Deep Neural Net Approaches for Natural Language Processing

Gary Geunbae Lee
Dept of CSE/AIGS
POSTECH
Artificial Neural Networks (ANN)
Layered Networks

\[ \sum_{i,j} = f(w^1 x + w^2 x + w^3 x + \cdots + w^m x_m) \]

\[ = f(\sum_j w^j x_j) \]
Deep learning Innovation

• Combining Feature Learning and Classification as Unified Framework (※ Learning what to learn, how to learn)

Feature learning aspect of DNN based Image Classification
Vanilla recurrent neural networks (RNNs)

• RNNs have connections from the outputs of previous time steps to inputs of next time steps

• For sequential data, a RNN usually computes hidden state $h_t$ from the previous hidden state $h_{t-1}$ and the input $x_t$
  • $h_t = \sigma(W_h h_{t-1} + W_x x_t + b)$
Vanishing gradient problem

- $h_t = \sigma(W_h h_{t-1} + W_x x_t + b)$

- Let’s assume $\sigma$ is the identity function

$$\frac{\partial J^{(i)}(\theta)}{\partial h^{(j)}} = \frac{\partial J^{(i)}(\theta)}{\partial h^{(i)}} \prod_{j < t \leq i} \frac{\partial h^{(t)}}{\partial h^{(t-1)}}$$

$$= \frac{\partial J^{(i)}(\theta)}{\partial h^{(i)}} \prod_{j < t \leq i} W_h = \frac{\partial J^{(i)}(\theta)}{\partial h^{(i)}} W_h^\ell$$

If all $\frac{\partial h^t}{\partial h^{t-1}} < 1 \Rightarrow \frac{\partial J^t}{\partial h^1} \approx 0$
Long short-term memory networks (LSTMs)

- LSTMs explicitly keep and update cell memory $c(t)$ by
  - Removing the previous cell content $c(t-1)$ by multiplying it with $f(t)$
  - Adding the new cell content $\tilde{c}(t)$ multiplied by $i(t)$
- LSTMs produce output $h(t) = o(t) \odot \text{tanh} c(t)$

$$f(t) = \sigma \left( W_f h(t-1) + U_f x(t) + b_f \right)$$
$$i(t) = \sigma \left( W_i h(t-1) + U_i x(t) + b_i \right)$$
$$o(t) = \sigma \left( W_o h(t-1) + U_o x(t) + b_o \right)$$
$$\tilde{c}(t) = \text{tanh} \left( W_c h(t-1) + U_c x(t) + b_c \right)$$
$$c(t) = f(t) \odot c(t-1) + i(t) \odot \tilde{c}(t)$$
$$h(t) = o(t) \odot \text{tanh} c(t)$$
Gated recurrent units (GRUs)

- GRUs keeps and update $h^{(t)}$ by two gates:
  - Update gate $u^{(t)}$ decides
    - How much the old hidden representation $h^{(t)}$ is removed
    - how much the new hidden representation $\tilde{h}^{(t)}$ is added
  - Reset gate $r^{(t)}$ decides how much old representation $h^{(t)}$ is needed to compute new representation $\tilde{h}^{(t)}$

- GRUs also use less number of gates and have smaller parameters than LSTMs

\[
\begin{align*}
  u^{(t)} &= \sigma \left( W_u h^{(t-1)} + U_u x^{(t)} + b_u \right) \\
  r^{(t)} &= \sigma \left( W_r h^{(t-1)} + U_r x^{(t)} + b_r \right) \\
  \tilde{h}^{(t)} &= \tanh \left( W_h (r^{(t)} \circ h^{(t-1)}) + U_h x^{(t)} + b_h \right) \\
  h^{(t)} &= (1 - u^{(t)}) \circ h^{(t-1)} + u^{(t)} \circ \tilde{h}^{(t)}
\end{align*}
\]
Bidirectional Multi-Layer RNNs

the movie was terribly exciting!
Parallel computing for Deep Learning

- History of parallel/ distributed systems for Deep Learning computing

Google taps 16k computers to look for cats –for Science!

Univ. of Toronto uses 2 GPUs for 1.2M training Images for 1000 classes Image classification (※ ImageNet Large Scale Visual Recognition Challenge)

Stanford uses 12 GPUs for Large-scale Video Classification With Convolutional Neural Networks (※ 10M Youtube video)

Google uses 16K CPU cores for Training 22-layers Deep neural network (※ GoogLeNet, 2014)

Baidu’s Artificial Intelligence Supercomputer Beats Google at Image Recognition
Deep Learning for NLP
Word Vector

- Represent words as vectors

\[ \text{expect} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \\ 0.487 \end{pmatrix} \]
Word Vector

• Distributional semantics: A word’s meaning is given by the words that frequently appear close-by

• “You shall know a word by the company it keeps”

• Word2vec objective function (skip-grams)

\[ J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{-m \leq j \leq m, j \neq 0} \log P(w_{t+j} | w_t; \theta) \]
Contextual word embedding

- A word’s **contextual embedding** must consider its context

\[
\epsilon(\text{plays})
\]

\[
\epsilon(\text{plays} \mid \text{the actor _ a show})
\]

GloVe

Some contextual method

the actor plays a show
**ELMo: Embeddings from Language Model**

- Multi-layer bidirectional LSTM language model

\[
\begin{align*}
R_k &= \{x_k^{LM}, \overrightarrow{h}_{k,j}^{LM}, \overleftarrow{h}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\
&= \{h_{LM}^{k,j} \mid j = 0, \ldots, L\},
\end{align*}
\]

\[
\begin{align*}
h_{LM}^{k,0} &= x_k^{LM} \quad \text{(token representation; GloVe)} \\
h_{LM}^{k,j} &= \overleftarrow{h}_{k,j}^{LM}, \overrightarrow{h}_{k,j}^{LM} \quad \text{(LSTM state)}
\end{align*}
\]

\[
\begin{align*}
\text{ELMo}_k^{\text{task}} &= E(R_k; \Theta^{\text{task}}) = \gamma^{\text{task}} \sum_{j=0}^{L} s_j^{\text{task}} h_{LM}^{k,j}
\end{align*}
\]

\(\gamma^{\text{task}}\): scale (hyper-parameter) \\
\(s_j^{\text{task}}\): weight (learned)
ELMo for MRC

- ELMo as a word embedding
Transformer

- Parallel self-attention
  - Looks at self, and determines where to focus

\[
\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V
\]

Q,K,V – vectors for every word, output attention summed up; head means different Q,K,V vector with different weights

Vaswani, Ashish, et al. "Attention is all you need." NIPS 2017
• Training 1. Masked words prediction
  • 15% of words are [MASK]ed

*GELU: Gaussian error linear unit
Training 2. Next sentence prediction
   - To understand texts more than a sentence

Input = [CLS] the man went to [MASK] store [SEP]
       he bought a gallon [MASK] milk [SEP]

Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]
       penguin [MASK] are flight #less birds [SEP]

Label = NotNext
BERT: Bidirectional Encoder Representations from Transformers

- BERT as universal pre-trained model for NLP
  - BERT requires minimal additional layers and fine-tuning

In pre-training, optimize $L_1(u)$

$u$: Unlabeled dataset

$\Theta$: Model parameters

In fine-tuning, optimize $L_3(c)$

$c$: Labeled dataset

$\lambda$: Hyper-parameter weight

*GPT-2/3: zero/few-shot learning
Sequence labeling

• Sequence labeling is the task of assigning a categorical label to each member of an observed sequence.

• Examples of sequence labeling
  • **Part-of-speech tagging** labels each word with a grammatical category
    - e.g. The | trees | are | ... → DT (determiner) | NNS (plural noun) | VBP (plural verb)
  • **Named entity recognition** locates and classifies named entity in text. It can be tackled by labeling each word with a named entity category
    - e.g Barack | Obama | said | ... → B-PERSON | I-PERSON | O | ...
Bi-directional LSTM-CNNs-CRF

- **Bi-directional LSTMs** encode word embeddings and character representations

- **Conditional random fields** compute the distribution of output sequence
  - Viterbi algorithm is applied during training and decoding
  - The objective is the negative log-likelihood of the output sequence distribution

\[
p(y|z; W, b) = \frac{\prod_{i=1}^{n} \psi_i(y_{i-1}, y_i, z)}{\sum_{y' \in Y(z)} \prod_{i=1}^{n} \psi_i(y'_{i-1}, y'_i, z)}
\]

- **\(z\):** input sequence
- **\(y\):** output sequence
- **\(Y(z)\):** a set of all possible output sequences when given the input sequence \(z\)

[Ma 2016]
Seq2Seq NMT via fixed-length representations

- Encoder RNN compresses input sequence into a fixed-length representation
- Decoder RNN produces output sequence from the representation
  - Each produced output token is fed into the next RNN’s input

[Sutskever 2014]
S2S NMT with attention mechanism

• It’s hard to encode all the information of an input sequence into a fixed-length representation

• We can focus important parts of input sequences for each decoding step by attention mechanism

[Bahdanau 2015]
Dependency parsing

- Dependency parsing is the task of extracting **dependencies** between **head** and **dependent** words from a sentence.

- A dependency is the arrow from a head to a dependent with a grammatical type called **relation** (e.g. nsubj).

- Dependencies show which words depend on (modify or are arguments of) which other words.
Neural transition-based dependency parsing

- Extract features from Stack and Buffer
  - lc/rc: leftmost/rightmost children
- Classify an action by neural networks
  - The objective is the negative log-likelihood of the action distribution

\[
p = \text{softmax}(W_2h)
\]

\[
h = (W_1^w x^w + W_1^t x^t + W_1^l x^l + b_1)^3
\]

Feature extraction

Feedforward neural network-based action classifier

[Chen 2014]
Semantic parsing (weakly supervised)

- Semantic parsing is a task of mapping natural language to programs
- We aim to develop semantic parsers without direct supervision on programs

Natural language
“Greece held its last Summer Olympics in which year?”

Context

<table>
<thead>
<tr>
<th>Year</th>
<th>City</th>
<th>Country</th>
<th>Nations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1896</td>
<td>Athens</td>
<td>Greece</td>
<td>14</td>
</tr>
<tr>
<td>1900</td>
<td>Paris</td>
<td>France</td>
<td>24</td>
</tr>
<tr>
<td>1904</td>
<td>St. Louis</td>
<td>USA</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td>Athens</td>
<td>Greece</td>
<td>201</td>
</tr>
<tr>
<td>2008</td>
<td>Beijing</td>
<td>China</td>
<td>204</td>
</tr>
<tr>
<td>2012</td>
<td>London</td>
<td>UK</td>
<td>204</td>
</tr>
</tbody>
</table>

Program
(map (argmax (filter all-rows (λ (x) (= (string:country x) "greece")))
  index)
  number:year)
• Top-down neural semantic parsing
  • Neural net generate derivation actions
  • Type system reduce search space (entity matching)

Neural Semantic Parsing with Type Constraints for Semi-Structured Tables, Krishnamurthy et al., EMNLP 2017
Iterative Search for Weakly Supervised Semantic Parsing, Dasigi et al., NAACL 2019
code generation by semantic parsing

Abstract Syntax Description Language (ASDL) for Python

**ASDL Grammar**

```
stmt → Expr(expr value)
expr → Call(expr func, expr* args, keyword* keywords)
    | Attribute(expr value, identifier attr)
    | Name(identifier id)
    | Str(string s)
expr* → expr
expr → Name
Name → str
Name → str
GenToken[sorted]
GenToken[\(<\)\n>]
GenToken[my_list]
GenToken[\(<\)\n>]
GenToken with Copy
```

(b)

**Code:**

```
sorted(my_list, reverse=True)
```

*parse tree to verify python code

```
stmt = Select(agg_op? agg, idx column_idx,
               cond_expr* conditions)
cond_expr = Condition(cmp_op op, idx column_idx,
                      string value)
agg_op = Max | Min | Count | Sum | Avg
cmp_op = Equal | GreaterThan | LessThan | Other
```

ASDL for SQL

[Yin 2017, Yin 2018, Rabinovich 2017]
Sentiment Analysis

- XLNet based classification

*XLNet = GPT (AR)+BERT(AE): permutation AR (Transformer-XL)
ConvNet for NLP

RNN for NLP - softmax is often only calculated at the last step

CNN for NLP
ConvNet for NLP

- CNN architecture for sentence classification

MRC-QA: SQuAD2.0

- Unanswerable question (negative example)
  - Relevant to the topic
  - Existence of plausible answers

**Article**: Endangered Species Act

**Paragraph**: “…Other legislation followed, including the Migratory Bird Conservation Act of 1929, a 1937 treaty prohibiting the hunting of right and gray whales, and the Bald Eagle Protection Act of 1940. These later laws had a low cost to society—the species were relatively rare—and little opposition was raised.”

**Question 1**: “Which laws faced significant opposition?”
**Plausible Answer**: later laws

**Question 2**: “What was the name of the 1937 treaty?”
**Plausible Answer**: Bald Eagle Protection Act

Figure 1: Two unanswerable questions written by crowdworkers, along with plausible (but incorrect) answers. Relevant keywords are shown in **blue**.
SQuAD2.0

- BERT – no answer prediction
- NLP enjoys rapid progress in the last 10 years due to deep learning.

- Even more rapid progress in the last few years due to larger models, better usage of unlabeled data

- NLP is reaching the point of having big social impact, making issues like bias and security increasingly important.

- Big model, big computation resources, huge training times are problematic, need to focus more light way of doing NLP (even embedded model)