Language Modeling
Hung-yi Lee
Language modeling

• Language model: Estimated the probability of word sequence
  • Word sequence: \( w_1, w_2, w_3, \ldots, w_n \)
  • \( P(w_1, w_2, w_3, \ldots, w_n) \)

• Application: speech recognition
  • Different word sequence can have the same pronunciation

If \( P(\text{recognize speech}) \) > \( P(\text{wreck a nice beach}) \)
Output = “recognize speech”

• Application: sentence generation
N-gram

• How to estimate $P(w_1, w_2, w_3, ..., w_n)$
• Collect a large amount of text data as training data
  • However, the word sequence $w_1, w_2, ..., w_n$ may not appear in the training data

• N-gram language model: $P(w_1, w_2, w_3, ..., w_n) = P(w_1|\text{START})P(w_2|w_1) ... P(w_n|w_{n-1})$
  • 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 ......
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 …
……

Minimizing cross entropy

Neural Network

Neural Network

Neural Network
NN-based LM

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

\[ P(b | a) \text{: the probability of NN predicting the next word.} \]
RNN-based LM

• Training

Collect data:
潮水 退了 就 知道 誰 ...
不爽 不要 買 ...
公道價 八萬 一 ...
........
RNN-based LM

➢ Modeling long-term information
➢ People also use Deep RNN or LSTM

• To compute $P(w_1, w_2, w_3, \ldots, w_n)$ by RNN

$$P(w_1, w_2, w_3, \ldots, w_n) = P(w_1)P(w_2 | w_1)P(w_3 | w_1, w_2) \ldots \ldots P(w_n | w_1, w_2 \ldots w_{n-1})$$
Challenge of N-gram

• The estimated probability is not accurate.
  • Especially when we consider n-gram with large n
  • Because of data sparsity
    • Large model, not sufficient data

P( jumped | the, dog ) = 0
P( ran | the, cat ) = 0

This is called language model smoothing.

Training Data:

The dog ran ......
The cat jumped ......

Give some small probability

0.0001 0.0001
Matrix Factorization

Recommendation System:
History as customer, vocabulary as product ......

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

\[ n_{12} = v^1 \cdot h^2 \]
\[ n_{21} = v^2 \cdot h^1 \ldots \]

\[ v^i, h^j \] found by gradient descent

\[ P(\text{jumped} | \text{cat}) \]

\[ v^i, h^j \text{ are vectors to be learned} \]

Not observed

Vocabulary

<table>
<thead>
<tr>
<th></th>
<th>dog</th>
<th>cat</th>
<th>......</th>
<th>child</th>
</tr>
</thead>
<tbody>
<tr>
<td>ran</td>
<td>( v^1 )</td>
<td>0.2</td>
<td>0.3</td>
<td>0.1</td>
</tr>
<tr>
<td>jumped</td>
<td>( v^2 )</td>
<td>0</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>cried</td>
<td>( v^3 )</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>laughed</td>
<td>( v^4 )</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>......</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Matrix Factorization

<table>
<thead>
<tr>
<th>Vocabulary</th>
<th>dog</th>
<th>cat</th>
<th>......</th>
<th>child</th>
</tr>
</thead>
<tbody>
<tr>
<td>ran</td>
<td>0.2</td>
<td>0.3</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>jumped</td>
<td>0</td>
<td>0.2</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>cried</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>laughed</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>......</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Recommendation System: History as customer, vocabulary as product ......

History “dog” and “cat” can have similar vector $h^{\text{dog}}$ and $h^{\text{cat}}$.

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

Even if we have never seen “dog jumped ...”

Smoothing is automatically done.
Matrix Factorization

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

Consider it as a NN ……
If we use 1-of-N encoding to represent the history, history cannot be very long.
Class-based Language Modeling

class 1: Animal
- dog
- cat
- bird

class 2: Verb
- ran
- jumped
- walk

class 3: Function word
- the
- by
- a

\[ W = "w_1 \, w_2 \, w_3" \]
\[ C(w_i): \text{class of word } w_i \]

\[ P(W) = P(w_1 | \text{START}) \, P(w_2 | w_1) \, P(w_3 | w_2) \]

\[ P(W) = P(C(w_1) | \text{START}) \, P(C(w_2) | C(w_1)) \, P(C(w_3) | C(w_2)) \]
\[ \times \, P(w_1 | C(w_1)) \, P(w_2 | C(w_2)) \, P(w_3 | C(w_3)) \]
Class-based Language Modeling

Class 1: Animal
dog
cat
bird

Class 2: Verb
ran
jumped
walk

Class 3: Function word
the
by
a

W = “the dog ran”

\[ P(W) = P(F|\text{START}) \cdot P(A|F) \cdot P(V|A) \]
\[ \times P(\text{the}|F) \cdot P(\text{dog}|A) \cdot P(\text{ran}|V) \]

\( P(\text{class i | class j}) \) and \( P(\text{word w | class i}) \) are estimated from training data.
Class-based Language Modeling

\[ P( \text{class } i \mid \text{class } j ) \] and \[ P(\text{word } w \mid \text{class } i) \] are estimated from training data.

**Training data**

\[
\begin{array}{ccc}
\text{the} & \text{dog} & \text{ran} \\
F & A & V \\
\end{array}
\quad
\begin{array}{ccc}
\text{the} & \text{cat} & \text{jumped} \\
F & A & V \\
\end{array}
\]

\[ W = \text{“the cat ran”} \]

\[
\begin{array}{ccc}
F & A & V \\
\end{array}
\]

\[ P(\text{ran} \mid \text{cat}) \] is zero given the training data.

However, \[ P(\text{Verb} \mid \text{Animal}) \] is not zero.
Soft Word Class

How to determine the classes of the words?

1-of-N Encoding

apple = [1 0 0 0 0 0]
bag   = [0 1 0 0 0 0]
cat   = [0 0 1 0 0 0]
dog   = [0 0 0 1 0 0]
elephant = [0 0 0 0 1 0]

Word Embedding
RNN-based LM + Embedding Layer
Character-based LM

Source of image:
http://karpathy.github.io/2015/05/21/rnn-effectiveness/
Long-term Information

\[ P_{LSTM}({})|_{history} - P_{N-gram}({})|_{history} \]

Andrej Karpathy, Justin Johnson, Li Fei-Fei, Visualizing and Understanding Recurrent Networks, https://arxiv.org/abs/1506.02078
Cell that turns on inside comments and quotes:

```c
/* Duplicate LSM field information. The ism_rule is opaque, so
 * re-initialized */
static inline int audit_duplicate_lsm_field(struct audit_field *df,
                                          struct audit_field *sf)
{
    int ret = 0;
    char *lsm_str;
    /* our own copy of lsm_str */
    lsm_str = kstrdup(df->lsm_str, GFP_KERNEL);
    if (unlikely(!lsm_str))
        return -ENOMEM;
    df->lsm_str = lsm_str;
    /* our own (refreshed) copy of lsm_rule */
    ret = security_audit_rule_init(df->type, df->op, df->lsm_str, (void **)&df->lsm_rule);
    /* Keep currently invalid fields around in case they become valid after a policy reload. */
    if (ret == -EINVAL)
        pr_warn("audit rule for LSM \"%s\" is invalid\n",
                df->lsm_str);
    df->lsm_str = lsm_str;
    ret = 0;
}
return ret;
```

Cell that is sensitive to the depth of an expression:

```c
#define CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
    int i;
    for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
        if (mask[i] & classes[class][i])
            return 1;
    return 0;
}
```

Cell that might be helpful in predicting a new line. Note that it only turns on for some ")":

```c
char *audit_unpack_string(void *bufp, size_t *remain, si
{
    char *str;
    l𝐝|= weaponry (len = 8) || (len > *remain))
        return ERR_PTR(-EINVAL);
    /* Or the currently implemented string fields, PATH_MAX */
    *defines the longest valid length.*
    /`
    if (len > PATH_MAX)
        return ERR_PTR(-ENOMEM);
    str = kmalloc(len + 1, GFP_KERNEL);
    if (unlikely(!str))
        return ERR_PTR(-ENOMEM);
    memcpy(str, *bufp, len);
    str[len] = 0;
    *bufp += len;
    *remain -= len;
    return str;
```
The sole importance of the crossing of the Berezina lies in the fact that it plainly and indubitably proved the fallacy of all the plans for cutting off the enemy's retreat and the soundness of the only possible line of action—the one Kutuzov and the general mass of the army demanded—namely, simply to follow the enemy up. The French crowd fled at a continually increasing speed and all its energy was directed to reaching its goal. It fled like a wounded animal and it was impossible to block its path. This was shown not so much by the arrangements it made for crossing as by what took place at the bridges. When the bridges broke down, unarmed soldiers, people from Moscow and women with children who were with the French transport, all—carried on by vis inertiae—pressed forward into boats and into the ice-covered water and did not surrender.

"You mean to imply that I have nothing to eat out of... On the contrary, I can supply you with everything even if you want to give dinner parties," warmly replied Chichagov, who tried by every word he spoke to prove his own rectitude and therefore imagined Kutuzov to be animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating smile: "I meant merely to say what I said."

A large portion of cells are not easily interpretable. Here is a typical example:

```c
* Unpack a filter field's string representation from user-space buffer. *
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
    char *str;
    if (!*bufp || (len == 0) || (len > *remain))
        return ERR_PTR(-EINVAL);
    /* Of the currently implemented string fields, PATH_MAX defines the longest valid length. */
```
CNN for LM

Wei-Jen Ko, Bo-Hsiang Tseng, Hung-yi Lee, “Recurrent Neural Network based Language Modeling with Controllable External Memory”, ICASSP, 2017
For Large Output Layer

- Factorization of the Output Layer
  - Mikolov Tomáš: Statistical Language Models based on Neural Networks. PhD thesis, Brno University of Technology, 2012. (chapter 3.4.2)
- Noise Contrastive Estimation (NCE)
- Hierarchical Softmax
  - F Morin, Y Bengio, “Hierarchical Probabilistic Neural Network Language Model”, Aistats, 2005
- Blog posts:
To learn more ......


• Ke Tran, Arianna Bisazza, Christof Monz, Recurrent Memory Networks for Language Modeling, NAACL, 2016

• Jianpeng Cheng, Li Dong and Mirella Lapata, Long Short-Term Memory-Networks for Machine Reading, arXiv preprint, 2016
Acknowledgement

- 感謝 傅彥禎、楊喻涵 同學發現投影片上的打字錯誤