Introduction

Feedforward Neural Networks

Stochastic Gradient Descent

Computational Graph & Backpropagation

Dropout
Outline Part 2

- Word Embeddings
- Recurrent Neural Network
- Some Use Cases in NLP
- Comparisons to Traditional Machine Learning
Word Embeddings

Word (Discrete atomic units) :

\textit{l}e\textit{mon}, \textit{G}o\textit{ogle}, \textit{Beijing}, \textit{apple}, \textit{Apple}, \textit{Uppsala}

$\rightarrow$ Embeddings (Continuous vector representations)

\[ [0.1, 0.2, -0.5], [-0.2, 0.3, 0.4], [0.3, -0.2, 0.1] \]
\[ [0.5, -0.2, 0.1], [-0.1, -0.4, 0.1], [-0.2, 0.1, 0.2] \]......
Word Embeddings

One-hot vectors:

<table>
<thead>
<tr>
<th></th>
<th>[0, 0, 0, 0, 0, 0]</th>
<th>[0, 1, 0, 0, 0, 0]</th>
<th>[0, 0, 1, 0, 0, 0]</th>
<th>[0, 0, 0, 1, 0, 0]</th>
<th>[0, 0, 0, 0, 1, 0]</th>
<th>[0, 0, 0, 0, 0, 1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>lemon</td>
<td>[1, 0, 0, 0, 0, 0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Google</td>
<td>[0, 1, 0, 0, 0, 0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Beijing</td>
<td>[0, 0, 1, 0, 0, 0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>apple</td>
<td>[0, 0, 0, 1, 0, 0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apple</td>
<td>[0, 0, 0, 0, 1, 0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uppsala</td>
<td>[0, 0, 0, 0, 0, 1]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Word Embeddings

Random embeddings:

- lemon: [0.1, 0.2, -0.5]
- Google: [-0.2, 0.3, 0.4]
- Beijing: [0.3, -0.2, 0.1]
- apple: [0.5, -0.2, 0.1]
- Apple: [-0.1, 0.4, 0.1]
- Uppsala: [-0.2, 0.1, 0.2]
Word Embeddings

- Dense Vectors
- Capture the similarities between the words

lemon  Google  Beijing
apple  Apple  Uppsala

Train the word embeddings with large amount of plain text.
Word Embeddings

- Dense Vectors
- Capture the similarities between the words

lemon  Google  Beijing
apple  Apple  Uppsala

Train the word embeddings with large amount of plain text.
Word Embeddings

Skip N-gram (Word2vec)

\[
\frac{1}{T} \sum_{t=1}^{T} \sum_{-c \leq j \leq c, \ j \neq 0} \log p(w_{t+j} | w_t); \quad p(w_O | w_I) = \frac{\exp(v_w' T v_w)}{\sum_{w=1}^{W} \exp(v_w' T v_w)}
\]

(Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." (2013).)
Word Embeddings

(Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." (2013).)
Word Embeddings

<table>
<thead>
<tr>
<th>Rank</th>
<th>Word</th>
<th>L2 Distance</th>
<th>Word</th>
<th>L2 Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Apple</td>
<td>0.0</td>
<td>apple</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>Dell</td>
<td>3.21376</td>
<td>tomato</td>
<td>2.1517</td>
</tr>
<tr>
<td>3</td>
<td>Paramount</td>
<td>3.73771</td>
<td>bean</td>
<td>2.28461</td>
</tr>
<tr>
<td>4</td>
<td>Mac</td>
<td>3.75451</td>
<td>onion</td>
<td>2.32823</td>
</tr>
<tr>
<td>5</td>
<td>Flex</td>
<td>3.95375</td>
<td>potato</td>
<td>2.33879</td>
</tr>
<tr>
<td>6</td>
<td>Link</td>
<td>3.97127</td>
<td>chicken</td>
<td>2.6566</td>
</tr>
<tr>
<td>7</td>
<td>Fox</td>
<td>4.12825</td>
<td>chocolate</td>
<td>2.68221</td>
</tr>
<tr>
<td>8</td>
<td>HP</td>
<td>4.13556</td>
<td>lemon</td>
<td>2.70843</td>
</tr>
<tr>
<td>9</td>
<td>Oracle</td>
<td>4.24255</td>
<td>almond</td>
<td>2.73388</td>
</tr>
<tr>
<td>10</td>
<td>Cream</td>
<td>4.26531</td>
<td>berry</td>
<td>2.77824</td>
</tr>
</tbody>
</table>

(Polyglot, https://sites.google.com/site/rmyeid/projects/polyglot)
Recurrent Neural Network

Feedforward neural network

\[ h = g(Vx + c) \]
\[ \hat{y} = Wh + b \]
Recurrent Neural Network

Recurrent NN

\[
\begin{align*}
h_t &= g(V[x_t; h_{t-1}] + c) \\
\hat{y}_t &= Wh_t + b
\end{align*}
\]
Recurrent Neural Network

Recurrent NN

\[ h_t = g(V[x_t; h_{t-1}] + c) \]
\[ \hat{y}_t = Wh_t + b \]
Recurrent Neural Network

Recurrent NN

-one to one

-one to many

-many to one

-many to many
Recurrent Neural Network

Simple RNN

\[ h_{t-1} \xrightarrow{} A \xrightarrow{} h_t \xrightarrow{} \text{tanh} \xrightarrow{} h_{t+1} \]

\[ X_{t-1} \xrightarrow{} A \xrightarrow{} X_t \xrightarrow{} X_{t+1} \]
Recurrent Neural Network

Long Short Term Memory Cell (LSTM)

The five key Architectural Elements of LSTM

- Input Gate
- Forget Gate
- Cell
- Output Gate
- Hidden state output

\[
\begin{align*}
    i_t &= \sigma (W_{xi} x_t + W_{hi} h_{t-1} + W_{ci} c_{t-1} + b_i) \\
    f_t &= \sigma (W_{xf} x_t + W_{hf} h_{t-1} + W_{cf} c_{t-1} + b_f) \\
    c_t &= f_t c_{t-1} + i_t \tanh (W_{xc} x_t + W_{hc} h_{t-1} + b_c) \\
    o_t &= \sigma (W_{xo} x_t + W_{ho} h_{t-1} + W_{co} c_t + b_o) \\
    h_t &= o_t \tanh (c_t)
\end{align*}
\]
Recurrent Neural Network

LSTM

![LSTM Diagram]
Recurrent Neural Network

Gated Recurrent Unit Cell (GRU)

\[ z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \]

\[ r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \]

\[ \tilde{h}_t = \tanh(W \cdot [r_t \ast h_{t-1}, x_t]) \]

\[ h_t = (1 - z_t) \ast h_{t-1} + z_t \ast \tilde{h}_t \]
Some Use Cases in NLP

- Transition-based Dependency Parsing
- Tagging
- Neural Machine Translation
Transition-based Dependency Parsing

Arc-standard system

(Chen and Manning. "A Fast and Accurate Dependency Parser using Neural Networks." EMNLP. 2014.)
Using feedforward neural network as the classifier for the oracle

(Chen and Manning. "A Fast and Accurate Dependency Parser using Neural Networks." EMNLP. 2014.)
Transition-based Dependency Parsing

Using LSTM RNNs as Buffer, Stack and Action queue.

(Dyer et al. "Transition-Based Dependency Parsing with Stack Long Short-Term Memory." ACL. 2015)
### Transition-based Dependency Parsing

<table>
<thead>
<tr>
<th></th>
<th>Dev</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UAS</td>
<td>LAS</td>
<td>UAS</td>
</tr>
<tr>
<td>Malt (eager)</td>
<td>90.1</td>
<td>88.9</td>
<td>90.1</td>
</tr>
<tr>
<td>MST</td>
<td>92.1</td>
<td>90.8</td>
<td>92.0</td>
</tr>
<tr>
<td>Chen &amp; Manning</td>
<td>92.2</td>
<td>91.0</td>
<td>92.0</td>
</tr>
<tr>
<td>Dyer et al.</td>
<td>93.2</td>
<td>90.9</td>
<td>93.1</td>
</tr>
<tr>
<td>Ma et al.</td>
<td>95.8</td>
<td>93.9</td>
<td>95.9</td>
</tr>
</tbody>
</table>

(Ma et al. (2018))
Tagging

- Part-of-Speech Tagging
- Named Entity Recognition
- Segmentation (tokenisation)
- Semantic Role Labelling
- Transliteration
Tagging

(Ma and Hovy. "End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF." ACL. 2016)
## Tagging

### Named Entity Recognition (CoNLL 2003)

<table>
<thead>
<tr>
<th>Researcher</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chieu and Ng (2002)</td>
<td>88.31</td>
</tr>
<tr>
<td>Florian et al. (2003)</td>
<td>88.76</td>
</tr>
<tr>
<td>Ando and Zhang (2005)</td>
<td>89.31</td>
</tr>
<tr>
<td>Collobert et al. (2011)</td>
<td>89.59</td>
</tr>
<tr>
<td>Huang et al. (2015)</td>
<td>90.10</td>
</tr>
<tr>
<td>Chiu and Nichols (2015)</td>
<td>90.77</td>
</tr>
<tr>
<td>Ratinov and Roth (2009)</td>
<td>90.80</td>
</tr>
<tr>
<td>Lin and Wu (2009)</td>
<td>90.90</td>
</tr>
<tr>
<td>Passos et al. (2014)</td>
<td>90.90</td>
</tr>
<tr>
<td>Lample et al. (2016)</td>
<td>90.94</td>
</tr>
<tr>
<td>Luo et al. (2015)</td>
<td>91.20</td>
</tr>
<tr>
<td>Ma and Hovy (2016)</td>
<td>91.21</td>
</tr>
</tbody>
</table>

(Ma and Hovy. "End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF." ACL. 2016)
Neural Machine Translation

Autoencoder-decoder

Neural Machine Translation

Attention-based NMT

(Luong et al. "Effective approaches to attention-based neural machine translation." EMNLP. 2015)
Neural Machine Translation

Google’s GNMT

(Wu et al. "Google’s Neural Machine Translation System: Bridging
Neural Machine Translation

Attention is all you need.

Figure 1: The Transformer - model architecture.

Neural Machine Translation

Mean of human side-by-side evaluation scores

Deep Learning vs Traditional Machine Learning

Classical NLP

Documents → Language Detection → Pre-processing
- Tokenization (English)
- Tokenization (Spanish)
- Tokenization (Arabic)
- PoS Tagging (English)
- PoS Tagging (Spanish)
- PoS Tagging (Arabic)
- Stopword Removal (English)
- Stopword Removal (Spanish)
- Stopword Removal (Arabic)

... → Modeling
- Feature Extraction (English)
- Feature Extraction (Spanish)
- Feature Extraction (Arabic)
- Modeling (English)
- Modeling (Spanish)
- Modeling (Arabic)
- Inference (English)
- Inference (Spanish)
- Inference (Arabic)

Output
- Sentiment
- Classification
- Entity Extraction
- Translation
- Topic Modelling

Deep Learning-based NLP

Documents → Preprocessing → Dense Embeddings
- Obtained via word2vec, doc2vec, GloVe, etc.

Hidden Layers

Output Units

Output
- Sentiment
- Classification
- Entity Extraction
- Translation
- Topic Modelling

...
Deep Learning vs Traditional Machine Learning

Advantages

- More natural and intuitive
- Fit the data better
- Based on less dependency (Markov) assumptions
- Unrestricted by the size of search space
Deep Learning vs Traditional Machine Learning

Disadvantages

- Heavily rely on parameter settings
- Harder to reproduce
- Randomness
- Requires more powerful machines to build
The End.