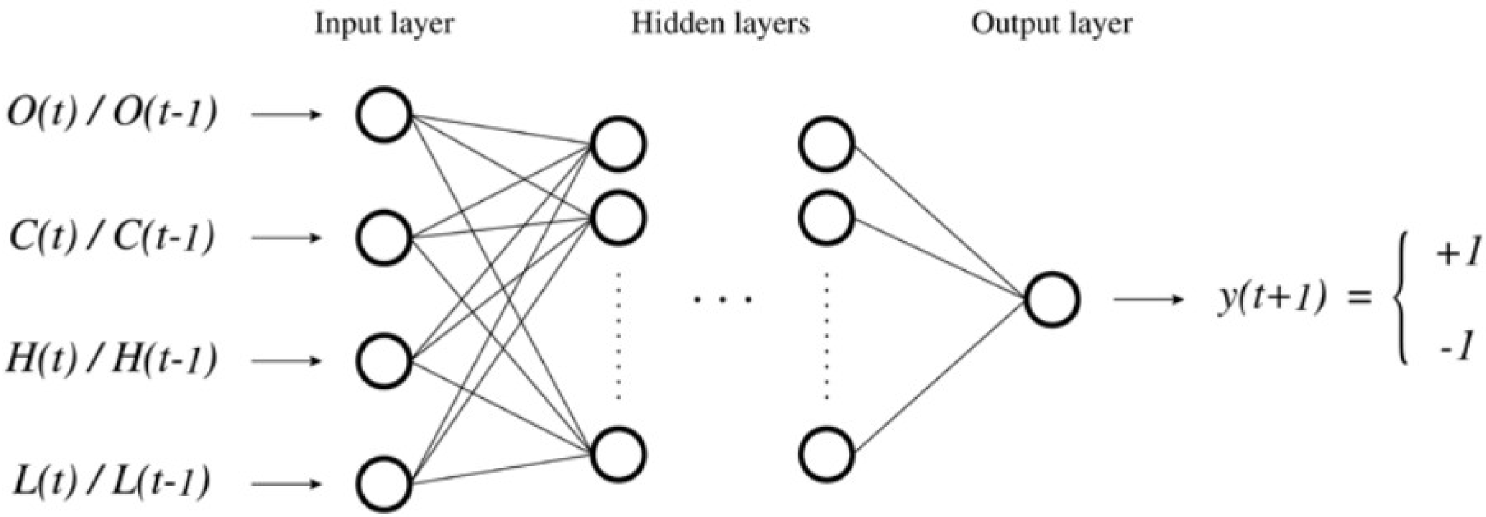
- Similar to the approach considered in traditional machine learning algorithms
- Feature engineering is the key to improving precision in machine learning
  - Engineering under the constraints of a machine learning model
    - E.g. make inputs discrete or continuous
  - Feature engineering to increase precision by machine learning
    - Rely on the sense of a researcher

Breakdown-oriented Approach

- Deep learning doesn't have to focus on the second
- First one is the important part
  - For example, it's difficult to predict stock prices using deep learning
    - Stock prices are volatile
    - Difficult to define inputs
    - How to apply an output value
  - Enabling deep learning to handle these inputs and outputs is also said to be feature engineering in the wider sense

Breakdown-oriented Approach

- Simplified stock price prediction: close price up or down



Class	Description
Class 1	Up more than 3 percent from the closing price
Class 2	Up more than 1~3 percent from the closing price
Class 3	Up more than 0~1 percent from the closing price
Class 4	Down more than 0~-1 percent from the closing price
Class 5	Down more than -1~-3 percent from the closing price
Class 6	Down more than -3 percent from the closing price

Breakdown-oriented Approach

- Feature engineering for models
  - designing inputs or adjusting values to fit deep learning models
  - enabling classification by setting a limitation for the outputs
- Model engineering for features
  - Devising new neural network models or algorithms to solve problems in a focused field

Output-oriented Approach

- Gain the world's interest by thinking of ideas in creative fields
  - The world pay attention to what a machine can't do

it might be better to emphasize the point that machines make mistakes

Outline

- Fields where deep learning is active
- The difficulties of deep learning
- The approaches to maximizing deep learning possibilities and abilities
- Summary

Summary

- Deep learning algorithms for practical applications: NLP
- Two new deep learning models: the RNN and LSTM networks
  - Training algorithm: BPTT
- Three approaches to make the best of the deep learning ability
  - Field-oriented, breakdown-oriented, output-oriented approach