Unlabeled data in NLP

CS 585, Fall 2017: Introduction to Natural Language Processing
http://people.cs.umass.edu/~brenocon/inlp2017

Brendan O’Connor
College of Information and Computer Sciences
University of Massachusetts Amherst
• Lots of unlabeled data, not much labeled data. How to use the unlabeled data?
• One trick: Learn **lexical** information (distributional/embeddings, first-order co-occurrence, etc.)
• More general ML settings
  • Unsupervised learning
  • Semi-supervised learning
• More general linguistic/knowledge structure settings
  • Relationships or events between entities
• Examples
  • EM algorithm: learning a generative model with latent variables
  • MNB/LDA document clusters, HMMs, translation...
  • Brown word clustering: a weird unsupervised HMM
Expectation-Maximization

• For latent-variable learning situations
  • \( w \): known
  • \( z \): unknown “nuisance” variable: need to infer
  • \( \theta \): want to learn
  • Learning goal: \( \arg\max_\theta P(w | \theta) = \arg\max_\theta \sum_z P(w,z | \theta) \)
  • ... when parameter learning would be easy if only you had \( z \).

• EM is a “meta”-algorithm
  • Initialize parameters.
  • Iterate until convergence (or stop early):
    • (E step): Infer \( Q(z) := P(z | w, \theta) \)
    • (M step): Learn new \( \theta := \arg\max_\theta E_Q[\log P(w,z | \theta)] \)
  • “Bootstrapping” intuition
  • It will converge to a local maximum solution to the original marginal likelihood learning goal
Doc catag/clusterig

**MNB**

\[ y = \text{doc catag} \]
\[ w = \text{doc text} \]

Model \[ P(w, y) = P(y) P(w|y) \]

Sup. Learning:
\[ \max_\theta P(w_{\text{tr}}, y_{\text{tr}}) \]

Unsup. Learning:
\[ \max_\theta \sum_y P(w_{\text{unlab}}, y) \]

\[ P(w_{\text{tr}}, y_{\text{tr}}) \]
\[ P(w_{\text{unlab}}) = \sum_k P(y_{\text{tr}} = k) \cdot P(w_{\text{unlab}}|y_{\text{tr}} = k) \]

**EM:** kinda like K-Means
\[ \text{but... for MNB (or more!)} \]
EM performance

- Guaranteed to find a locally-maximum solution. Guaranteed to converge.
- But can take a while
- Initialization-dependent

Figure 1: Variation in negative log likelihood with increasing iterations for 10 EM runs from different random starting points.

Johnson 2007, “Why doesn’t EM find good HMM POS-taggers?”
Semi-supervised learning with EM

- “Semi-supervised”: combine unlabeled and labeled data
Semi-supervised learning with EM

- "Semi-supervised": combine unlabeled and labeled data
Word embeddings/clusters as features

- Two-phase strategy
  1. Unsupervised learning of word representations (embeddings or clusters)
  2. Use word clusters as features for your small-data supervised model
- Word embeddings in a linear model
  - Turian et al. 2010: they work well in a CRF
  - Scaling issue: since they go alongside binary features
  - (IMO, they work even better in nonlinear models?)
- Or: Word clusters in a linear model

Assume that the embeddings are represented by a matrix $E$:

$$E \leftarrow \sigma \cdot E / \text{stddev}(E)$$

$\sigma$ is a scaling constant that sets the new standard deviation after scaling the embeddings.

Figure 1: Effect as we vary the scaling factor $\sigma$ (Equation 1) on the validation set F1. We experiment with Collobert and Weston (2008) and HLBL embeddings of various dimensionality. (a) Chunking results. (b) NER results.

Figure 2: Effect as we vary the capacity of the word representations on the validation set F1. (a) Chunking results. (b) NER results.

There are capacity controls for the word representations: number of Brown clusters, and number of dimensions of the word embeddings.

In general, it appears that more Brown clusters are better. We would like to induce 10000 Brown clusters, however this would take several months. In Turian et al. (2009), we hypothesized on the basis of solely the HLBL NER curve that higher-dimensional word embeddings would give higher accuracy. Figure 2 shows that this hypothesis is not true. For NER, the C&W curve is almost flat, and we were surprised to find the even 25-dimensional C&W word embeddings work so well. For chunking, 50-dimensional embeddings had the highest validation F1 for both C&W and HLBL. These curves indicates that the optimal capacity of the word embeddings is task-specific.
Application: Social Media POS Tagging

• Any NLP system, starting with POS tagging, needs different models/resources than traditional written English
  • Annotate ~2300 tweets
  • Train word clusters on 56 million tweets, use as features
Hierarchical HMM-based word clustering (“Brown clustering”)

- Only a little labeled data (2374 tweets)
- Lots of unlabeled data (56 million tweets): use for lexical generalization
- Distributional hypothesis:
  “you shall know a word by the company it keeps”
- Unsupervised HMM with hierarchical clusters
  [Percy Liang (2005)’s version of Brown clustering]
- 1000 clusters over 217k word types

http://www.ark.cs.cmu.edu/TweetNLP/cluster_viewer.html
What does it learn?

• Orthographic normalizations

so so -so so- $o /so //so

soo soo soooo soooooo soooooo sooooooo soooooooo soooooooo soooooooo soooooooo soooooooo
soooooooo soooooooo soooooooo soooooooo soooooooo soooooooo soooooooo soooooooo soooooooo
soososo superrrr sooooooooooooooooooo sssoo so0o superrrr so0 soooooooooooooooooooooo sososososo
soooooooo soooooooo soooooooo soooooooo soooooooo soooooooo soooooooo soooooooo soooooooo
#too so0o ss0oo so0
soooooooo soooooooo soooooooo soooooooo sssooo sooooooo sooooooo sooooooo superrrrr
very2 s000 sooooooo sooooooo sooooooo sooooooo sooooooo sooooooo sooooooo
_so_ soooooooo sooooooo sooooooo sooooooo /so/ sssooo sososososo
Emoticons etc.
(Clusters/tagger useful for sentiment analysis: NRC-Canada SemEval 2013, 2014)
**Immediate? future auxiliaries**

- **gonna** ~ “going to”
- **gunna** ~ “gonna”
- **gona** ~ “gonna”
- **guna** ~ “gonna”
- **qonna** ~ “gonna”
- **gonnna** ~ “gonna”
- **ganna** ~ “gonna”
- **qonna** ~ “gonna”
- **gonnae** ~ “gonna”

- **tryna** ~ “trying to”
- **finna** ~ “fixing to”
- **bouta** ~ “about to”

---

Thursday, November 9, 17
### Application: Social Media POS Tagging

<table>
<thead>
<tr>
<th>ikr</th>
<th>smh</th>
<th>he</th>
<th>asked</th>
<th>fir</th>
<th>yo</th>
<th>last</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>O</td>
<td>V</td>
<td>P</td>
<td>D</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>

name so he can add u on

N P O V V O P

fb lololol

^ !

- w fo fa fr fro ov fer fir whit abou aft serie fore fah fuh w/her w/that fron isn agains
- yeah yea nah naw yeahh nooo yeh noo noooo yeaa ikr nvm yeahhh nahn nooooo
- facebook fb itunes myspace skype ebay tumblr bbm flickr aim msn netflix pandora
- smh jk #fail #random #fact smfh #smh #winning #realtalk smdh #dead #justsaying

- "non-standard prepositions"
- "interjections"
- "online service names"
- "hashtag-y interjections"??
Highest-weighted POS–treenode features give hierarchical structure gives multiresolutional generalization

<table>
<thead>
<tr>
<th>Cluster prefix</th>
<th>Tag</th>
<th>Types</th>
<th>Most common word in each cluster with prefix</th>
</tr>
</thead>
<tbody>
<tr>
<td>11101010*</td>
<td>!</td>
<td>8160</td>
<td>lol lmao haha yes yea oh omg aww ah btw wow thanks sorry congrats welcome yay ha hey goodnight hi dear please huh wtf exactly idk bless whatever well ok</td>
</tr>
<tr>
<td>11000*</td>
<td>L</td>
<td>428</td>
<td>i'm im you're we're he's there's its it's</td>
</tr>
<tr>
<td>1110101100*</td>
<td>E</td>
<td>2798</td>
<td>x &lt;3 :d :p :) :o :/</td>
</tr>
<tr>
<td>111110*</td>
<td>A</td>
<td>6510</td>
<td>young sexy hot slow dark low interesting easy important safe perfect special different random short quick bad crazy serious stupid weird lucky sad</td>
</tr>
<tr>
<td>1101*</td>
<td>D</td>
<td>378</td>
<td>the da my your ur our their his</td>
</tr>
<tr>
<td>01*</td>
<td>V</td>
<td>29267</td>
<td>do did kno know care mean hurts hurt say realize believe worry understand forget agree remember love miss hate think thought knew hope wish guess bet have</td>
</tr>
<tr>
<td>11101*</td>
<td>O</td>
<td>899</td>
<td>you yall u it mine everything nothing something anyone someone everyone nobody</td>
</tr>
<tr>
<td>100110*</td>
<td>&amp;</td>
<td>103</td>
<td>or n &amp; and</td>
</tr>
</tbody>
</table>
Clusters help POS tagging

- A little annotation + lots of unlabeled data
- Unsupervised word representation learning (clusters, embeddings) is a crucial technique in NLP