CSE 5525: Foundations of Speech and Language Processing

Pre-trained Models & Machine Translation
Huan Sun (CSE@OSU)

Slides were largely adapted from Prof. Greg Durrett @ UT Austin.
Pre-training
Pre-training on text data

NSP: next sentence prediction
LM: language model

Devlin et al., 2019
https://arxiv.org/abs/1810.04805
Semi-supervised Sequence Learning
context2Vec
Pre-trained seq2seq

ULMFiT
Transformer

ELMo
Bidirectional LM

MultiFiT
Larger model
More data

Cross-lingual

Transformer-XL

Multi-task

Knowledge distillation

Generation

Permutation LM

Transformer-XL

More data

Knowledge Graph

Multi-lingual

Generation

ERNE (Tsinghua)

Cross-modal

Whole-Word Masking

ERNE (Baidu)

BERT-wmm

By Xiaozhi Wang & Zhengyan Zhang @THUNLP

Pretrained language models

https://github.com/thunlp/PLMpapers
A pre-training/fine-tuning paradigm to relational Web tables and test the general model on 6 table-focused tasks [VLDB'21]
How about semi-structured data, like relational web tables?

List of largest technology companies by revenue

From Wikipedia, the free encyclopedia

Companies are ranked by total revenues for their respective fiscal years ended on or before March 31, 2019.[1] All data in the table is taken from the Fortune Global 500 list of technology sector companies for 2019[2] unless otherwise specified.

2019 list  [ edit ]

<table>
<thead>
<tr>
<th>Rank</th>
<th>Company</th>
<th>Fiscal year ending</th>
<th>Revenue (S$) USD</th>
<th>Employees</th>
<th>Headquarters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Apple Inc.</td>
<td>2019</td>
<td>$265,065</td>
<td>132,000</td>
<td>Cupertino, California, US</td>
</tr>
<tr>
<td>2</td>
<td>Samsung Electronics</td>
<td>2019</td>
<td>$221,679</td>
<td>309,630</td>
<td>Suwon, South Korea</td>
</tr>
<tr>
<td>3</td>
<td>Foxconn</td>
<td>2019</td>
<td>$175,617</td>
<td>667,680</td>
<td>New Taipei City, Taiwan</td>
</tr>
<tr>
<td>4</td>
<td>Alphabet Inc.</td>
<td>2019</td>
<td>$130,819</td>
<td>98,771</td>
<td>Mountain View, California, US</td>
</tr>
<tr>
<td>5</td>
<td>Microsoft</td>
<td>2019</td>
<td>$110,360</td>
<td>131,000</td>
<td>Redmond, Washington, US</td>
</tr>
<tr>
<td>6</td>
<td>Huawei</td>
<td>2019</td>
<td>$109,030</td>
<td>188,000</td>
<td>Shenzhen, China</td>
</tr>
<tr>
<td>7</td>
<td>Deji Technologies</td>
<td>2019</td>
<td>$90,621</td>
<td>157,000</td>
<td>Round Rock, Texas, US</td>
</tr>
<tr>
<td>8</td>
<td>Hitachi</td>
<td>2019</td>
<td>$85,507</td>
<td>295,941</td>
<td>Tokyo, Japan</td>
</tr>
<tr>
<td>9</td>
<td>IBM</td>
<td>2019</td>
<td>$79,691</td>
<td>381,100</td>
<td>Armonk, New York, US</td>
</tr>
<tr>
<td>10</td>
<td>Sony</td>
<td>2019</td>
<td>$78,167</td>
<td>114,400</td>
<td>Tokyo, Japan</td>
</tr>
<tr>
<td>11</td>
<td>Panasonic</td>
<td>2019</td>
<td>$72,179</td>
<td>271,869</td>
<td>Osaka, Japan</td>
</tr>
<tr>
<td>12</td>
<td>Intel</td>
<td>2019</td>
<td>$70,848</td>
<td>107,400</td>
<td>Santa Clara, California, US</td>
</tr>
<tr>
<td>13</td>
<td>HP Inc.</td>
<td>2019</td>
<td>$58,472</td>
<td>55,000</td>
<td>Palo Alto, California, US</td>
</tr>
<tr>
<td>14</td>
<td>Facebook Inc.</td>
<td>2019</td>
<td>$55,838</td>
<td>35,567</td>
<td>Menlo Park, California, US</td>
</tr>
<tr>
<td>15</td>
<td>LG Electronics</td>
<td>2019</td>
<td>$55,767</td>
<td>72,600</td>
<td>Seoul, South Korea</td>
</tr>
<tr>
<td>16</td>
<td>Lenovo Group</td>
<td>2019</td>
<td>$51,037</td>
<td>87,000</td>
<td>Quarry Bay, Hong Kong[3]</td>
</tr>
</tbody>
</table>

Challenges

1. How to model the table meta-data and table body at the same time?

2. How to model the row-and-column structure in the table?

3. How to learn the relational knowledge in Web tables in pre-training?

4. How to effectively apply the pre-trained model in downstream tasks?
TURL: Table Understanding via Representation Learning

1. A structure-aware transformer for table encoding
2. Two pretraining objectives for learning factual knowledge from web tables
TURL: Table Understanding via Representation Learning

1. A structure-aware transformer for table encoding
2. Two pretraining objective for learn factual knowledge from web tables
3. Finetune and evaluate on 6 table-focused tasks
After pre-processing, 570171 / 5036 / 4964 tables for pre-training / validation / testing

• We use type and position embeddings to represent different types of the table
• Reuse pre-trained embeddings when possible
• Each entity has a unique entity embedding and one mention embedding which is obtained from its surface form in the table
Visibility matrix

- A table is treated as a graph, so each component can only aggregate from its neighbors.
- Apply mask in self-attention to model the row-column structure.
Pre-training objectives:

- **Masked language model (MLM):** learn the construction of table meta-data
- **Masked entity recovery (MER):** learn factual knowledge in tables
  - Mask mention and entity embedding to predict based on context
  - Mask only entity embedding to predict with help of entity mention, like entity linking
6 downstream tasks:

- **Entity Linking**
  - national
  - film
  - year
  - recipient
  - 1967 (15th)
  - Satyajit

- **Column Type Annotation**
  - national
  - film
  - year
  - recipient
  - [15th]
  - Satyajit
  - event
  - person, actor

- **Relation Extraction**
  - national
  - film
  - year
  - recipient
  - [15th]
  - Satyajit
  - award_winner
6 downstream tasks:

Row Population

Cell Filling

Schema Augmentation

[15th], [16th], [17th], [18th]
## Selected results

<table>
<thead>
<tr>
<th>Method</th>
<th>F1</th>
<th>P</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT-based</td>
<td>90.94</td>
<td>91.18</td>
<td>90.69</td>
</tr>
<tr>
<td>TURL + fine-tuning (only table metadata)</td>
<td>92.13</td>
<td>91.17</td>
<td>93.12</td>
</tr>
<tr>
<td>TURL + fine-tuning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w/o table metadata</td>
<td>94.91</td>
<td>94.57</td>
<td>95.25</td>
</tr>
<tr>
<td>w/o learned embedding</td>
<td>93.85</td>
<td>93.78</td>
<td>93.91</td>
</tr>
<tr>
<td></td>
<td>93.35</td>
<td>92.90</td>
<td>93.80</td>
</tr>
</tbody>
</table>

Relation extraction from Web tables
Our Contribution

• Introduce the pre-training/fine-tuning paradigm to relational web tables and related tasks

• A structure-aware transformer encoder to model relational tables and Masked Entity Recovery pretraining objective to learn the semantics as well as the factual knowledge about entities in relational tables.

• A benchmark that consists of 6 different tasks for table understanding.

TURL: Table Understanding via Representation Learning

How about other tasks like table-based semantic parsing or QA?

Structure-Grounded Pretraining for Text-to-SQL, arXiv’20
Machine Translation
Recall: Attention

- For each decoder state, compute weighted sum of input states
- No attn: \( P(y_i|\mathbf{x}, y_1, \ldots, y_{i-1}) = \text{softmax}(W\bar{h}_i) \)
- Weighted sum of input hidden states (vector)
- Some function \( f \) (TBD)

\[
\begin{align*}
\alpha_{ij} &= \frac{\exp(e_{ij})}{\sum_{j'} \exp(e_{ij'})} \\
e_{ij} &= f(\bar{h}_i, h_j) \\
c_i &= \sum_j \alpha_{ij} h_j
\end{align*}
\]
This Lecture

- MT basics, evaluation
- Word alignment
- Language models
- Phrase-based decoders
MT Basics
Trump Pope family watch a hundred years a year in the White House balcony
MT Ideally

- I have a friend => $\exists x$ friend($x$, self) => J'ai un ami
  - J'ai une amie (friend is female)

- May need information you didn’t think about in your representation

- Hard for semantic representations to cover everything

- Everyone has a friend => $\exists x \forall y$ friend($x$, $y$) => Tous a un ami
  - $\forall x \exists y$ friend($x$, $y$)

- Can often get away without doing all disambiguation — same ambiguities may exist in both languages
Key idea: translation works better the bigger chunks you use
Phrase-Based MT

- Key idea: translation works better the bigger chunks you use

- Remember phrases from training data, translate piece-by-piece and stitch those pieces together to translate

  - How to identify phrases? Word alignment over source-target bitext
  
  - How to stitch together? Language model over target language

  - Decoder takes phrases and a language model and searches over possible translations
Key idea: translation works better the bigger chunks you use

Remember phrases from training data, translate piece-by-piece and stitch those pieces together to translate

How to identify phrases? Word alignment over source-target bitext

How to stitch together? Language model over target language

Decoder takes phrases and a language model and searches over possible translations

NOT like standard discriminative models (take a bunch of translation pairs, learn a ton of parameters in an end-to-end way)
Phrase-Based MT

Phrase table $P(f|e)$

Unlabeled English data

Language model $P(e)$

Noisy channel model: combine scores from translation model + language model to translate foreign to English

$P(e|f) \propto P(f|e)P(e)$

“Translate faithfully but make fluent English”
Evaluating MT

- Fluency: does it sound good in the target language?
- Fidelity/adequacy: does it capture the meaning of the original?
Evaluating MT

- Fluency: does it sound good in the target language?
- Fidelity/adequacy: does it capture the meaning of the original?
- BLEU score: geometric mean of 1-, 2-, 3-, and 4-gram precision vs. a reference, multiplied by brevity penalty (penalizes short translations)

\[ \text{BLEU} = \text{BP} \cdot \exp \left( \sum_{n=1}^{N} w_n \log p_n \right) \]

\[ \text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases} \]

Typically \( n = 4, \ w_i = 1/4 \)

- \( r = \) length of reference
- \( c = \) length of prediction
Evaluating MT

- Fluency: does it sound good in the target language?
- Fidelity/adequacy: does it capture the meaning of the original?
- BLEU score: geometric mean of 1-, 2-, 3-, and 4-gram precision vs. a reference, multiplied by brevity penalty (penalizes short translations)

\[
\text{BLEU} = \text{BP} \cdot \exp \left( \sum_{n=1}^{N} w_n \log p_n \right). 
\]

Typically \( n = 4, \ w_i = 1/4 \)

\[
\text{BP} = \begin{cases} 
1 & \text{if } c > r \\
\frac{1}{e^{(1-r/c)}} & \text{if } c \leq r 
\end{cases} 
\]

\( r \) = length of reference
\( c \) = length of prediction

- Does this capture fluency and adequacy?
At a *corpus* level, BLEU correlates pretty well with human judgments.
At a corpus level, BLEU correlates pretty well with human judgments

Better methods with human-in-the-loop

BLEU scores + user studies
Word Alignment
Word Alignment

- **Input**: a bitext, pairs of translated sentences
  - nous acceptons votre opinion . ||| we accept your view
  - nous allons changer d’avis ||| we are going to change our minds

- **Output**: alignments between words in each sentence
  - “accept and acceptons are aligned”
1-to-Many Alignments

And₁ the₂ program₃ has₄ been₅ implemented₆

Le₁ programme₂ a₃ été₄ mis₅ en₆ application₇
Word Alignment

- Models $P(f|e)$: probability of “French” sentence being generated from “English” sentence according to a model

- Latent variable model: $P(f|e) = \sum_a P(f, a|e) = \sum_a P(f|a, e)P(a)$

- Correct alignments should lead to higher-likelihood generations, so by optimizing this objective we will learn correct alignments
Decoding
Recall: \( n \)-gram Language Models

\[
P(w) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \ldots
\]

- \( n \)-gram models: distribution of next word is a multinomial conditioned on previous \( n-1 \) words

\[
P(w_i|w_1, \ldots, w_{i-1}) = P(w_i|w_{i-n+1}, \ldots, w_{i-1})
\]

I visited San _____ put a distribution over the next word

\[
P(w|\text{visited San}) = \frac{\text{count(visited San, w)}}{\text{count(visited San)}}
\]

Maximum likelihood estimate of this 3-gram probability from a corpus

- Typically use \(~5\)-gram language models for translation
Phrase-Based Decoding

- **Inputs:**
  - n-gram language model: \( P(e_i|e_1, \ldots, e_{i-1}) \approx P(e_i|e_{i-n-1}, \ldots, e_{i-1}) \)
  - Phrase table: set of phrase pairs \((e, f)\) with probabilities \(P(f|e)\)
  - What we want to find: \(e\) produced by a series of phrase-by-phrase translations from an input \(f\), possibly with reordering:
Phrase lattices are big!

Slide credit: Dan Klein
Phrase-Based Decoding

- **Input**: lo haré rápidamente.
- **Translations**: I’ll do it quickly.
- **Translations**: quickly I’ll do it.
- **Decoding objective (for 3-gram LM)**:

\[
\arg \max_e \left[ \prod_{\langle e, f \rangle} P(f|e) \cdot \prod_{i=1}^{\left| e \right|} P(e_i|e_{i-1}, e_{i-2}) \right]
\]

The decoder...
- tries different segmentations,
- translates phrase by phrase,
- and considers reorderings.

Slide credit: Dan Klein
If we translate with beam search, what state do we need to keep in the beam?

What have we translated so far? \[
\text{arg max}_e \left[ \prod_{\langle \bar{e}, \bar{f} \rangle} P(\bar{f} | \bar{e}) \cdot \prod_{i=1}^{|e|} P(e_i | e_{i-1}, e_{i-2}) \right]
\]

What words have we produced so far?

When using a 3-gram LM, only need to remember the last 2 words!

Koehn (2004)
Monotonic Translation

Mary did not give a slap to the witch

\[
\text{score} = \log [P(\text{Mary}) P(\text{not} \mid \text{Mary}) P(\text{Maria} \mid \text{Mary}) P(\text{no} \mid \text{not})]
\]

LM

TM

In reality: score = $\alpha \log P(\text{LM}) + \beta \log P(\text{TM})$

...and TM is broken down into several features

Koehn (2004)
Non-monotonic translation: can visit source sentence “out of order”

State needs to describe which words have been translated and which haven’t

Big enough phrases already capture lots of reorderings, so this isn’t as important as you think
Training Decoders

score = α \log P(LM) + β \log P(TM)

...and TM is broken down into several feature

- Usually 5-20 feature weights to set, want to optimize for BLEU score which is not differentiable

- MERT (Och 2003): decode to get 1000-best translations for each sentence in a small training set (<1000 sentences), do line search on parameters to directly optimize for BLEU
Moses

- Toolkit for machine translation due to Philipp Koehn + Hieu Hoang
  - Pharaoh (Koehn, 2004) is the decoder from Koehn’s thesis

- Moses implements word alignment, language models, and this decoder, plus *a ton* more stuff
  - Highly optimized and heavily engineered, could more or less build SOTA translation systems with this from 2007-2015

- Next time: results on these and comparisons to neural methods
Takeaways

- Phrase-based systems consist of 3 pieces: aligner, language model, decoder
  - HMMs work well for alignment
  - N-gram language models are scalable and historically worked well
  - Decoder requires searching through a complex state space
- Lots of system variants incorporating syntax