Representation Learning for Information Retrieval

Kuan-Yu Chen (陳冠宇)

2018/11/30 @ TR-514, NTUST
Review

- Deep Learning
  - Feed-Forward Fully-Connected Neural Networks
  - Recurrent Neural Network
    - LSTM
    - GRU
  - Convolutional Neural Network
Language Modeling

• A goal of statistical language modeling is to learn the joint probability function of sequences of words in a language

\[ P(w_1, w_2, ..., w_T) \]

• A statistical model of language can be represented by the conditional probability of the next word given all the previous ones (chain rule)

\[ P(w_1, w_2, ..., w_T) = \prod_{t=1}^{T} P(w_t | w_1, w_2, ..., w_{t-1}) \]

\[ \approx \prod_{t=1}^{T} P(w_t | w_{t-n+1}, ..., w_{t-1}) \]

– Such statistical language models have already been found useful in many technological applications involving natural language
Neural Network Language Modeling

- The Neural Network Language Mode (NNLM) estimated a statistical (n-gram) language model for predicting future words

\[ P(w_1, w_2, \ldots, w_T) \approx \prod_{t=1}^{T} P(w_t | w_{t-n+1}, \ldots, w_{t-1}) \]
From Modeling to Vectorization

• Recent methods for learning vector space representations of words have succeeded in capturing fine-grained semantic and syntactic regularities
  – One-hot representation vs. Distributed representation

• The two main model families for learning word vectors
  1. Global matrix factorization methods
     • Global Vector
  2. Local context window methods
     • Continuous Bag-of-Words model, and Skip-gram model
Perhaps one of the most-known seminal studies on developing word embedding methods was rooted in the Neural Network Language Modeling (NNLM) – It estimated a statistical (n-gram) language model for predicting future words in context while inducing word embeddings as a by-product.

- It estimated a statistical (n-gram) language model for predicting future words in context while inducing word embeddings as a by-product.
Continuous Bag-of-Words Modeling – 1

- Rather than seeking to learn a statistical language model, the CBOW model manages to obtain a dense vector representation (embedding) of each word directly.

\[
\prod_{t=1}^{T} P(w_t | w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}) = \prod_{t=1}^{T} \frac{\exp(v_{\bar{w}_t} \cdot v_{w_t})}{\sum_{w \in V} \exp(v_{\bar{w}_t} \cdot v_w)}
\]

\[
v_{\bar{w}_t} = \frac{1}{2c} \sum_{j=-c \& \& j \neq 0}^{c} v_{w_{t+j}}
\]
Continuous Bag-of-Words Modeling – 2

\[
\prod_{t=1}^{T} P(w_t | w_{t-c}, ..., w_{t-1}, w_{t+1}, ..., w_{t+c}) = \prod_{t=1}^{T} \frac{\exp(v_{w_t} \cdot v_{w_t})}{\sum_{w \in V} \exp(v_{\overline{w}_t} \cdot v_w)}
\]

One-hot vs. Dense – 1
One-hot vs. Dense – 2

\[ \text{Concatenate/Average} \]

\[ \text{the} \quad \text{on} \quad \text{cat} \quad \text{sat} \]

\[ \begin{array}{c}
\text{1} \\
\end{array} \begin{array}{c}
\times \\
\end{array} \begin{array}{c}
\text{=}
\end{array} \begin{array}{c}
\text{} \\
\end{array} \]
Skip-Gram Modeling

- In contrast to the CBOW model, the SG model employs an inverse training objective for learning word representations

\[
\prod_{t=1}^{T} P(w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}|w_t)
\]

\[
= \prod_{t=1}^{T} \prod_{j=-c\&\&j\neq0}^{c} P(w_{t+j}|w_t)
\]

\[
= \prod_{t=1}^{T} \prod_{j=-c\&\&j\neq0}^{c} \frac{\exp(v_{w_{t+j}} \cdot v_{w_t})}{\sum_{w\in V} \exp(v_w \cdot v_{w_t})}
\]

- In the implementations of CBOW and SG, the hierarchical soft-max algorithm and the negative sampling algorithm can make the training process more efficient and effective.
The Training Process

• Negative Sampling
  – Noise contrastive estimation (NCE) posits that a good model should be able to differentiate data from noise

• Hierarchical Softmax
  – The main advantage is that it is needed to evaluate only about $\log_2(V)$ nodes
Hierarchical Softmax – 1

\[ P(A \mid w) = 0.7 \times 0.2 \times 0.9 \]
\[ \vdots \]
\[ P(H \mid w) = 0.3 \times 0.5 \times 0.2 \]
Hierarchical Softmax – 2

cat

Average

the

sat
Hierarchical Softmax – 3

The diagram represents a hierarchical softmax model, which is used in natural language processing to classify words. The model is structured in a tree-like manner, with each level representing a finer granularity of classification. The top level might represent broad categories, while the lower levels provide more specific classification. The diagram shows how the model averages the softmax scores across different paths to predict the most likely class for a given input.
Global Vector Model (GloVe)

• The idea is:
  – For word solid related to ice but not steam, we expect the ratio:
    • $P_{\text{ice-solid}} / P_{\text{steam-solid}}$ will be large
    • $P_{\text{ice-gas}} / P_{\text{steam-gas}}$ will be small
    • $P_{\text{ice-fashion}} / P_{\text{steam-fashion}}$ should be close to one

• The starting point for word vector learning should be with ratios of co-occurrence probabilities rather than the probabilities themselves
  – A weighted least squares regression model can be introduced to addresses these problems

$$\sum_{i=1}^{|V|} \sum_{j=1}^{|V|} f(X_{ij}) \left(v_{wi} \cdot v_{wj} + b_i + b_j - \log(X_{ij})\right)^2$$

Singular Value Decomposition

• It is worthy to note that **SG** and **GloVe** have an implicit/explicit relation with the classic weighted matrix factorization approach

• Motivated by the relationship between word embedding methods with matrix factorization, we also leverage the singular value decomposition (SVD) to derive the word embeddings

\[
A_{|V|\times|V|} \approx U_{|V|\times K} \Sigma_{K\times K} V^T_{K\times |V|} = A'_{|V|\times|V|}
\]

\[
\|A - A'\|_F^2 = \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} (\log(X_{ij}) - U_i \cdot U_j)^2
\]

- Each row vector of matrix \( U \) (or the column vector of matrix \( V^T \)) is the word embeddings corresponding to each distinct word in the vocabulary
Classic Word Embeddings

- Various word embeddings have been proposed and applied to several NLP-related tasks
  - Prediction-based Methods
    - CBOW and Skip-gram
      - Local context
  - Count-based Methods
    - GloVe and SVD
      - Global matrix

- The \textbf{interpretation} of the learned value of each dimension in the representation is not intuitively clear
A Novel Word Embeddings

- The Distributional Skip-gram Model (DSG)
  - The desired word representations $\mathbf{M}$
  - The context word representations $\mathbf{W}$
  - Two assumptions

\[
\prod_{t=1}^{T} P(w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c} | w_t)
= \prod_{t=1}^{T} \prod_{j=-c \& \& j \neq 0}^{c} P(w_{t+j} | w_t)
= \prod_{t=1}^{T} \prod_{j=-c \& \& j \neq 0}^{c} \frac{\mathbf{W}_{w_{t+j}} \cdot \mathbf{M}_{w_t}}{\sum_{w \in \mathcal{V}} \mathbf{W}_w \cdot \mathbf{M}_{w_t}}
= \prod_{i=1}^{V} \prod_{j=1}^{|V|} \left( \mathbf{W}_{w_i} \cdot \mathbf{M}_{w_j} \right)^{c(w_i, w_j)}
\]
A Novel Word Embeddings

• Since we assume that each row of the target word matrix $W$ stands for a multinomial distribution, so we can select higher probability words from each row.

• It is obvious that “apple” is not only a kind of matter in our daily life but also is a famous technology company.
  - The embeddings learned by DSG can be interpreted reasonably and systematically.

\[
\begin{array}{c}
\text{Daily Life} \\
\text{Tech. Company}
\end{array}
\]

\[
\begin{array}{c}
M_{\text{apple}} \\
\vdots \\
0.55 \\
\vdots \\
0.25 \\
\vdots
\end{array}
\]
A straightforward way to leverage the word embedding methods for IR is to represent a document (or query) by averaging the vector representations of words occurring in the document (query)

\[
\tilde{d} = \sum_{w \in d} \frac{c(w, d)}{|d|} v_w \\
\tilde{q} = \sum_{w \in q} \frac{c(w, q)}{|q|} v_w
\]

Accordingly, the cosine similarity measure can be used to quantify the relevance degree between a document and a query

\[
sim(d, q) = \cos(\tilde{d}, \tilde{q}) = \frac{\tilde{d} \cdot \tilde{q}}{||\tilde{d}|| \cdot ||\tilde{q}||}
\]
In addition to the vector space model, we can construct a new word-based language model for predicting the occurrence probability of any arbitrary word by using the word embeddings:

\[ P(w_j|w_i) = \frac{\exp(v_{w_j} \cdot v_{w_i})}{\sum_{w \in V} \exp(v_w \cdot v_{w_i})} \]

Consequently, the document model can be obtained by linearly combining the associated word-based language models of the words occurring in the document:

\[ P(q|d) = \prod_{j=1}^{\left|q\right|} P(w_j|d) = \prod_{j=1}^{\left|q\right|} \left( \sum_{i=1}^{\left|d\right|} P(w_j|w_i)P(w_i|d) \right) \]
Distributed Memory (DM) Model

- Learning of paragraph representations is more reasonable and suitable for some tasks
  - The distributed memory model, the distributed bag-of-words model, and the thought vector model

- The DM model is inspired from the CBOW model
  - The idea is that a given paragraph also contributes to the prediction of a next word

Distributed Bag-of-words (DBOW) Model

- Opposite to the DM model, a simplified version is to only leverage the paragraph representation to predict all of the words occurring in the paragraph.

- Since the model ignores the contextual words at the input layer, it is named the distributed bag-of-words (DBOW) model.

Learning Representations by RNNs

- The last hidden state can be thought as a representation for the entire sequence
Machine Translation

- RNN can be used to encode a variable-length source sentence, and then a variable-length target sentence will be generated by considering the encoded information
  - RNN Encoder-Decoder

你好嗎 <eof>

How are you <eof>
The skip-thought vector model presents an objective function that abstracts the skip-gram model to the sentence level.

- Instead of using a word to predict its surrounding context, thought vector encodes a sentence to predict the sentences around it.

Classic Paragraph Embedding Methods

- Classic paragraph embedding methods infer the representation of a given paragraph by considering all of the words occurring in the paragraph
  - Such as the Distributed Memory model, the Distributed Bag-of-words model, and the Skip-Thought Vector model

- The stop or function words that occur frequently may mislead the embedding learning process
  - The learned representation for the paragraph might be undesired
  - The performance is limited
  - Our goal is to
    - Distill the most representative information from a given paragraph
    - Get rid of the general background information
Learning to Distill

- We assume that each paragraph can be assembled by the paragraph specific information and the general background information
  - This assumption also holds in the low-dimensional representation space
  - Three modules
    - Paragraph encoder $f(\cdot)$
    - Background encoder $g(\cdot)$
    - Decoder $h(\cdot)$

A brilliant property inherits in the EV model is that it can be readily inferred a “paragraph” specific language model

\[ \hat{P}(w) \equiv h \left( f \left( P_{Dt} \right) \right) \]
Homework 5 – Description

• In this project, you will have
  – 500 Queries
  – 2265 Documents

• Our goal is to implement a word embedding-based retrieval method
  – The first step is to obtain a set of word embeddings
    • https://radimrehurek.com/gensim/models/word2vec.html
    • https://github.com/facebookresearch/fastText
    • https://nlp.stanford.edu/projects/glove/
  – Next, you can incorporate word embeddings into VSM, QLM or KLM
• Please login our competition page at Kaggle
  – https://www.kaggle.com/t/7671ea5f1d834eec883b329f1580cda4

• Your team name is ID_Name
  – M123456_陳冠宇
Homework 5 – Submission Format
Homework 5 – Scoring

• The evaluation measure is **MAP@100**

• In order to encourage you to try any methods and ideas, we **do not** set the maximum number of daily submissions
  – Please **Do Not** register several teams

• The **hard** deadline is 12/28 11:00
  – If your score is higher than the simple baseline
    • YourScore = 2 + \( \frac{YourMAP - MinimumMAP}{HighestMAP - MinimumMAP} \) × 8%

• You should also upload source codes onto the Moodle system
  – TA will ask you to demo your program
  – In this HW, you **must** leverage word embeddings to do retrieval
Tentative Grading

- Homework: 70%
  - Programming with C/C++/Python/Matlab
    - HW1: 10%
    - HW2: 10%
    - HW3: 15%
    - HW4: 15%
    - HW5: 15%
- Midterm: 10%
- Presentation: 10%
- Competition: 20%
Final Competition.

• You should select a paper before 12/7
  – Oral presentation for 20 minutes
    • Motivations
    • Methods
    • Performances
    • Others

<table>
<thead>
<tr>
<th>Team List</th>
<th>Paper Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. 王0楷林0哲沈0一施0仰</td>
<td>An accelerated PSO for query expansion in web information retrieval: application to medical dataset</td>
</tr>
<tr>
<td>B. 余0浩楊0復時0軒吳0甄</td>
<td></td>
</tr>
<tr>
<td>C. 蘇0群鄭0元林0宇鍾0芯</td>
<td></td>
</tr>
<tr>
<td>D. 吳0康楊0洋蔡0軒</td>
<td></td>
</tr>
<tr>
<td>E. 戈0磊廖0明郭0為</td>
<td></td>
</tr>
<tr>
<td>F. 王0修周0晨陳0動林0鐸</td>
<td></td>
</tr>
<tr>
<td>G. 吳0琪苗0綺許0煥周0穎</td>
<td></td>
</tr>
<tr>
<td>H. 藍0中史0維呂0字</td>
<td></td>
</tr>
<tr>
<td>I. 邱0倫胡0聖江0邦陳0峰</td>
<td></td>
</tr>
<tr>
<td>J. 徐0瑋呂0祥</td>
<td></td>
</tr>
<tr>
<td>K. 潘0原簡岳0杰楊0穎</td>
<td></td>
</tr>
<tr>
<td>L. 郭0園陳0雲郭0詩楊0淮</td>
<td></td>
</tr>
<tr>
<td>M. 戈0磊郭0為廖0明</td>
<td>Learning a Deep Listwise Context Model for Ranking Refinement</td>
</tr>
</tbody>
</table>
Final Competition..

- In the final competition, we will have 100,000 documents, 150 training queries and 50 test queries
  - 100,000 documents and 150 training queries will be released at 12/28
  - 50 test queries will be released at 1/4 12:00
  - The kaggle system will open at 1/4 12:00 and will close at 1/5 6:00
  - The maximum number of submissions is 20 (or 40) for each team
  - After the competition, you can test your model freely on kaggle
  - \[ \text{YourScore} = 4 + \frac{\text{YourMAP} - \text{BaselineMAP}}{\text{HighestMAP} - \text{BaselineMAP}} \times 6\% \]

- Please submit a report and source codes before 1/9 23:59
Questions?

kychen@mail.ntust.edu.tw