Threshing Word Embeddings

Xiang Lisa Li and Jason Eisner
Pre-trained Word Embeddings

Syntactic Knowledge

ELMo

Bert
How to extract syntax automatically?

Specialize the Embeddings!

for the task (parsing)
I would like the cookies to be sweeter.
I would like the cookies to be sweeter.
What Is Threshing?

Specializing
Word Embeddings

Chaff
Sentiment, Topic, Semantics …

Grain
Syntax
I would like the cookies to be sweeter.
I would like the cookies to be sweeter.
I would like the cookies to be sweeter.
Our specialization technique is easily adaptable by swapping blackboxes!

I would like the cookies to be sweeter.
Pros of Specialization vs. Fine Tuning

• Faster — trains on 100 sentences per second

• Better generalization — few parameters, so harder to overfit.
Accuracy

Information Kept

Worse Accuracy

Best Accuracy

Smaller representation

Bigger representation

Baseline Accuracy

Accuracy
Baseline Accuracy

Bigger representation

Worse Accuracy

Smaller representation

Best Accuracy

Accuracy

Information Kept

Worse
Baseline Accuracy

Worse Accuracy

Smaller representation

Best Accuracy

Accuracy

Information Kept
Baseline Accuracy

Worse Accuracy

Smaller representation

Best Accuracy

Accuracy

Information Kept

Worse
Accuracy

Information Kept

Baseline Accuracy

Bigger representation
Worse Accuracy
Smaller representation

Best Accuracy

16

Accuracy

Information Kept

Baseline Accuracy

Bigger representation
Worse Accuracy
Smaller representation

Best Accuracy

16
Accuracy

Information Kept

Worse

Bigger representation

Smaller representation

Best Accuracy
How exactly does the specialization work?

Information Bottleneck
A blackbox parser scorer

Parser

parse tree

ELMo embeddings

specialized taggings

NO back-prop

back-prop

back-prop

chaff

A blackbox parser scorer
Information Bottleneck

Objective

\[ \mathcal{L} = - \text{I}
\]

\[ \text{mutual information between dependency parses and specialized representation.} \]

\[ \text{mutual information between pre-trained ELMo embeddings and specialized representation.} \]

keep more syntactic information

discard more information

bigger $\beta$ —— compression Oscar happy
\[ \mathcal{L} = -I(\text{chased by the cat}; \cdot) + \beta I(\cdot; \cdot) \]

\[ H(\cdot) \quad \text{constant} \quad \text{need to know} \quad p(\cdot; \cdot) \]

\[ \text{need to know} \quad p(\cdot; \cdot) \quad \text{and} \quad p(\cdot; \cdot) \]

\[ \text{KL}(p(\cdot; \cdot) || p(\cdot; \cdot)) \]
Olivander sold the wand.
Olivander sold the wand.
one word token
For some tokens, we need all 4 dimensions
For some tokens, retaining less information does not hurt parsing.
Example: In English, we don’t care whether an object pronoun is singular or plural.
ELMo Layer-1

IB (stochastic)

one word token

\[ \mu \]

\[ \sigma \]
Q: Is our specialization contextual?

A: YES, because we compress ELMo layer 1, which depends on the context.
ELMo Embeddings

The purchase price

Structured Prediction

part-of-speech taggings  Det  Noun  Noun
ELMo Embeddings

The purchase price

Structured Prediction

part-of-speech taggings

CCG taggings

NP/NN  NN/N  N
<table>
<thead>
<tr>
<th>Tag</th>
<th>ELMo Layer-1</th>
<th>HMM</th>
<th>IB (stochastic)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag 1</td>
<td>DET</td>
<td>0</td>
<td>0.05</td>
</tr>
<tr>
<td>Tag 2</td>
<td>ADJ</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>Tag 3</td>
<td>VERB</td>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>Tag 4</td>
<td>NOUN</td>
<td>0</td>
<td>0.35</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

One word token
ELMo Layer-1

Softmax

IB (stochastic)

Tag 1  0.05
Tag 2  0.1
Tag 3  0.7
Tag 4  0.15

one word token
Olivander sold the wand.
Dependency Parsing

the deep biaffine dependency parser (Dozat and Manning, 2016)
Olivander sold the wand.

reparametrization trick or Gumbel-softmax trick

Sampling

ELMo Layer
Token Encoder
Decoder
Variational Upper Bound

• To minimize the IB objective directly, some terms are intractable.

\[
p(\text{chased by the ci}) \quad \text{and} \quad p(\quad )
\]

• So we instead minimize a variational upper bound.

• Which is what the previous slide estimated by sampling

• For mathematical details, please refer to our paper ;)
Do our specialized tags correlate with traditional POS tags?

Yes!
Results
What do specialized tags look like? [Continuous]

(c) $I(X; T) \approx 0.069$

(b) $I(X; T) \approx 24.3$

(a) ELMo, $I(X; T) = H(X) \approx 400.6$

Too much Compression

Moderate Compression

All Information Kept
What our specialized tags look like? [Discrete]
Do our specialized tags correlate with POS tags?

Yes!

- Our results agree with the intuition that POS keeps information about parsing.
- POS is contextual. Our taggings are contextual as well!!

WAIT!? The word type (without context) should also be a strong predictor of POS.

How do we specialize so that we mostly depend on type information?
1. We could use ELMo Layer-0.

ELMo layer-0 is based on a character-level convolutional network. Thus, it inherently does not contain contextual information. Specializing is to extract from existing embeddings, which is also non-contextual.

2. We could do a softer version — make the specialized tagging depend “mostly” on its word type.

\[
\mathcal{L} = -\mathbf{I}(\text{ver}
\begin{array}{c}
\text{say}
\end{array}
\text{ed}
\begin{array}{c}
\text{by}
\end{array}
\text{the}
\begin{array}{c}
\text{cat}
\end{array}
\text{;}
\text{;}
\) + \beta \cdot \mathbf{I}(\text{Elmo} 
\begin{array}{c}
i
\end{array}
\text{;}
\text{;}
) + \gamma \cdot \sum_{\text{word } i \text{ in sentence}} \mathbf{I}(\text{Elmo} 
\begin{array}{c}
i
\end{array}
\text{;}
\text{;}
\text{layer-0, i}
) \]
Parsing Performance [continuous]

- **ELMo**: train by 1024 dimensional ELMo-layer2
- **PCA**: train by 256 dimensional embeddings after applying PCA to ELMo-layer2
- **MLP**: train a non-linear cleanup layer after the ELMo embeddings jointly with the parser.
- **VIBc**: our method ;)

English wins or ties across 9 languages.

≈ 1.0 point improvement on average
Parsing Performance [discrete]

- POS: train the parser on gold POS tags
- VIBd: Our discrete version.
Parses Performance [continuous]

- **ELMo**: train by 1024 dimensional ELMo-layer2
- **PCA**: train by 256 dimensional embeddings after applying PCA to ELMo-layer2
- **MLP**: train a non-linear cleanup layer after the ELMo embeddings jointly with the parser.
- **VIBc**: our method ;)
- **finetune + mild hyperparameter tuning**
Thanks