Learning text embeddings with recurrent neural language models

Grzegorz Chrupała
Tilburg University
Delft 2014
Linguistic data

- Speech: acoustic signal
- Text: streams of characters/bytes
- In order to understand them, add many layers of annotation.
A great recipe for pulpo a la gallega!

<table>
<thead>
<tr>
<th>EN</th>
<th>ES</th>
</tr>
</thead>
<tbody>
<tr>
<td>A great recipe</td>
<td>for pulpo a la gallega</td>
</tr>
</tbody>
</table>

D A Noun P Noun P D Noun !

[ A [ great recipe ] ]
[ for [ pulpo a la gallega ] ] ]
Most analyses are variations of sequence labeling.
Deeper analyses often build tree or graph structures.
**This talk: sequence labeling**
Traditional NLP

- Supervised learning
- Linear models
- Complex manually engineered features
Some recent developments

1. Language models based on recurrent neural networks (Mikolov)
2. Word embeddings induced from unlabeled data (Collobert & Weston, ...)
3. Recursive autoencoders for composing word-into sentence representations (Socher)
Character-level text representations

This research brings together:

- Linear models for sequences (Conditional Random Fields)
- Simple Recurrent Network (SRN) for language modeling
- Representations/embeddings as features with a twist: at character/byte level
Architecture

- **Inputs**
  - Large amount of raw text
  - Smallish amount of labeled text

- **Process**
  - Build feature extractor by training SRN on raw text (unsupervised)
  - Build sequence labeler by training CRF on labeled data extended with extracted features (supervised)
Simple recurrent networks

Character-level text representations

- Trained to predict next character in sequence
- Hidden layer stores a compressed representation of seen characters
- Encodes generalizations
- Embeds string at each position in a low-dimensional space
Visualizing embeddings

SRN trained on 400,000 bytes of Twitter stream. Some nearest neighbors in 400-dimensional embedding space.

should h  should d  will s  will m  should a
@justth   @neenu   @raven_ @lanae @despic
maybe    u maybe y  cause i  wen i  when i
Tweets randomly generated from the trained SRN language model

@YuszLAL100A なすぎるw w w w wとか従役者についてる...(ゝ＞ゝ)
晒せ 信じに行けていいんだな...。 RT @yaepdrrafa:
@fsch_chany siaa,, dobek taha subus sama kiri kabur
wanak... hahah
なかなかない。
おび
But I'm the good first—Good Chulc
Tasks

- Detect code blocks embedded in natural language text
- Segment text into words and sentences
- **Normalize tweets**

Chrupała, G. (2014). Normalizing tweets with edit scripts and recurrent neural embeddings. ACL.
Tweets and similar user-generated content

- Heterogeneous in style (from slangy to formal)
- Frequent mis- and respellings and abbreviations
- Non-standard vocabulary
- Non-standard syntax
- ...
Normalization

Convert text to a canonical, normalized form

- Expand abbreviations
- Correct spellings
- Replace non-standard words

In hope of making text easier to process/understand for downstream applications
## Normalization examples

<table>
<thead>
<tr>
<th>I will c wat i can do</th>
<th>I will see what I can do</th>
</tr>
</thead>
<tbody>
<tr>
<td>imma jus start puttn it out there</td>
<td>I’m going to just start putting it out there</td>
</tr>
</tbody>
</table>
Noisy-channel model

\[ P(\text{target}|\text{original}) \propto \]

\[ P(\text{original}|\text{target}) \times P(\text{target}) \]

Noise model: which respellings are probable? E.g. dictionaries

Language model: which target strings are probable? Trained on large amounts of target data.
Noisy-channel approach

- Good match for spell-checking formal text
  - Large amounts or proof-read formal newspaper text to train language model on

- What kind of text to use for tweet normalization?
  - Newspaper text?
  - Transcribed spoken language?
Direct approach

\[
\hat{\text{target}} = \arg\max_{\text{target}} P(\text{diff(original, target)}|\text{original})
\]

- \(P\) is a linear-chain Conditional Random Field model
- \(\text{diff(original, target)}\) is a series of string edits which transform original to target
P is trained on labeled examples (original-target pairs)
No explicit target language model
Instead, bring in information from unlabeled original data via features learned with SRN LM on lots of unedited tweets
### Diff: Edit script

<table>
<thead>
<tr>
<th>Input</th>
<th>c</th>
<th>w</th>
<th>a</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diff</strong></td>
<td>DEL</td>
<td>INS(see)</td>
<td>NIL</td>
<td>INS(h)</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>see</td>
<td>w</td>
<td>ha</td>
<td>t</td>
</tr>
</tbody>
</table>

- Each position in input string associated with edit operation.
  - A sequence labeling task
Features

- **Baseline features**: byte n-grams
  \[c\_w\_a\_t\_c\_\_w\_w\_a\_t\_c\_w\_w\_a\_t\_c\_w\_w\_a\_t\_c\_w\_w\_a\_t\_c\_w\_w\_a\_t\_c\_w\]

- **SRN features**
  - SRN trained on 400 MB of raw Twitter feed.
  - Activations of 400 hidden units when network is predicting current byte.
  - Discretized: for 10 most active units, on/off with threshold 0.5.
Dataset

- Tweet normalization dataset from Han and Baldwin 2011
- 549 tweets, with normalized versions
- Only word-to-word transformations
Model versions

- No-op: make no changes
- Doc: train on and label whole tweets
- OOV: train on and label OOV-words
Word error rates

![Graph showing word error rates for different conditions: Noop, Doc, Doc, OOV, OOV. The graph compares ngram and ngram+srn models.]

Grzegorz CHRUPAŁA (TiU)  Text embeddings  Delft 2014  25 / 40
## Compared to previous work

<table>
<thead>
<tr>
<th>Method</th>
<th>WER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO-OP</td>
<td>11.2</td>
</tr>
<tr>
<td>HB-dict</td>
<td>6.6</td>
</tr>
<tr>
<td>GHM-dict</td>
<td>7.6</td>
</tr>
<tr>
<td>S-dict</td>
<td>9.7</td>
</tr>
<tr>
<td>Dict-combo</td>
<td>4.9</td>
</tr>
<tr>
<td>Dict-combo + HB-norm</td>
<td>7.9</td>
</tr>
<tr>
<td>OOV-ONLY NGRAM + SRN (test)</td>
<td>4.8</td>
</tr>
</tbody>
</table>
Where SRN features help

<table>
<thead>
<tr>
<th>9 cont continued</th>
<th>5 gon gonna</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 bro brother</td>
<td>4 congrats congratulations</td>
</tr>
<tr>
<td>3 yall you</td>
<td>3 pic picture</td>
</tr>
<tr>
<td>2 wuz what’s</td>
<td>2 mins minutes</td>
</tr>
<tr>
<td>2 juss just</td>
<td>2 fb facebook</td>
</tr>
</tbody>
</table>
Lemmatization of historical text

(Work in progress with Mike Kestemont)

Wanneer nu die ynnige ziel
wanneer nu de innig ziel

hoer selven in god dus ontszoncken ...
zieh zelf in god dus ontzinken ...

Grzegorz CHRUPAŁA (TiU)  Text embeddings  Delft 2014  28 / 40
Preliminary results

![Error comparison graph]

- noop
- morfette
- crf-ngrams
- crf-ngrams+srn

Error:
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7
It didn’t matter if the faces were male, female or those of children. Eighty-three percent of people in the 30-to-34 year old age range gave correct responses.
Results

<table>
<thead>
<tr>
<th>Language</th>
<th>Baseline</th>
<th>+SRN</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>0.0</td>
<td>0.2</td>
</tr>
<tr>
<td>Dutch</td>
<td>1.4</td>
<td>1.0</td>
</tr>
<tr>
<td>Italian</td>
<td>1.2</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Errors per thousand
F1 error on sentence segmentation

- English: 0.2
- Dutch: 1.4
- Italian: 1.2

Graphs show the performance of Punkt and +SRN models for English, Dutch, and Italian languages.
Where SRN features helped

<table>
<thead>
<tr>
<th>+SRN</th>
<th>prof. Teulings het bleek 0,4 procent per costringerlo al</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T---T.S--------.T--. .T----.TT-.T--------. .T-----T-.T--.</td>
</tr>
<tr>
<td>+SRN</td>
<td>T----.T-------.T--. .T----.T-------.T--.</td>
</tr>
<tr>
<td>+SRN</td>
<td>T----.T---------.T--. .T----.T---.T---------.T--.</td>
</tr>
<tr>
<td>+SRN</td>
<td>T---.T----------.T- T--.T----------.T--.</td>
</tr>
</tbody>
</table>
Java - Convert String to enum

```java
public enum Blah {
    A, B, C, D
}
```

Say I have an enum which is just `public enum Blah { A, B, C, D }` and I would like to find the enum value of a string of for example "A" which would be `Blah.A`. How would it be possible to do this? Is the `Enum.ValueOf()` the method I need? If so, how would I use this?
Labeled examples equivalence

![Graph showing F1 error vs Labeled training set size (MB) with two lines: Baseline and +SRN. The Baseline line is represented by red dots, and the +SRN line is represented by blue dots. The graph shows a decrease in F1 error as the labeled training set size increases.]
Conclusion

Features learned by SRNs when combined with linear sequence models:

- Improve performance
- Or reduce amount of supervision needed
Future

- Work in progress on applications
  - Lemmatization of historical language
  - Code-switching in tweets
- Feature extraction vs integrated recurrent network models
Thank you
<table>
<thead>
<tr>
<th>4</th>
<th>1</th>
<th>one</th>
<th>2</th>
<th>with</th>
<th>h with</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>uu</td>
<td>you</td>
<td>2</td>
<td>toni</td>
<td>te tonight</td>
</tr>
<tr>
<td>2</td>
<td>thx</td>
<td>thanks</td>
<td>2</td>
<td>thiis</td>
<td>this</td>
</tr>
<tr>
<td>2</td>
<td>smh</td>
<td>somehow</td>
<td>2</td>
<td>outta</td>
<td>out</td>
</tr>
<tr>
<td>2</td>
<td>n</td>
<td>in</td>
<td>2</td>
<td>m</td>
<td>am</td>
</tr>
<tr>
<td>2</td>
<td>hmwrk</td>
<td>homework</td>
<td>2</td>
<td>gf</td>
<td>girlfriend</td>
</tr>
<tr>
<td>2</td>
<td>fxckin</td>
<td>fucking</td>
<td>2</td>
<td>dha</td>
<td>the</td>
</tr>
<tr>
<td>2</td>
<td>de</td>
<td>the</td>
<td>2</td>
<td>d</td>
<td>the</td>
</tr>
<tr>
<td>2</td>
<td>bhee</td>
<td>be</td>
<td>2</td>
<td>bb</td>
<td>baby</td>
</tr>
</tbody>
</table>
## Sizes of datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Lang</th>
<th>Labeled</th>
<th>Unlabeled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack</td>
<td>en</td>
<td>10.00M</td>
<td>465.0M</td>
</tr>
<tr>
<td>Tweet</td>
<td>en</td>
<td>0.02M</td>
<td>414.0M</td>
</tr>
<tr>
<td>Elephant</td>
<td>en</td>
<td>0.32M</td>
<td>2.5M</td>
</tr>
<tr>
<td>Elephant</td>
<td>nl</td>
<td>4.30M</td>
<td>43.0M</td>
</tr>
<tr>
<td>Elephant</td>
<td>it</td>
<td>4.30M</td>
<td>39.0M</td>
</tr>
</tbody>
</table>