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: \( Y = y_1, y_2, \ldots, y_n \)
  • \( P(y_1, y_2, \ldots, y_n) \)

\[
\text{HMM} \quad Y^* = \arg\max_Y P(X|Y)P(Y)
\]

LM is usually helpful when your model outputs text

\[
\text{LAS} \quad Y^* = \arg\max_Y P(Y|X) \quad P(Y)
\]

Need paired data  Easy to collect
Why we need LM?

\[ Y^* = \arg \max_Y P(Y|X) P(Y) \]

Need paired data

Easy to collect

Words in Transcribed Audio

12,500 hours transcribed audio

\[ = 12,500 \times 60 \times 130 \approx \text{一億!} \]

(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
Why we need LM?

\[ Y^* = \arg \max_Y P(Y|X) \cdot P(Y) \]

Need paired data     Easy to collect

---

**Words in Transcribed Audio**

12,500 hours transcribed audio

\[ = 12,500 \times 60 \times 130 \approx \text{一億}! \]

(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

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

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

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

$P(\text{“wreck a nice beach”}) = P(\text{wreck} | \text{START})P(\text{a} | \text{wreck})P(\text{nice} | \text{a})P(\text{beach} | \text{nice})$
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:

\[
P( \text{jumped} \mid \text{the, dog} ) = 0.0001
\]

\[
P( \text{ran} \mid \text{the, cat} ) = 0.0001
\]

Give some small probability

This is called **language model smoothing.**
Continuous LM

- Recommendation system

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>5</th>
<th>5</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>5?</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

Matrix Factorization

Ref: https://youtu.be/ihwh5o_M4BNU?t=4673
Borrowing the idea from recommendation system

Continuous LM

<table>
<thead>
<tr>
<th></th>
<th>dog $h^1$</th>
<th>cat $h^2$</th>
<th>......</th>
<th>child</th>
</tr>
</thead>
<tbody>
<tr>
<td>ran</td>
<td>$v^1$</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>jumped</td>
<td>$v^2$</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>cried</td>
<td>$v^3$</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>laughed</td>
<td>$v^4$</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

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

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

Minimizing

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

$v^i, h^j$ found by gradient descent
### Continuous LM

Borrowing the idea from recommendation system

<table>
<thead>
<tr>
<th></th>
<th>dog $h^1$</th>
<th>cat $h^2$</th>
<th>......</th>
<th>child</th>
</tr>
</thead>
<tbody>
<tr>
<td>ran</td>
<td>$v^1$ 2</td>
<td>$n_{11}$ 3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>jumped</td>
<td>$v^2$ 0</td>
<td>$n_{22}$ 2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cried</td>
<td>$v^3$ 0</td>
<td>0</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>laughed</td>
<td>$v^4$ 0</td>
<td>0</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>......</td>
<td>......</td>
<td>......</td>
<td>......</td>
<td>......</td>
</tr>
</tbody>
</table>

Not observed

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.
Continuous LM

Consider it as a NN …

\[ L = \sum_{(i,j)} (v^i \cdot h^j - n_{ij})^2 \]
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

\[ P(\text{“wreck a nice beach”}) = P(\text{wreck} | \text{START})P(\text{a} | \text{wreck})P(\text{nice} | \text{a})P(\text{beach} | \text{nice}) \]

\[ P(b | a): \text{the probability of NN predicting the next word.} \]
\[ i\text{-th output} = P(w_t = i \mid \text{context}) \]

[Bengio, et al., JMLR’03]
RNN-based LM

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

\[ w_{t+1} = 1 \text{ (0 otherwise)} \]
Can be very complex …

LSTM with proper optimization and regularization can be good.

[Ko, et al., ICASSP’17]

[Merity, et al., ICLR’18]
How to use LM to improve LAS?

<table>
<thead>
<tr>
<th></th>
<th>Output</th>
<th>Hidden</th>
</tr>
</thead>
<tbody>
<tr>
<td>After</td>
<td>Shallow Fusion</td>
<td>Deep Fusion</td>
</tr>
<tr>
<td>Training</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Before</td>
<td>Cold Fusion</td>
<td></td>
</tr>
<tr>
<td>Training</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

how to integrate

when to integrate
\[ \log P_{LAS}(y) + \lambda \log P_{LM}(y) \]

[Ref: Kannan, et al., ICASSP'18]
Deep Fusion

Shallow Fusion

Size $V$

$P_{LM}(y)$

$P_{LAS}(y)$

If you change your LM

Network

Max $c$

Re-train

Need to be trained

$[Gulcehre, et al., arXiv'15]$
Size $V$ \( P_{LM}(y) \) \( \max \) \( c \) 

Size $V$ \( P_{LAS}(y) \) 

Shallow Fusion

Before SoftMax

Can swap LM

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.

LAS is trained \textit{from scratch}
Concluding Remarks

<table>
<thead>
<tr>
<th>when to integrate</th>
<th>Output</th>
<th>Hidden</th>
</tr>
</thead>
<tbody>
<tr>
<td>After Training</td>
<td>Shallow Fusion</td>
<td>Deep Fusion</td>
</tr>
<tr>
<td>Before Training</td>
<td></td>
<td>Cold Fusion</td>
</tr>
</tbody>
</table>
Reference


• [Mikolov, et al., INTERSPEECH’10] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, Sanjeev Khudanpur, Recurrent Neural Network Based Language Model INTERSPEECH, 2010


• [Merity, et al., ICLR’18] Stephen Merity, Nitish Shirish Keskar, Richard Socher, Regularizing and optimizing LSTM language models, ICLR, 2018
Reference


• [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