# LANGUAGE MODELING FOR SPEECH RECOGNITION

HUNG-YI LEE 李宏毅

# Why Language Model?

- Language model (LM): Estimated the probability of token sequence
  - Token sequence: $Y = y_1, y_2, \ldots \ldots, y_n$
  - $P(y_1, y_2, \ldots \ldots, y_n)$

***HMM*** $$\mathrm{Y}^* = arg \max_{\mathrm{Y}} P(X|\mathrm{Y})P(\mathrm{Y})$$

LM is usually helpful when your model outputs text

***LAS*** $$\mathrm{Y}^* = arg \max_{\mathrm{Y}} \underbrace{P(\mathrm{Y}|X)}_{\text{Need paired data}} \; \underbrace{P(Y)}_{\text{Easy to collect}}$$

https://numberofwords.com/faq/how-many-words-are-in-a-1-minute-speech/

# Why we need LM?

$$Y^* = arg \max_{Y} \underbrace{P(Y|X)}_{\text{Need paired data}} \; \underbrace{P(Y)}_{\text{Easy to collect}}$$

**Words in Transcribed Audio**

12,500 hours transcribed audio

= 12,500 x 60 x 130 ≈ 一億!

(哈利波特全套約 100 萬個詞)

Moschitta had been credited in The Guinness Book of World Records as the World's Fastest Talker

Source of video: https://youtu.be/ExKCcndqK5c https://numberofwords.com/faq/how-many-words-are-in-a-1-minute-speech/

# Why we need LM?

$$Y^* = arg \max_{Y} \underbrace{P(Y|X)}_{\text{Need paired data}} \; \underbrace{P(Y)}_{\text{Easy to collect}}$$

**Words in Transcribed Audio**

12,500 hours transcribed audio

= 12,500 x 60 x 130 ≈ 一億!

(哈利波特全套約 100 萬個詞)

BERT:
https://youtu.be/UYPa347-DdE

**Just Words …**

BERT (一個巨大的 LM) 用了
30 億個以上的詞

# N-gram

P("wreck a nice beach")
=P(wreck|START)P(a|wreck)
P(nice|a)P(beach|nice)

- How to estimate $P(y_1, y_2, \ldots\ldots, y_n)$
- Collect a large amount of text data as training data
  - However, the token sequence $y_1, y_2, \ldots\ldots, y_n$ may not appear in the training data
- *N-gram language model*: $P(y_1, y_2, \ldots\ldots, y_n) = P(y_1|BOS)P(y_2|y_1) \ \ldots \ P(y_n|y_{n-1})$

2-gram

  - E.g. Estimate P(beach|nice) from training data

$$P(\text{beach}|\text{nice}) = \frac{C(nice\ beach)}{C(nice)}$$

Count of "nice beach"

Count of "nice"

  - It is easy to generalize to 3-gram, 4-gram ……

# Challenge of N-gram

- The estimated probability is not accurate.
  - Especially when we consider n-gram with large n
  - Because of data sparsity (many n-grams never appear in training data)

Training Data:

The dog ran ……
The cat jumped ……

P( jumped | the, dog ) = ~~0~~ 0.0001

P( ran | the, cat ) = ~~0~~ 0.0001

Give some small probability

This is called **language model smoothing**.

# Continuous LM

- Recommendation system

| |  |  |  |  |
|---|---|---|---|---|
| A | 5 | 5 | | 1 |
| B | 5 | 5? | 1 | |
| C | 5 | 5 | | 2 |
| D | 1 | | 4 | 4 |
| E | | 1 | 5 | 4 |

Matrix Factorization

Ref: https://youtu.be/iwh5o_M4BNU?t=4673

Borrowing the idea from recommendation system

# Continuous LM

| | dog $h^1$ | cat $h^2$ | …… | child |
|---|---|---|---|---|
| ran $v^1$ | 2 $n_{11}$ | 3 | | 1 |
| jumped $v^2$ | 0 | 2 $n_{22}$ | | 1 |
| cried $v^3$ | 0 | 0 | | 3 |
| laughed $v^4$ | 0 | 0 | | 3 |
| …… | | | | |

history

Vocabulary

Not observed

Count of "cat jumped"

$v^i$, $h^j$ are vectors to be learned

$$n_{12} = v^1 \cdot h^2$$
$$n_{21} = v^2 \cdot h^1 \; \ldots$$

Minimizing

$$L = \sum_{(i,j)} \left( v^i \cdot h^j - n_{ij} \right)^2$$

$v^i$, $h^j$ found by gradient descent

# Continuous LM

Borrowing the idea from recommendation system

history

| | dog $h^1$ | cat $h^2$ | …… | child |
|---|---|---|---|---|
| ran $v^1$ | 2 $n_{11}$ | 3 | | 1 |
| jumped $v^2$ | 0 | 2 $n_{22}$ | | 1 |
| cried $v^3$ | 0 | 0 | | 3 |
| laughed $v^4$ | 0 | 0 | | 3 |
| …… | | | | |

Vocabulary

Not observed

Count of "cat jumped"

History "dog" and "cat" can have similar vector $h^{dog}$ and $h^{cat}$

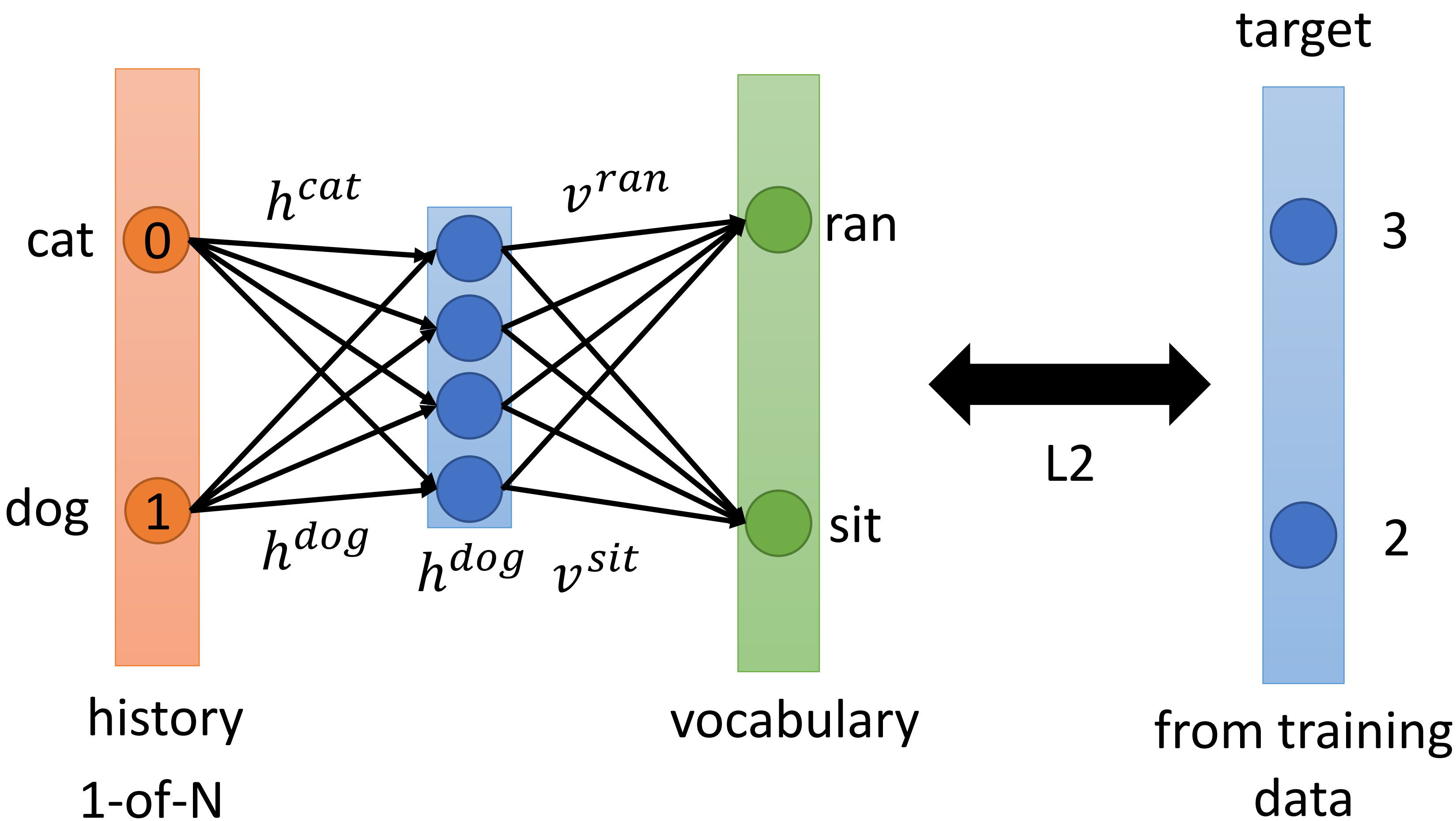
If $v^{jumped} \cdot h^{cat}$ is large, $v^{jumped} \cdot h^{dog}$ would be large accordingly.

Even if we have never seen "dog jumped …"

Smoothing is automatically done.

$$L = \sum_{(i,j)} \left(v^i \cdot h^j - n_{ij}\right)^2$$

# Continuous LM




Consider it as a NN ……

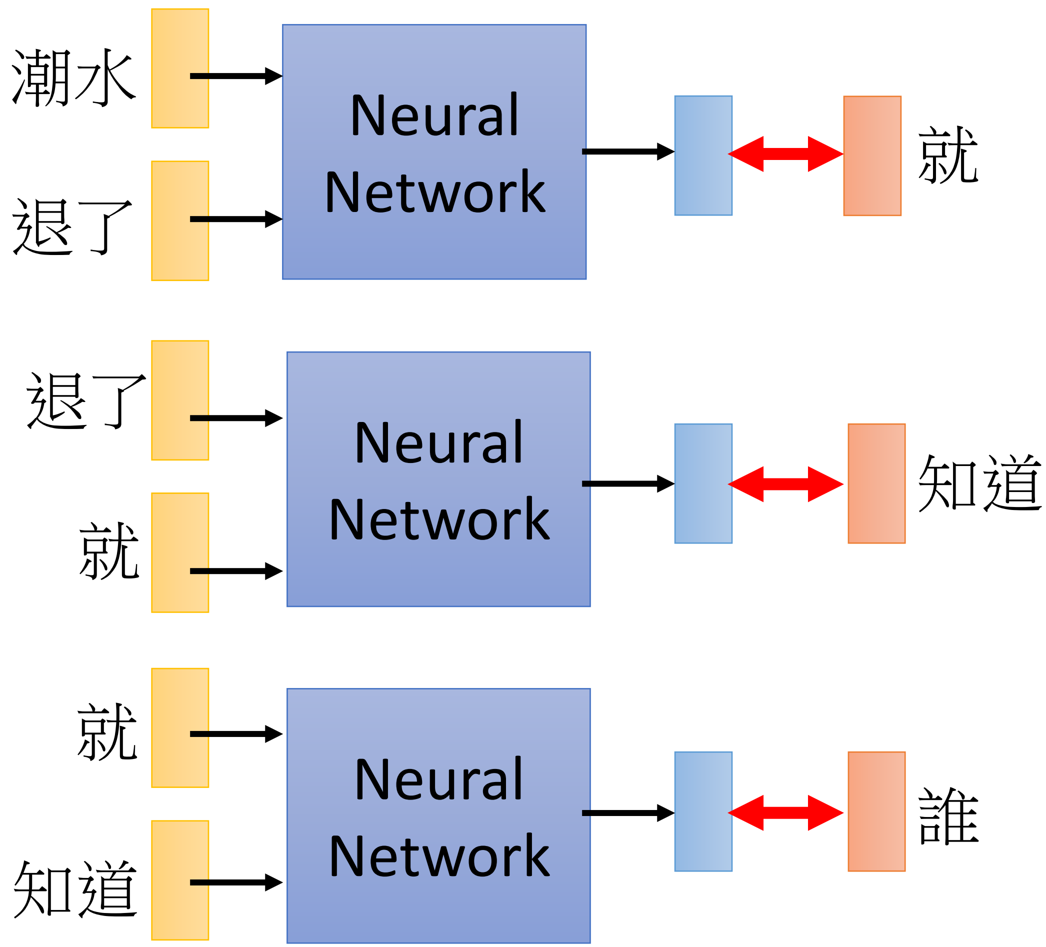
# NN-based LM

- Training:

Collect data:

潮水 退了 就 知道 誰 …
不爽 不要 買 …
公道價 八萬 一 …
………

**Learn to predict the next word**

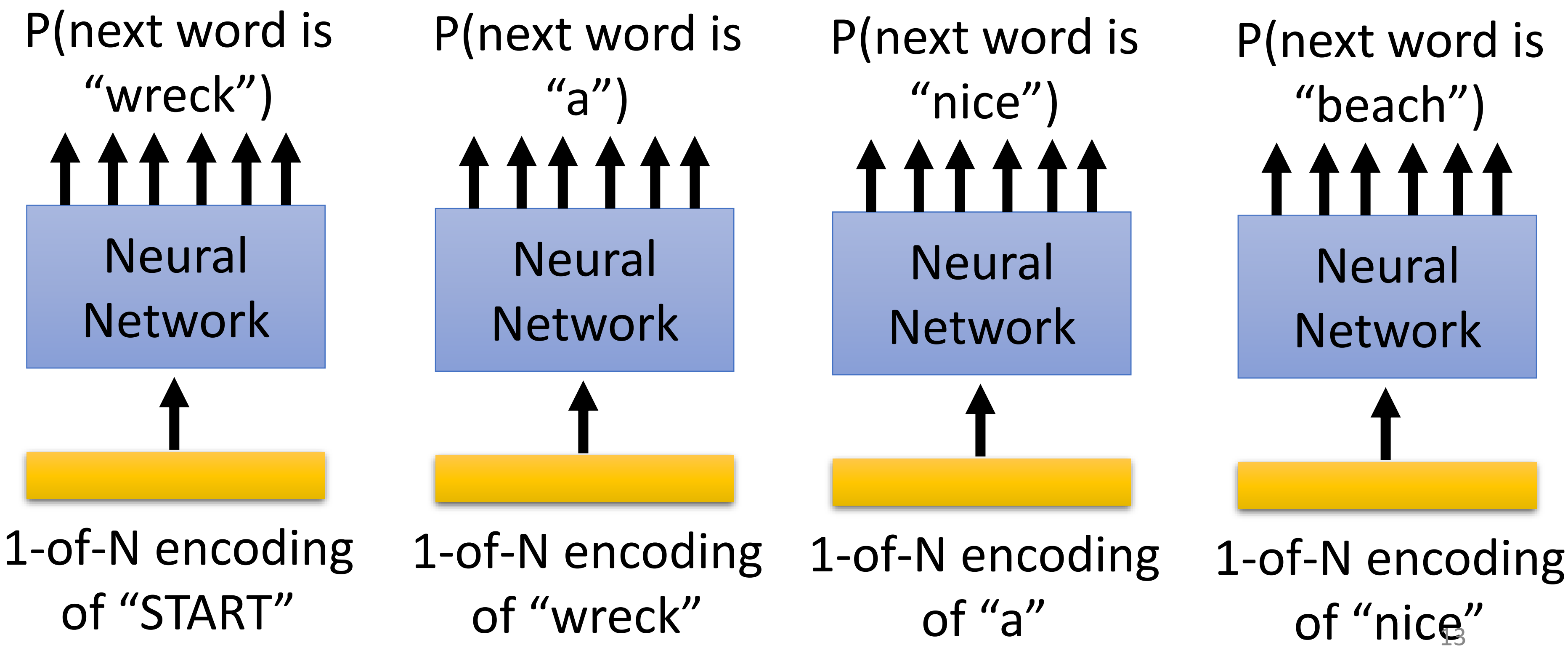


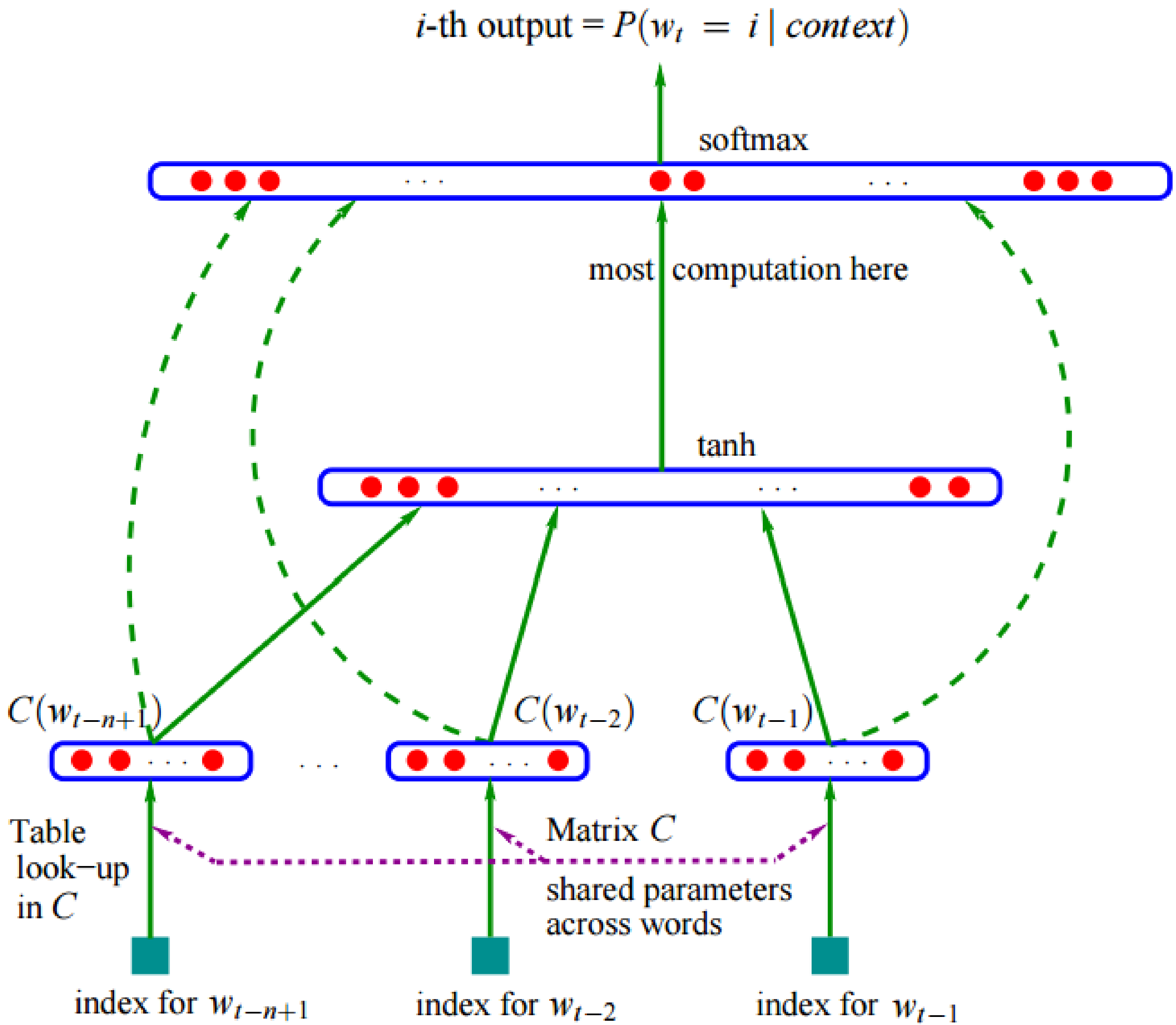
# NN-based LM

P("wreck a nice beach")

=P(wreck|START)P(a|wreck)P(nice|a)P(beach|nice)

P(b|a): the probability of NN predicting the next word.



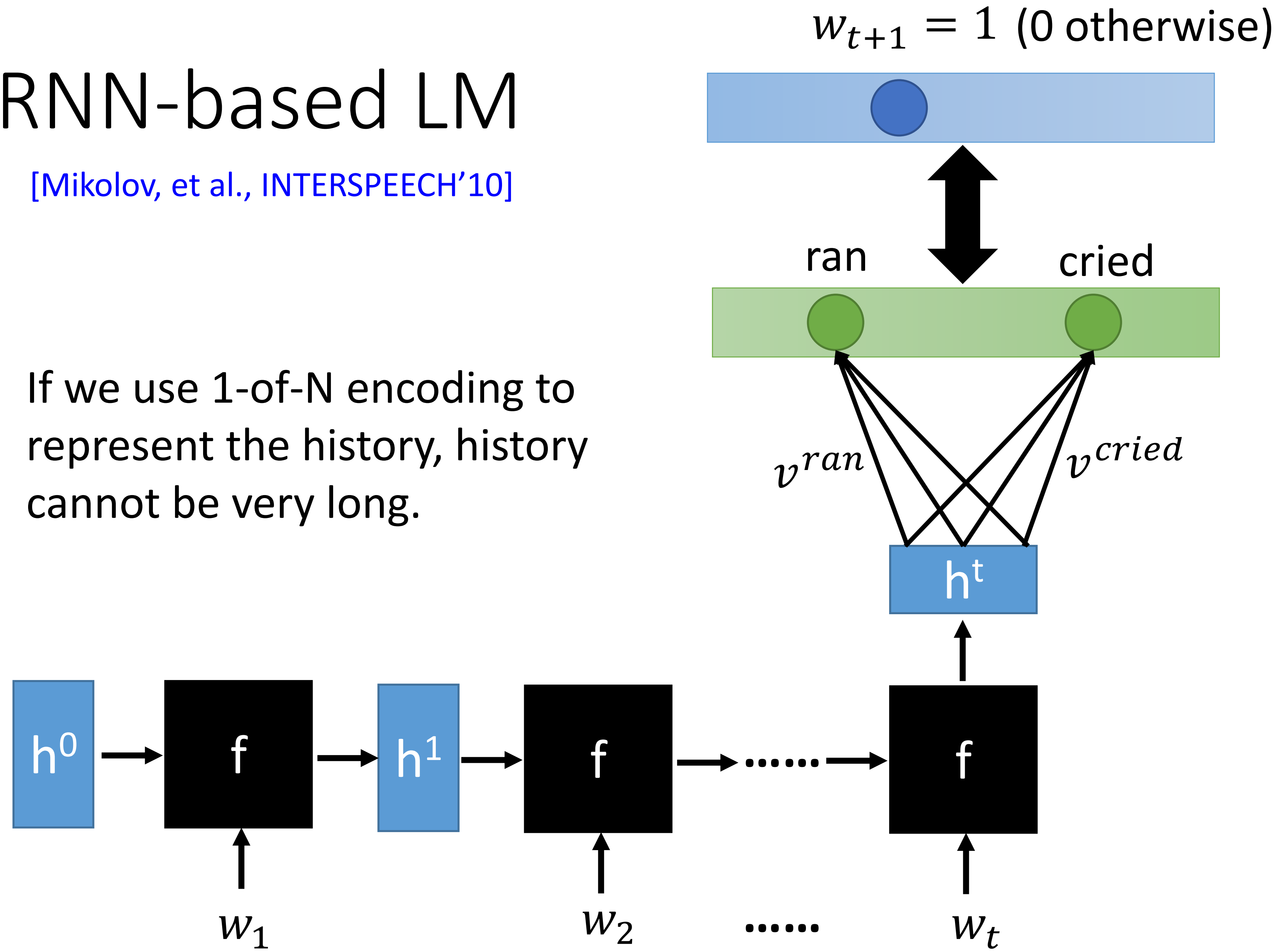



[Bengio, et al., JMLR’03]

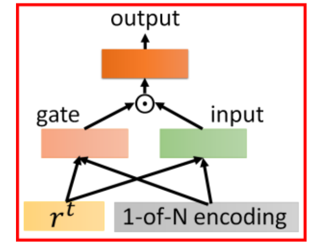
# RNN-based LM

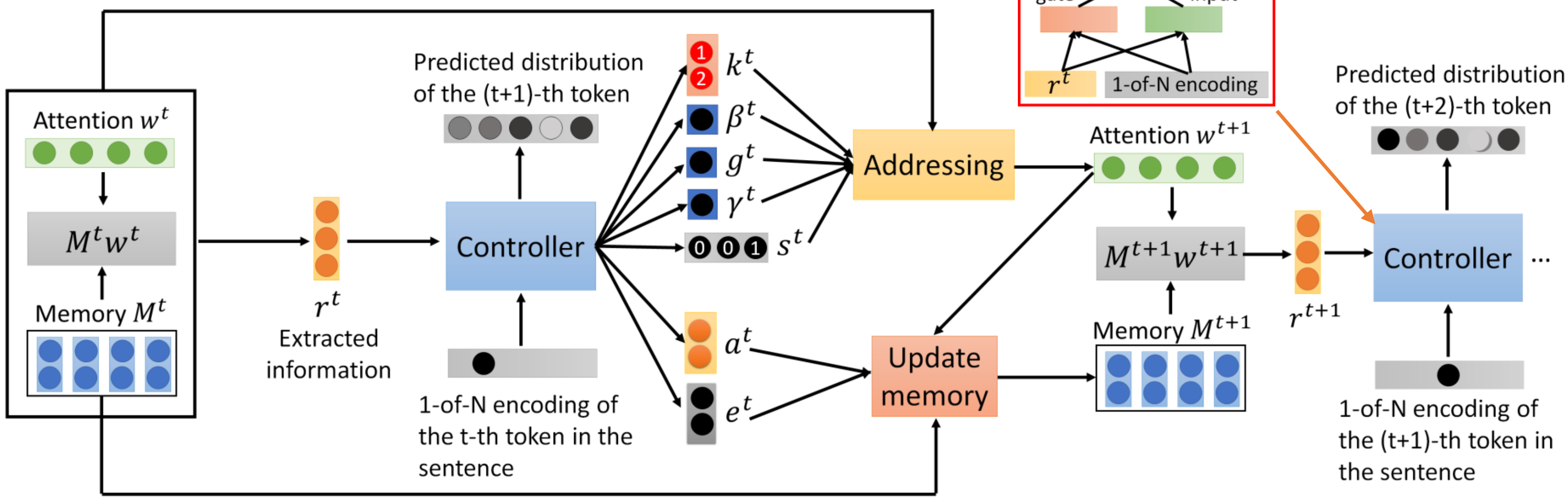
[Mikolov, et al., INTERSPEECH’10]

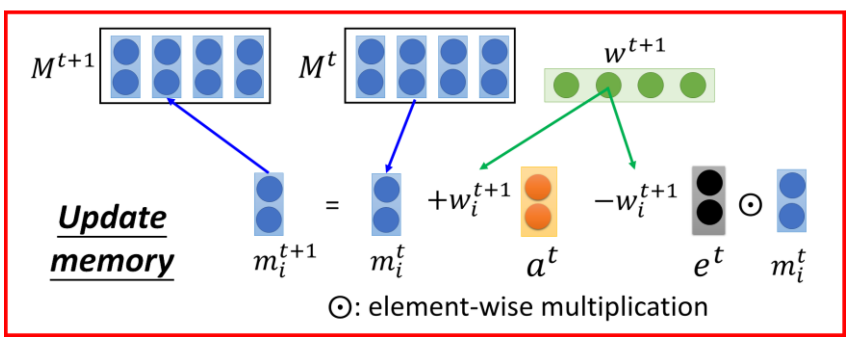
If we use 1-of-N encoding to represent the history, history cannot be very long.



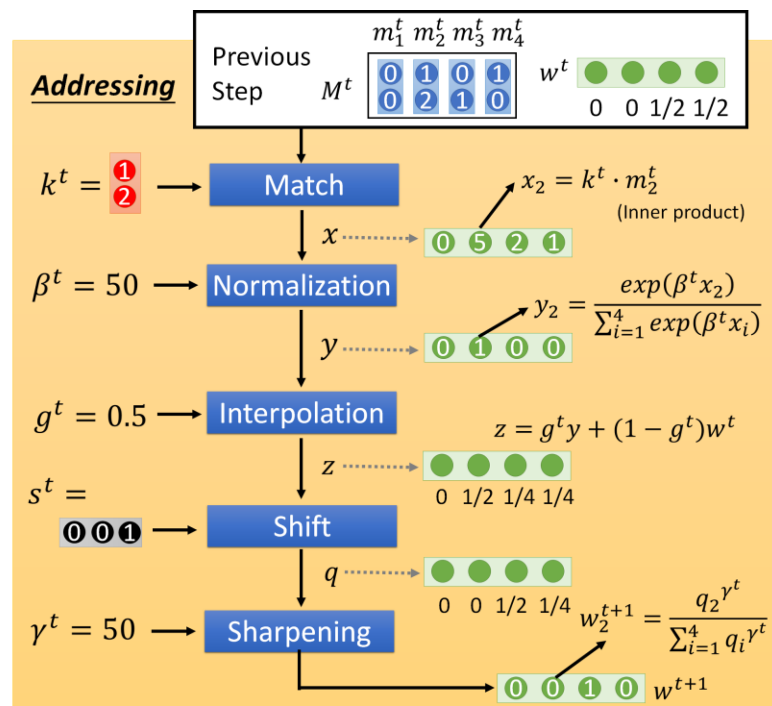
# Can be very complex ……




[Ko, et al., ICASSP'17]




LSTM with proper optimization and regularization can be good.

[Merity, et al., ICLR'18]

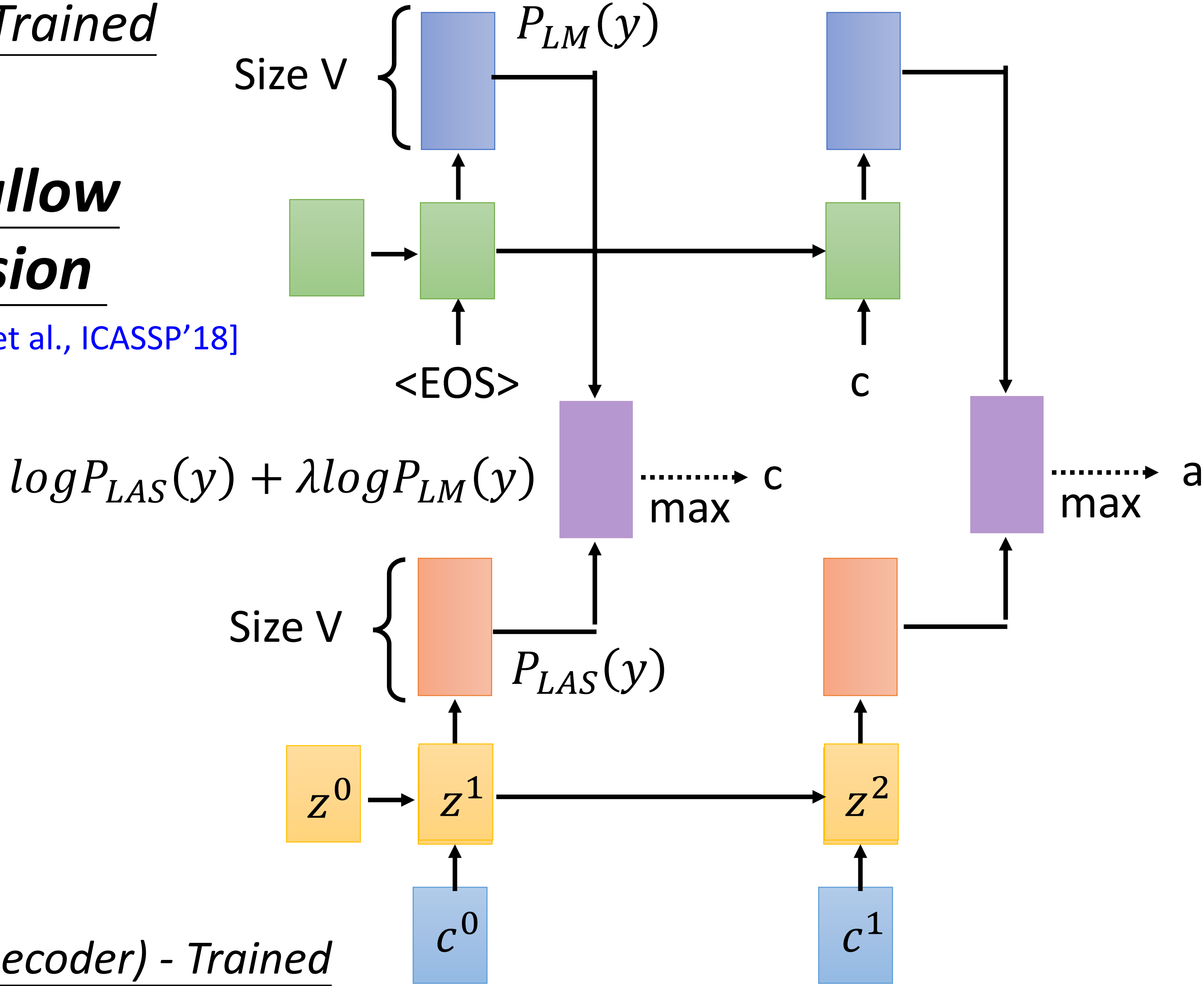
# How to use LM to improve LAS?

| | | how to integrate | |
|---|---|---|---|
| | | Output | Hidden |
| when to integrate | After Training | Shallow Fusion | Deep Fusion |
| | Before Training | | Cold Fusion |

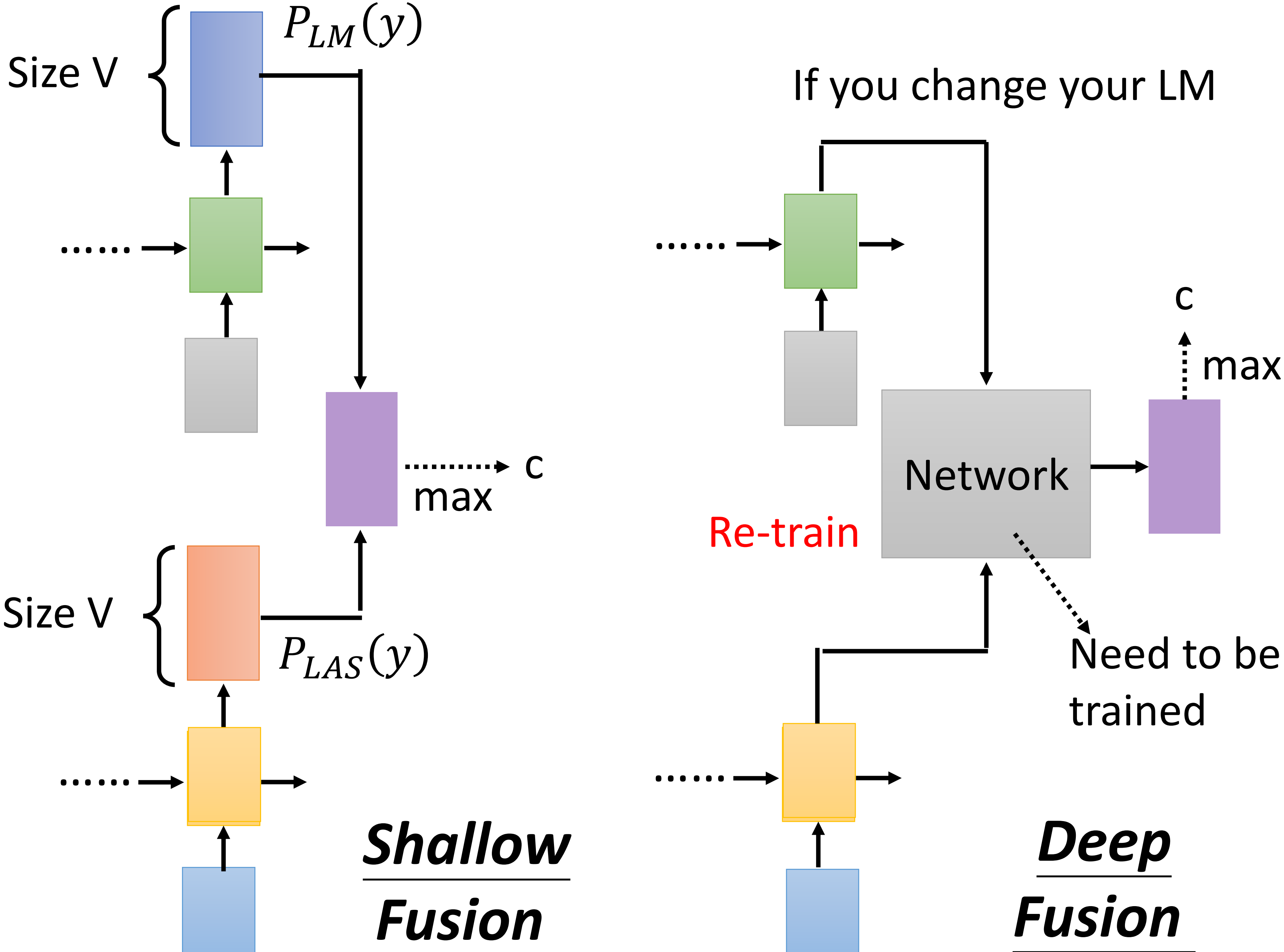
*LM - Trained*

***Shallow Fusion***

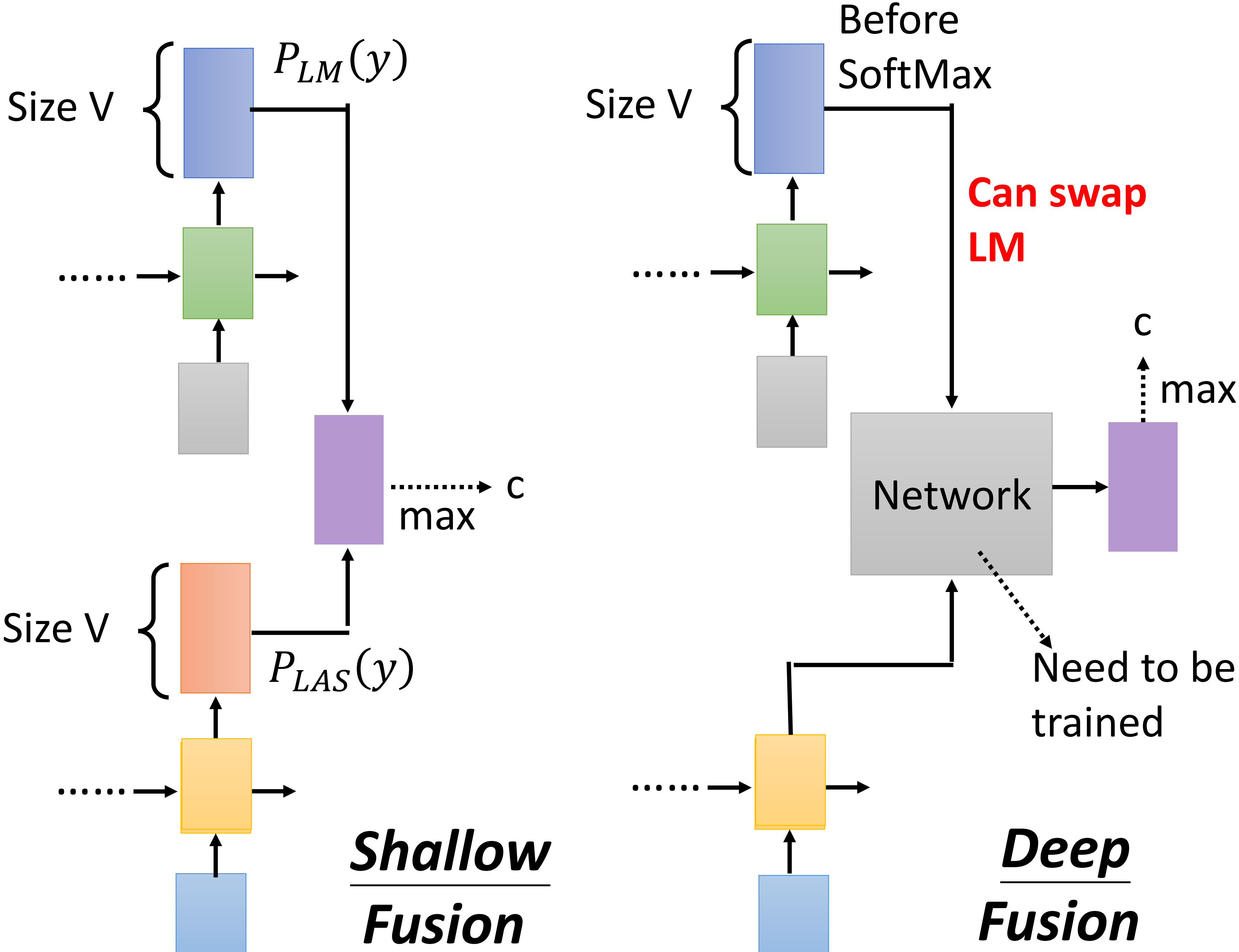
[Kannan, et al., ICASSP'18]




*LAS (Decoder) - Trained*



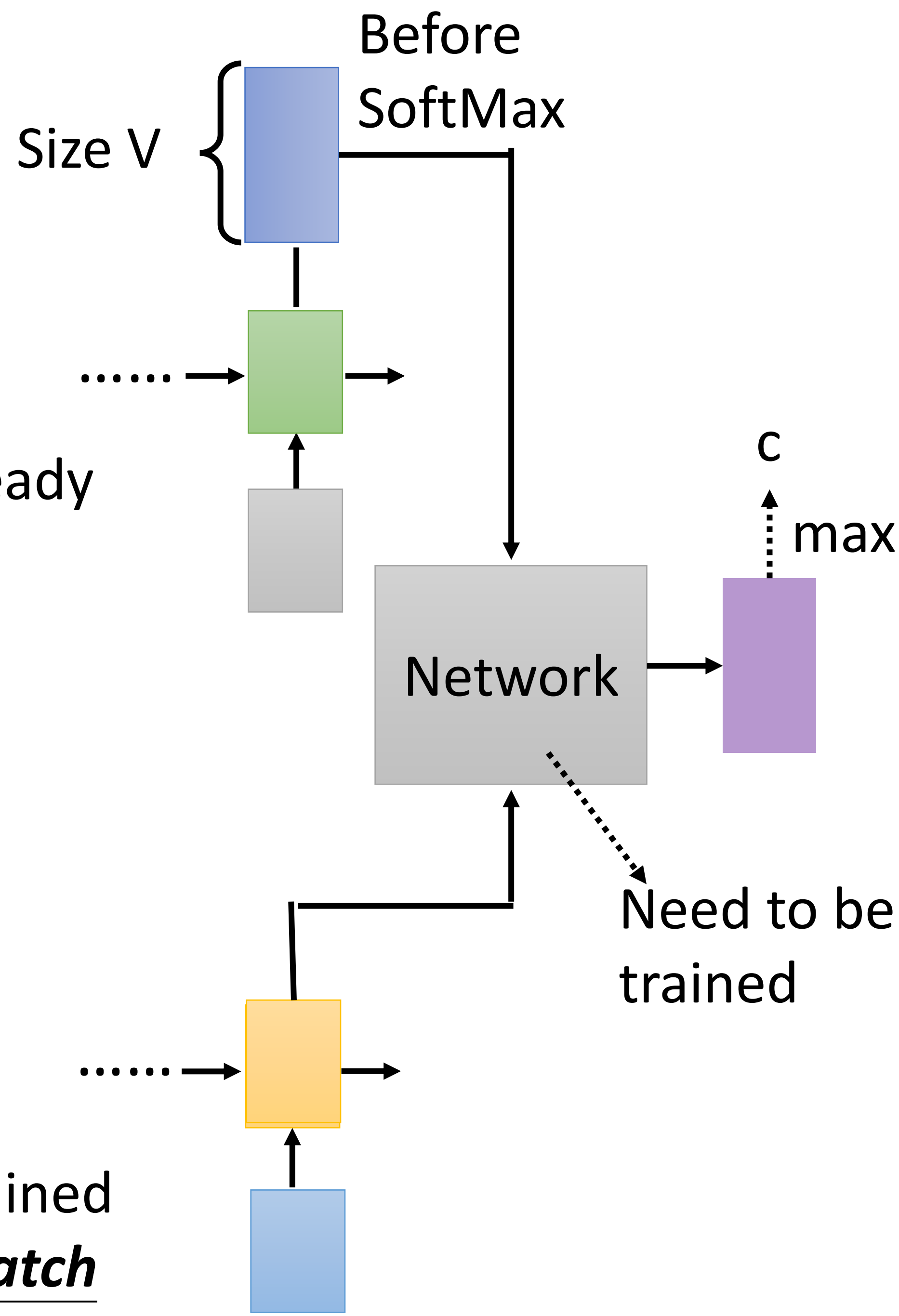

[Gulcehre, et al., arXiv'15]



Size V
$P_{LM}(y)$
max
c
Size V
$P_{LAS}(y)$
Shallow
Fusion
Before
SoftMax
Size V
Can swap
LM
c
max
Network
Need to be
trained
Deep
Fusion

# ***Cold Fusion***

[Sriram, et al., INTERSPEECH'18]




- LAS converges faster during training
- LAS has to be trained again if you have a new LM.

# Concluding Remarks

how to integrate

when to integrate

| | Output | Hidden |
|---|---|---|
| After Training | Shallow Fusion | Deep Fusion |
| Before Training | | Cold Fusion |

# Reference


- [Bengio, et al., JMLR'03] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, Christian Janvin, A neural probabilistic language model, The Journal of Machine Learning Research, March 2003
- [Mikolov, et al., INTERSPEECH'10] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, Sanjeev Khudanpur, Recurrent Neural Network Based Language Model INTERSPEECH, 2010
- [Ko, et al., ICASSP'17] Wei-Jen Ko, Bo-Hsiang Tseng, Hung-yi Lee, "Recurrent Neural Network based Language Modeling with Controllable External Memory", ICASSP, 2017
- [Merity, et al., ICLR'18] Stephen Merity, Nitish Shirish Keskar, Richard Socher, Regularizing and optimizing LSTM language models, ICLR, 2018

# Reference


- [Gulcehre, et al., arXiv’15] Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, Yoshua Bengio, On Using Monolingual Corpora in Neural Machine Translation, arXiv, 2015
- [Sriram, et al., INTERSPEECH’18] Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, Adam Coates, Cold Fusion: Training Seq2Seq Models Together with Language Models, INTERSPEECH, 2018
- [Kannan, et al., ICASSP’18] Anjuli Kannan, Yonghui Wu, Patrick Nguyen, Tara N. Sainath, Zhifeng Chen, Rohit Prabhavalkar, An analysis of incorporating an external language model into a sequence-to-sequence model, ICASSP, 2018