Deep Learning for Information Retrieval

Hang Li & Zhengdong Lu
Huawei Noah’s Ark Lab
Outline of Tutorial

• Introduction
• Part 1: Basics of Deep Learning
• Part 2: Fundamental Problems in Deep Learning for IR
• Part 3: Applications of Deep Learning to IR
• Summary
Overview of Information Retrieval

**Key Questions:** How to Represent Intent and Content, How to Match Intent and Content

- Ranking, indexing, etc are less essential
- Interactive IR is not particularly considered here
Approach in Traditional IR

Query:
star wars the force awakens reviews

Document:
Star Wars: Episode VII
Three decades after the defeat of the Galactic Empire, a new threat arises.

\[
q = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad d = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 1 \end{bmatrix}
\]

\[
f(q,d) = \left\langle q, d \right\rangle / \| q \| \cdot \| d \|
\]

- Representing query and document as tf-idf vectors
- Calculating cosine similarity between them
- BM25, LM4IR, etc can be considered as non-linear variants
Approach in Modern IR

Query:
star wars the force awakens reviews

Document:
Star Wars: Episode VII
Three decades after the defeat of the Galactic Empire, a new threat arises.

- Conducting query and document understanding
- Representing query and document as feature vectors
- Calculating multiple matching scores between query and document
- Training ranker with matching scores as features using *learning to rank*
“Easy” Problems in IR

• Search
  – Matching between query and document

• Question Answering from Documents
  – Matching between question and answer

• Well studied so far
• Deep Learning may not help so much
“Hard” Problems in IR

• Image Retrieval
  – Matching between text and image
  – Not the same as traditional setting

• Question Answering from Knowledge Base
  – Complicated matching between question and fact in knowledge base

• Generation-based Question Answering
  – Generating answer to question based on facts in knowledge base

• Not well studied so far
• Deep Learning can make a big deal
Hard Problems in IR

Q: How tall is Yao Ming?

Q: A dog catching a ball

Q: How far is sun from earth?

Key Questions: How to Represent Intent and Content, How to Match Intent and Content

<table>
<thead>
<tr>
<th>Name</th>
<th>Height</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yao Ming</td>
<td>2.29m</td>
<td>134kg</td>
</tr>
<tr>
<td>Liu Xiang</td>
<td>1.89m</td>
<td>85kg</td>
</tr>
</tbody>
</table>

The average distance between the Sun and the Earth is about 92,935,700 miles.

A: It is about 93 million miles
Recent Progress: Deep Learning is particularly effective for hard IR problems.
Part 1: Basics of Deep Learning
Outline of Part 1

- Word Embedding
- Recurrent Neural Networks
- Convolutional Neural Networks
Word Embedding
Word Embedding

- Motivation: representing words with low-dimensional real-valued vectors, utilizing them as input to deep learning methods, vs one-hot vectors
- Method: SGNS (Skip-Gram with Negative Sampling)
- Tool: Word2Vec
- Input: words and their contexts in documents
- Output: embeddings of words
- Assumption: similar words occur in similar contexts
- Interpretation: factorization of mutual information matrix
- Advantage: compact representations (usually 100~ dimensions)
Skip-Gram with Negative Sampling
(Mikolov et al., 2013)

- Input: occurrences between words and contexts

<table>
<thead>
<tr>
<th>$M$</th>
<th>$C_1$</th>
<th>$C_2$</th>
<th>$C_3$</th>
<th>$C_4$</th>
<th>$C_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>5</td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$w_2$</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$w_3$</td>
<td></td>
<td>3</td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

- Probability model:

\[
P(D = 1 \mid w, c) = \sigma(\vec{w} \cdot \vec{c}) = \frac{1}{1 + e^{-\vec{w} \cdot \vec{c}}}
\]

\[
P(D = 0 \mid w, c) = \sigma(-\vec{w} \cdot \vec{c}) = \frac{1}{1 + e^{\vec{w} \cdot \vec{c}}}
\]
Skip-Gram with Negative Sampling

- Word vector and context vector: lower dimensional (parameter) vectors $\vec{w}, \vec{c}$
- Goal: learning of the probability model from data
- Take co-occurrence data as positive examples
- Negative sampling: randomly sample $k$ unobserved pairs $(w, c_N)$ as negative examples
- Objective function in learning

$$L = \sum_w \sum_c \#(w, c) \log \sigma(\vec{w} \cdot \vec{c}) + k \cdot \mathbb{E}_{c_N \sim P} \log \sigma(-\vec{w} \cdot \vec{c}_N)$$

- Algorithm: stochastic gradient descent
Interpretation as Matrix Factorization  
(Levy & Goldberg 2014)

- Pointwise Mutual Information Matrix

<table>
<thead>
<tr>
<th></th>
<th>(C_1)</th>
<th>(C_2)</th>
<th>(C_3)</th>
<th>(C_4)</th>
<th>(C_5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(w_1)</td>
<td>3</td>
<td>1.5</td>
<td>-0.5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>(w_2)</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>-0.5</td>
<td></td>
</tr>
<tr>
<td>(w_3)</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

\[
\log \frac{P(w, c)}{P(w)P(c)}
\]
Interpretation as Matrix Factorization

\[
M = WC^T
\]

Matrix factorization, equivalent to SGNS

<table>
<thead>
<tr>
<th>(w_1)</th>
<th>(c_1)</th>
<th>(c_2)</th>
<th>(c_3)</th>
<th>(c_4)</th>
<th>(c_5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>-0.5</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>-0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(w_1)</th>
<th>(t_1)</th>
<th>(t_2)</th>
<th>(t_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0.5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.5</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Word embedding
Recurrent Neural Network
Recurrent Neural Network

• Motivation: representing sequence of words and utilizing the representation in deep learning methods
• Input: sequence of word embeddings, denoting sequence of words (e.g., sentence)
• Output: sequence of internal representations (hidden states)
• Variants: LSTM and GRU, to deal with long distance dependency
• Learning of model: stochastic gradient descent
• Advantage: handling arbitrarily long sequence; can be used as part of deep model for sequence processing (e.g., language modeling)
Recurrent Neural Network (RNN) (Mikolov et al. 2010)

\[ h_t = f(h_{t-1}, x_t) \]

the cat sat on the mat

\[ h_t \rightarrow h_{t-1} \]

the cat sat x_{1\ldots t} mat
Recurrent Neural Network

\[ h_t = f(h_{t-1}, x_t) = \tanh(W_h h_{t-1} + W_x x_t + b_{hx}) \]
Long Term Short Memory (LSTM)  
(Hochreiter & Schmidhuber, 1997)

- A memory (vector) to store values of previous state
- Input gate, output gate, and forget gate to control
- Gate: element-wise product with vector of values in [0,1]

\[
\begin{align*}
    i_t &= \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i) \\
    f_t &= \sigma(W_{fh}h_{t-1} + W_{fx}x_t + b_f) \\
    o_t &= \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o) \\
    g_t &= \tanh(W_{gh}h_{t-1} + W_{gx}x_t + b_g) \\
    c_t &= i_t \otimes g_t + f_t \otimes c_{t-1} \\
    h_t &= o_t \otimes \tanh(c_t)
\end{align*}
\]
Gated Recurrent Unit (GRU)
(Cho et al., 2014)

• A memory (vector) to store values of previous state
• Reset gate and update gate to control

\[
\begin{align*}
    r_t &= \sigma(W_{rh}h_{t-1} + W_{rx}x_t + b_r) \\
    z_t &= \sigma(W_{zh}h_{t-1} + W_{zx}x_t + b_z) \\
    g_t &= \tanh(W_{gh}(r_t \otimes h_{t-1}) + W_{gx}x_t + b_g) \\
    h_t &= z_t \otimes h_{t-1} + (1 - z_t) \otimes g_t
\end{align*}
\]
Recurrent Neural Network Language Model

Model

\[ h_t = \tanh(W_h h_{t-1} + W_x x_t + b_{hx}) \]
\[ p_t = P(x_t | x_1 \cdots x_{t-1}) = \text{soft max}(W h_t + b) \]

Objective of Learning

\[ \frac{1}{T} \sum_{t=1}^{T} - \log \hat{p}_t \]

- Input one sequence and output another
- In training, input sequence is same as output sequence
Convolutional Neural Network
Convolutional Neural Network

• Motivation: representing sequence of words and utilizing the representation in deep learning methods

• Input: sequence of word embeddings, denoting sequence of words (e.g., sentence)

• Output: representation of input sequence

• Learning of model: stochastic gradient descent

• Advantage: robust extraction of n-gram features; can be used as part of deep model for sequence processing (e.g., sentence classification)
**Convolutional Neural Network (CNN)** (Kim 2014, Blunsom et al. 2014, Hu et al., 2014)

The cat sat on the mat

**Concatenation**
- Shared parameters on same level
- Fixed length, zero padding

**Max pooling**

**Convolution**
Example: Image Convolution

Filter

Dark Pixel Value = 1, Light Pixel Value = 0
Dot in Filter = 1, Others = 0

Leow Wee Kheng
Example: Image Convolution

Convolution Operation

- Scanning image with filter having 3*3 cells, among them 3 are dot cells
- Counting number of dark pixels overlapping with dot cells at each position
- Creating feature map (matrix), each element represents similarity between filter pattern and pixel pattern at one position
- Equivalent to extracting feature using the filter
- Translation-invariant

Feature Map

| 0 0 0 0 0 |
| 0 0 1 1 0 |
| 0 1 3 2 0 |
| 0 1 3 1 0 |
| 0 1 1 0 0 |
Convolution

\[ z_{i}^{(l,f)} = \sigma(w^{(l,f)} \cdot z_{i}^{(l-1)} + b^{(l,f)}) \quad f = 1, 2, \ldots, F_{l} \]

- \( z_{i}^{(l,f)} \) is output of neuron of type \( f \) for location \( i \) in layer \( l \)
- \( w^{(l,f)}, b^{(l,f)} \) are parameters of neuron of type \( f \) in layer \( l \)
- \( \sigma \) is sigmoid function

- \( z_{i}^{(l-1)} \) is input of neuron for location \( i \) from layer \( l-1 \)
- \( z_{i}^{(0)} \) is input from concatenated word vectors for location \( i \)

\[
 z_{i}^{(0)} = [x_{i}^{T}, x_{i+1}^{T}, \ldots x_{i+h-1}^{T}]^{T}
\]

Equivalent to n-gram feature extraction at each position
Max Pooling

\[ z_{i}^{(l,f)} = \max( z_{2i-1}^{(l-1,f)}, z_{2i}^{(l-1,f)} ) \]

\( z_{i}^{(l,f)} \) is output of pooling of type \( f \) for location \( i \) in layer \( l \)

\( z_{2i-1}^{(l-1,f)}, z_{2i}^{(l-1,f)} \) are input of pooling of type \( f \) for location \( i \) in layer \( l \)

Equivalent to n-gram feature selection
Sentence Classification
Using Convolutional Neural Network

\[ y = f(x) = \text{soft max}(Wz + b) \]
\[ z = \text{CNN}(x) \]
References


Part 2: Fundamental Problems in Deep Learning for IR
Outline of Part 2

• Representation Learning
• Matching
• Translation
• Classification
• Structured Prediction
Representation Learning
Representation of Word

Using real-valued vectors to represent the meaning of words
Breakthrough: learning of sentence representation becomes possible

Using real-valued vectors to represent the meaning of sentences
## Learning of Sentence Representation

### Task
- Compositional: from words to sentences
- Representing syntax, semantics, and even pragmatics of sentences

### Means
- Deep neural networks
- Big data
- Task-dependent
- Error-driven and usually gradient-based training
Fundamental Problems in Information Retrieval (and also Natural Language Processing)

- Classification: assigning a label to a string
  \[ S \rightarrow C \]
- Matching: matching two strings
  \[ s, t \rightarrow R^+ \]
- Translation: transforming one string to another
  \[ S \rightarrow t \]
- Structured prediction: mapping string to structure
  \[ S \rightarrow s' \]

- In general, \( s \) and \( t \) can be any type of data
- Non-interactive setting is mainly considered
Example: Fundamental Problems in Search

- Query Understanding (Classification and Structured Prediction)
  - Query Classification
  - Named entity Recognition in Query
- Document Understanding (Classification and Structured Prediction)
  - Document Classification
  - Named Entity Recognition in Document
- Query Document Matching (Matching)
  - Matching of Query and Document
- Summary Generation (Translation)
  - Generating Summaries of Relevant Documents
Learning of Representations in Fundamental Problems

- Classification
  \[ S \rightarrow r \rightarrow c \]

- Matching
  \[ S, t \rightarrow r \rightarrow R^+ \]

- Translation
  \[ S \rightarrow r \rightarrow t \]

- Structured Prediction
  \[ S \rightarrow s' + r \]
Matching
Matching

• Tasks
  – **Search**: query-document (title) matching, similar query finding
  – **Question Answering**: question answer matching

• Approaches
  – Projection to Latent Space
  – One Dimensional Matching
  – Two Dimensional Matching
  – Tree Matching

![Diagram of Deep Neural Network with arrows pointing to Question and Answer, and a Score arrow pointing upwards]
Matching: Projection to Latent Space

- Natural extension of Vector Space Model

Latent Space

Dot Product (Score)

Question Representation

Answer Representation

Neural Networks:
- Convolutional Neural Network
- Deep Neural Network
- Recurrent Neural Network

Neural Networks:
- Huang et al. 2013
- Shen et al. 2014
- Severyn & Moschitti 2015
Matching: One Dimensional Matching

Neural Network 1:
Convolutional Neural Network

Neural Network 2:
Deep Neural Network, Tensor Network

• Hu et al. 2014
• Qiu & Huang 2015
Matching: Two Dimensional Matching

Neural Network 1: Convolutional Neural Network
Neural Network 2: Deep Neural Network

Score

Question-Answer Matching Representation

• Hu et al. 2014
• Pang et al. 2016
• Wan et al. 2016
Matching: Tree Matching

**DNN with Sparse Input Layer**

**Neural Network**

**Tree Matching Representation**

**Matching Pattern Identification**

**Score**

**Tree Matching Patterns**

E.g. Q="how is X[city]", A="The food in X[city] is great"

- Question Parse Tree
- Answer Parse Tree

**Question**

**Answer**

• Wang et al. 2015
Key Observations

• CNN (Convolutional Neural Networks) usually works better than RNN (Recurrent Neural Networks) for matching (Ma et al.’15)

• 2-dimensional CNN works better than 1-dimensional CNN (Hu et al.’14)

• Representing matched tree patterns in neural network also works well, when there is enough training data (Wang et al.’15)

• Matching scores can be used as features of learning to rank models (Severyn & Moschitti’15)
References


References

Translation
Translation

• Tasks
  – **Question Answering**: answer generation from question
  – **Search**: similar query generation

• Approaches
  – Sequence-to-Sequence Learning
  – RNN Encoder-Decoder
  – Attention Mechanism
Translation: Sequence-to-Sequence Learning (Same for RNN Encoder-Decoder)

Encoder: Recurrent Neural Network
Decoder: Recurrent Neural Network
Translation: Sequence to Sequence Learning

- Hierarchical LSTM
- Different LSTM models for encoder and decoder
- Reverse order of words in source sentence

\[
P(y_t \mid y_1 \cdots y_{t-1}, x) = g(y_{t-1}, s_t)
\]
\[
h_t = f_e(x_t, h_{t-1}), s_t = f_d(y_{t-1}, s_{t-1})
\]

• Sutskever et al. 2014
Translation: RNN Encoder-Decoder

\[ P(y_t \mid y_1 \cdots y_{t-1}, x) = g(y_{t-1}, s_t, c), c = h_T \]
\[ s_t = f_d(y_{t-1}, s_{t-1}, c) \]
\[ h_t = f_e(x_t, h_{t-1}) \]

- Context vector represents source sentence
- GRU is used

\[ \text{Cho et al. 2014} \]
Translation: Attention Mechanism

- Context vector represents attention
- Corresponds to alignment relation
- Encoder: Bidirectional RNN

\[ P(y_t \mid y_1 \cdots y_{t-1}, x) = g(y_{t-1}, s_t, c_t) \]
\[ s_t = f_d(y_{t-1}, s_{t-1}, c_t) \]
\[ h_t = f_e(x_t, h_{t-1}) \]
\[ c_t = \sum_{j=1}^{T} \alpha_{ij} h_j \]
\[ \alpha_{ij} = q(s_{t-1}, h_j) \]

Bahdanau, et al. 2014
Key Observations

- RNNs (Recurrent Neural Networks) is more suitable for generation or translation
- LSTM and GRU can retain long distance dependency (Cho et al.’14)
- Bidirectional model works better than one-directional model (Bahdanau et al.’15)
- Attention mechanism can improve accuracy and efficiency of RNN models (Bahdanau et al.’15)
- Neural Machine Translation get generate more fluent but less faithful results than Statistical Machine Translation
References

• I. Sutskever, O. Vinyals, and Le, Q.V. Le. Sequence to Sequence Learning with Neural Networks. *NIPS* 2014.
Classification
Classification

• Tasks
  – **Search**: query classification, document classification
  – **Question Answering**: question classification, answer classification

• Approaches
  – World Level Model
  – Character Level Model
  – Hierarchical Model (for document classification)
Sentence Classification: Word Level Model

Classifier:
Softmax

Neural Network:
Convolutional Neural Network, Deep Neural Network

Input:
Continuous Word Embedding, Discrete Word Embedding (one-hot)

- Kim 2014
- Blunsom et al. 2014
- Johnson & Zhang 2015
- Iyyer et al. 2015
Document Classification: Character Level Model

Neural Network 1:
Deep Convolutional Neural Network

Neural Network 2:
3-Layer Fully-Connected Neural Network

Input:
Character Embedding

Data:
Large Scale Training Dataset

Class:
Semantic Topics

• Zhang et al. 2016
### Document Classification: Hierarchical Model

- **First Layer Network:** Recurrent Neural Network (LSTM, GRU)
- **Second Layer Network:** Recurrent Neural Network (LSTM, GRU)
- **Attention:** Can be Employed between Two Layers

- Tang et al. 2015
- Lai et al. 2015
- Yang et al. 2016
Key Observations

• CNN models are used for both sentence classification and document classification (Kim’14, Blunsom et al.’14, Johnson & Zhang’14, Zhang et al.’15)
• Input can be continuous word embedding (e.g., Kim), discrete word embedding (Johnson & Zhang’14), and even character level embedding (Zhang et al.’15)
• Two-layer models are used for document classification (Yang et al.’16)
• Bag-of-words models work better than syntax aware models (Iyyer et al.’15)
References

• Y. Kim. Convolutional Neural Networks for Sentence Classification. *EMNLP 2014.*
Structured Prediction
Structured Prediction

• Tasks
  – **Search**: named entity recognition in query and document
  – **Question Answering**: named entity recognition in question and answer

• Approaches
  – CNN
  – Sequence-to-Sequence Learning
  – Neural Network based Parsing
Structured Prediction: CNN

Classifier at Each Position:
- Softmax

Neural Network:
- Convolutional Neural Network

Sentence Representation

Sentence

Label at Each Position

Classifier

Neural Network

- Collobert et al. 2011
Structured Prediction: Sequence-to-Sequence Learning

Neural Network:
Sequence-to-Sequence Learning Model

Training Data:
Pairs of Sentence and Linearized Parse Tree

E.g.,
John has a dog  \rightarrow  (S (NP NNP)_{NP} (VP VBZ (NP DT NN)_{NP} )_{VP} . )_{S}

• Vinyals et al. 2015
Structured Prediction: Neural Network based Parsing

**Parser:**
Transition-based Dependency Parser, Constituency Parser, CRF Parser

**Neural Network:**
Deep Neural Networks

**Training Data:**
Pairs of Sentence and Parse Tree

- Chen & Manning, 2014
- Durrett & Klein, 2015
- Zhou et al., 2015
- Andor et al., 2016
Key Observations

• Simplest approach is to employ shallow CNN (Collobert et al.’11)

• Sequence to sequence learning can be employed, when labeled training data is available (Vinyals et al.’15)

• Neural networks based parsers can achieve state-of-the-art performance (Chen & Manning’14, Andor et al., ’16)
References


References

• T. Watanabe and E. Sumita. Transition-based Neural Constituent Parsing. ACL 2015.
Comparison with State-of-the-Art for Fundamental Problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>Accuracy</th>
<th>Domain Knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matching</td>
<td>DL significantly improves</td>
<td>Little is needed</td>
</tr>
<tr>
<td>Translation</td>
<td>DL significantly improves, with different flavor</td>
<td>Little is needed</td>
</tr>
<tr>
<td>Classification</td>
<td>DL significantly improves</td>
<td>Little is needed</td>
</tr>
<tr>
<td>Structured Prediction</td>
<td>DL is comparable</td>
<td>Little is needed</td>
</tr>
</tbody>
</table>
Part 3: Applications of Deep Learning to IR
Outline of Part 3

- Document Retrieval
- Retrieval-based Question Answering
- Generation-based Question Answering
- Question Answering from Relational Database
- Question Answering from Knowledge Graph
- Multi-Turn Dialogue
- Image Retrieval
Learning to Match for Document Retrieval

Query Document Pairs and Relevance Scores (e.g., click-through data)

Learning System

Web Search Engine

Web Page Index

Matching Model (Ranking Feature)

Ranking of relevant pages
Deep Structured Semantic Model (DSSM)

- Approach: Projection to Latent Space
- DSSM: deep neural network for semantic matching between query and document
- Using click through data as training data
- Tri-letter based word hashing for scalable word representation
System Architecture

conditional probabilities

matching scores

$P(d_1 \mid q)$ $f(q,d_1)$ $P(d_2 \mid q)$ $f(q,d_2)$ $P(d_n \mid q)$ $f(q,d_n)$

topic vectors $(W_4,b_4)$ 128

hidden layers $(W_3,b_3)$ 300

(word hashing $(W_2,b_1)$ 300

(term vectors $(W_1)$ 30k

$q$ 128

$d_1$ 128

$d_2$ 128

$\ldots$ 128

$d_n$ 128

$300k$ 30k

$500k$ 30k

$500k$ 30k

$500k$ 30k

$500k$ 30k
Tri-letter Hashing

Representation in vocabulary

\[
\begin{bmatrix}
0 \\
\vdots \\
\vdots \\
0
\end{bmatrix}
\]

\[
cat = \begin{bmatrix}
1 \\
\vdots \\
\vdots \\
0
\end{bmatrix}
\]

\[|\text{Voc}| = 500K\]

Representation with tri-letters

\[
\begin{bmatrix}
0 \\
\vdots \\
1 \\
\vdots \\
1 \\
\vdots \\
0
\end{bmatrix}
\]

\[
\text{cat} = \begin{bmatrix}
#cat# \\
\vdots \\
\text{at#} \\
\vdots \\
\text{#ca} \\
\vdots \\
\text{cat}
\end{bmatrix}
\]

\[|\text{TriL}| = 30K\]

- Generalizable to unknown words
- Robust to misspelling, inflection
- Very small collision
Experimental Results

- **Experiment**
  - Training: 100 million pairs of query-document title in click-through data
  - Testing: 16K queries each associated with about 15 documents

<table>
<thead>
<tr>
<th>Model</th>
<th>NDCG@1</th>
<th>NDCG@3</th>
<th>NDCG@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM25</td>
<td>30.8</td>
<td>37.3</td>
<td>45.5</td>
</tr>
<tr>
<td>LSA</td>
<td>29.8</td>
<td>37.2</td>
<td>45.5</td>
</tr>
<tr>
<td>Translation Model</td>
<td>33.2</td>
<td>40.0</td>
<td>47.8</td>
</tr>
<tr>
<td>DSSM</td>
<td>36.2</td>
<td>42.5</td>
<td>49.8</td>
</tr>
</tbody>
</table>
Document Retrieval

Severyn & Moschitti 2015
Learning to Rank for Document Retrieval

Query Document Pairs and Relevance Scores

Learning System

Ranking Model

Retrieval System

Index

Ranking of documents

Query
Learning to Rank System Using Neural Network

• Approach: simultaneously learn matching model and ranking model
• Matching model: Projection into Latent Space, Using CNN
• Ranking model: taking matching model output as features, as well as other features, Using DNN
Relation between Matching Model and Ranking Model

Matching Model

Ranking Model

Query-Document Matching Features

Other Query-Document Matching Features (e.g., BM25)
System Architecture

Sentence matrix -> convolution feature maps -> pooled representation -> similarity matching -> join layer -> hidden layer -> softmax

Document: $F_d$
- The cat sat on the mat

Query: $F_q$
- Where was the cat?

Additional features: $X_{\text{feat}}$
Experimental Results

• TREC QA Experiment
  – Training: 53K question answer pairs
  – Test: 13K question answer pairs

<table>
<thead>
<tr>
<th></th>
<th>MAP</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree Edit Model (Parsing)</td>
<td>60.9</td>
<td>69.2</td>
</tr>
<tr>
<td>Tree Kernel</td>
<td>67.8</td>
<td>73.6</td>
</tr>
<tr>
<td>CNN Model</td>
<td>74.6</td>
<td>80.8</td>
</tr>
</tbody>
</table>
Retrieval based Question Answering

Ji et al. 2014
Hu et al. 2014
Retrieval-based Question Answering

Q: What is the population of Hong Kong?
A: It is 7.18 million as in 2013.

Q: How many people are there in Hong Kong?
A: There are about 7 million.

Q: Do you know Hong Kong’s population?

Learning System

Question Answering System

A: There are about 7 million.
Retrieval based Question Answering System

- Question

  - Retrieval
  - Retrieved Questions and Answers
  - Matching
  - Matched Answers
  - Ranking
  - Ranked Answers

  - Index of Questions and Answers
  - Matching Models
  - Ranking Model

  - Online
  - Offline

  - Best Answer
Deep Match CNN - Architecture I

- First represent two sentences as vectors, and then match the vectors
Deep Match CNN
- Architecture II

- Represent and match two sentences simultaneously
- Two dimensional model
Experimental Results

- **Experiment**
  - 4.4 million Weibo data (Chinese)
  - 70% of responses are appropriate as replies

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word Embedding</td>
<td>54.3</td>
</tr>
<tr>
<td>SENNA + MLP</td>
<td>56.5</td>
</tr>
<tr>
<td>Deep Match CNN 1-dim</td>
<td>59.2</td>
</tr>
<tr>
<td>Deep Match CNN 2-dim</td>
<td>62.0</td>
</tr>
<tr>
<td>Whole System</td>
<td>70.0</td>
</tr>
</tbody>
</table>
Generation based Question Answering

Shang et al. 2015
Generation-based Question Answering

Q: What is the population of Hong Kong?
A: It is 7.18 million as in 2013.

Q: How many people are there in Hong Kong?
A: There are about 7 million.

Q: Do you know Hong Kong’s population?
A: It is 7 million
Neural Responding Machine

- Encoding questions to internal representations
- Decoding internal representations to answers
- Using GRU

\[
\begin{align*}
\mathbf{y} &= y_1 y_2 \cdots y_t \\
\mathbf{x} &= x_1 x_2 \cdots x_T \\
\mathbf{c} &= \text{Context Generator} \\
\mathbf{h} &= \text{Encoder} \\
\mathbf{y} &= \text{Decoder} \\
\text{Answer} &= \mathbf{y} \\
\text{Question} &= \mathbf{x}
\end{align*}
\]
$P(y_t | y_1 \cdots y_{t-1}, x) = g(y_{t-1}, s_t, c_t)$

$s_t = f(y_{t-1}, s_{t-1}, c_t)$

$y_t$ is one-hot vector
$s_t$ is hidden state of decoder
$c_t$ is context vector

$g()$ is softmax function, $f()$ is GRU

Similar to attention mechanism in RNN Encoder-Decoder
Encoder

Combination of global and local encoders

Local Encoder

Global Encoder

\[
c_t = \sum_{j=1}^{T} \alpha_{tj} \left[ h^l_j : h^g_T \right], \quad \alpha_{tj} = q(h_j, s_{t-1})
\]

\[c_t\] is context vector, \[\alpha_{tj}\] is weight

\[
\left[ h^l_j : h^g_T \right] \text{ is concatenation of local and global hidden states}
\]

\[h_t = f(x_t, h_{t-1})\]

\[x_t\] is word embedding

\[h_t\] is hidden state of encoder

\[f()\] is GRU
Experimental Results

- **Experiment**
  - Trained with 4.4 million Weibo data (Chinese)
  - 95% of responses are natural, 76% of responses are appropriate as replies

<table>
<thead>
<tr>
<th>Message</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Occupy Central is finally over</td>
<td>Will Lujiazui (finance district in Shanghai) be the next?</td>
</tr>
<tr>
<td>I want to buy a Samsung phone</td>
<td>Let us support our national brand.</td>
</tr>
</tbody>
</table>
Question Answering from Relational Database

Yin et al. 2016
Question Answering from Relational Database

Q: How many people participated in the game in Beijing?
A: 4,200
SQL: `select #_participants, where city=beijing`

Q: When was the latest game hosted?
A: 2012
SQL: `argmax(city, year)`

Q: Which city hosted the longest Olympic game before the game in Beijing?
A: Athens

<table>
<thead>
<tr>
<th>year</th>
<th>city</th>
<th>#_days</th>
<th>#_medals</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>Sydney</td>
<td>20</td>
<td>2,000</td>
</tr>
<tr>
<td>2004</td>
<td>Athens</td>
<td>35</td>
<td>1,500</td>
</tr>
<tr>
<td>2008</td>
<td>Beijing</td>
<td>30</td>
<td>2,500</td>
</tr>
<tr>
<td>2012</td>
<td>London</td>
<td>40</td>
<td>2,300</td>
</tr>
</tbody>
</table>
Neural Enquirer

• Query Encoder: encoding query
• Table Encoder: encoding entries in table
• Five Executors: executing query against table

Conducting matching between question and database entries multiple times

Which city hosted the longest Olympic game before the game in Beijing?

where year($(select year, where host_city=Beijing),
argmax(host_city, #_duration))
- Creating query embedding using RNN
- Creating table embedding for each entry using DNN
Executors

- Five layers, except last layer, each layer has reader, annotator, and memory
- Reader fetches important representation for each row, e.g., city=beijing
- Annotator encodes result representation for each row, e.g., row where city=beijing

Select #_participants where city = beijing
Experimental Results

• Experiment
  – Olympic database
  – Trained with 25K and 100K synthetic data
  – Accuracy: 84% on 25K data, 91% on 100K data
  – Significantly better than SemPre (semantic parser)
  – Criticism: data is synthetic

<table>
<thead>
<tr>
<th></th>
<th>25K Data</th>
<th>100K Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Parser</td>
<td>End-to-End</td>
<td>65.2%</td>
</tr>
<tr>
<td></td>
<td>Step-by-Step</td>
<td>96.4%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Question Answering from Knowledge Graph

Yin et al. 2016
Question Answering from Knowledge Graph

Q: How tall is Yao Ming?
A: He is 2.29m tall and is visible from space.
(Yao Ming, height, 2.29m)

Q: Which country was Beethoven from?
A: He was born in what is now Germany.
(Ludwig van Beethoven, place of birth, Germany)

Q: How tall is Liu Xiang?
A: He is 1.89m tall
**GenQA**

- **Interpreter**: creates representation of question using RNN
- **Enquirer**: retrieves top k triples with highest matching scores using CNN model
- **Generator**: generates answer based on question and retrieved triples using attention-based RNN
- **Attention model**: controls generation of answer

---

**Key idea:**
- Generation of answer based on question and retrieved result
- Combination of neural processing and symbolic processing
• Retaining both symbolic representations and vector representations
• Using question words to retrieve top $k$ triples
• Calculating matching scores between question and triples using CNN model
• Finding best matched triples
• Generating answer using attention mechanism
• At each position, a variable decides whether to generate a word or use the object of top triple
Experimental Results

- Experiment
  - Trained with 720K question-answer pairs (Chinese) associated with 1.1M triples in knowledge-base, *data is noisy*
  - Accuracy = 52%
  - Data is still noisy

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Correct/Wrong</th>
</tr>
</thead>
<tbody>
<tr>
<td>Who wrote the Romance of the Three Kingdoms?</td>
<td>Luo Guanzhong in Ming dynasty</td>
<td>correct</td>
</tr>
<tr>
<td>How old is Stefanie Sun this year?</td>
<td>Thirty-two, he was born on July 23, 1978</td>
<td>wrong</td>
</tr>
<tr>
<td>When will Shrek Forever After be released?</td>
<td>Release date: Dreamworks Pictures</td>
<td>wrong</td>
</tr>
</tbody>
</table>
Multi-turn Dialogue

Wen et al. 2016
Multi-turn Dialogue (Question Answering) System

- Multi-turn Dialogue Data
- Learning System
- Knowledge Base
- Dialogue (Question Answering) System
Neural Network based Multi-turn Dialogue System

• Neural network model for entire task
• Combination of symbolic processing and neural processing
• End-to-end trainable, but modularly connected
  – Intent Network: neural processing
  – Belief Trackers: symbolic and neural processing
  – Database Operator: symbolic processing
  – Policy Network: neural processing
  – Generation Network: neural and symbolic processing
System Architecture

Intent Network

Can I have <v.food>

Generation Network

<v.name> serves great <v.food>

Policy Network

Copy field

Database Operator

Database

MySQL query: "Select * where food=Korean"

Belief Tracker

Korean 0.7
British 0.2
French 0.1

DB pointer

Can I have <v.food>
Functions of Modules

• Intent Network
  – Creating *question representation* from user question, using CNN or LSTM

• Belief Trackers
  – Each database attribute has one tracker
  – Each tracker calculates *probabilities of values for the attribute*, based on dialog so far, using RNN

• Database Operator
  – Querying database using belief trackers’ updates, and obtaining result

• Policy Network
  – Creating *action representation*, based on question representation, belief trackers’ updates, database operator’s update, using DNN

• Generation Network
  – Generating answer, based on action representation, using LSTM
Example of Dialogue

• Human : restaurant in any area that serves vietnamese food
• Machine : thanh binh serves vietnamese food type , is there anything else i can help you with ?
• Human : what is the phone number and post code
• Machine : the number is 01223 362456 and the area code is c.b 3, 0 a.f . is there anything else i can do for you ?
• Human : good bye
• Machine : thank you for using the cambridge restaurant system .
Experimental Results

- **Experiment**
  - Database of 99 restaurants, 6 slots
  - 680 dialogues
  - Human evaluation with 245 dialogues

<table>
<thead>
<tr>
<th></th>
<th>Neural Network</th>
<th>Rule-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success rate</td>
<td>96.95</td>
<td>95.12</td>
</tr>
<tr>
<td>Avg. # of turns</td>
<td>3.95</td>
<td>4.54</td>
</tr>
</tbody>
</table>
Image Retrieval

Ma et al. 2015
Image Retrieval

- A lady in a car
- A man holds a cell phone
- Two ladies are chatting
- Having dinner with friends in a restaurant
Multimodal CNN

- Represent text and image as vectors and then match the two vectors
- Word-level matching, phrase-level matching, sentence-level matching
- CNN model works better than RNN models (state of the art) for text

A dog is catching a ball
**Sentence-level Matching**

- Combing image vector and sentence vector
Word-level Matching Model

- Adding image vector to word vectors

Diagram:
- CNN
- MLP
- Image of a dog catching a ball
- Words: "a", "dog", "is", "catching", "a", "ball"
Experimental Results

• Experiment
  – Trained with 30K Flickr data
  – Outperforming other state-of-the-art models

<table>
<thead>
<tr>
<th></th>
<th>R@1</th>
<th>R@5</th>
<th>R@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNLM-VGG</td>
<td>12.5</td>
<td>37.0</td>
<td>51.5</td>
</tr>
<tr>
<td>DVSA (BRNN)</td>
<td>15.2</td>
<td>37.7</td>
<td>50.5</td>
</tr>
<tr>
<td>NIC</td>
<td>17.0</td>
<td>NA</td>
<td>57.0</td>
</tr>
<tr>
<td>M-RNN-VGG</td>
<td>22.8</td>
<td>50.7</td>
<td>63.1</td>
</tr>
<tr>
<td>M-CNN</td>
<td>26.2</td>
<td>56.3</td>
<td>69.6</td>
</tr>
</tbody>
</table>
References


References (Question Answering)

References (Image Retrieval)


Summary
Summary

• Fundamental IR problems
  – Matching
  – Translation
  – Classification
  – Structured Prediction
• Matching is important issue for IR
• DL can learn better representations for matching and other problems
• Useful DL tools
  – Word Embedding
  – Recurrent Neural Networks
  – Convolutional Neural Networks
Summary (cont’)

• Recent progress made in IR tasks
  – Document Retrieval
  – Retrieval-based Question Answering
  – Generation-based Question Answering
  – Question Answering from Knowledge Graph
  – Question Answering from Database
  – Multi-turn Dialogue
  – Image Retrieval

• DL is particularly effective for hard IR problems
Open Question for Future Research

• How to combine symbolic processing and neural processing
• Advantage of symbolic processing: direct, interpretable, and easy to control
• Advantage of neural processing: flexible, robust, and automatic
• Challenge: difficult to make the combination
Acknowledgement

- We thank Xin Jiang, Xi Zhang, Lin Ma, Jun Xu, Shengxian Wan, Liang Peng for providing references for this tutorial.
Paper of This Tutorial:

hangli.hl@huawei.com
lu.zhengdong@huawei.com
Thank you!