Simple, Effective, Robust Semi-Supervised Learning, Thanks To Google N-grams

Shane Bergsma
Johns Hopkins University
Research Vision

Robust processing of human language requires knowledge beyond what’s in small manually-annotated data sets

Derive knowledge from real-world data:

1) Raw text on the web
2) Bilingual text (words plus their translations)
3) Visual data (labelled online images)
More data is better data

[Banko & Brill, 2001]

Grammar Correction Task
@Microsoft
Search Engines vs. N-grams

• Early web work: Use an Internet search engine to get data
  [Keller & Lapata, 2003]

“Britney Spears”  269,000,000 pages
“Britany Spears”  693,000 pages
Search Engines

• Search Engines for NLP: objectionable?
  – Scientifically: not reproducible, unreliable
    [Kilgarriff, 2007, “Googleology is bad science.”]
  – Practically: Too slow for millions of queries
N-grams

• **Google N-gram Data** [Brants & Franz, 2006]
  – N words in sequence + their count on web
  – A compressed version of all the text on web
    • 24 GB zipped fits on your hard drive
  – Enables better features for a range of tasks
    [Bergsma et al. ACL 2008, IJCAI 2009, ACL 2010, etc.]
Google N-gram Data Version 2

- Google N-grams Version 2 [Lin et al., LREC 2010]
  - Same source as Google N-grams Version 1
  - More pre-processing: duplicate sentence removal, sentence-length and alphabetical constraints

- Includes part-of-speech tags!

flies 1643568 NNS|611646 VBZ|1031922
caught the flies, 11 VBD|DT|NNS|,|11
plane flies really well 10 NN|VBZ|RB|RB|10
How to Create Robust Classifiers using Google N-grams

• Features from Google N-gram corpus:
  – Count(*some N-gram*) in Google corpus

• Open questions:
  1. How well do web-scale N-gram features work when combined with conventional features?
  2. How well do classifiers with web-scale N-gram features perform on new *domains*?

• **Conclusion:** N-gram features are *essential*

[Bergsma, Pitler & Lin, ACL 2010]
Feature Classes

• Lex (lexical features): $x_{\text{Lex}}$
  – Many thousands of binary features indicating a property of the strings to be classified

• N-gm (N-gram count features): $x_{\text{Ngm}}$
  – A few dozen real-valued features for the logarithmic counts of various things

• The classifier:
  \[ h(x) = w \cdot x \]
  \[ x = (x_{\text{Lex}}, x_{\text{Ngm}}) \]
Google N-gram Data (HUGE)

Training Examples (small)

Feature Vectors $x^1, x^2, x^3, x^4$

Machine Learning

Classifier: $h(x)$
Uses of New N-gram Data

• Applications:
  1. Adjective Ordering
  2. Real-Word Spelling Correction
  3. Noun Compound Bracketing

• All experiments: linear SVM classifier, report Accuracy (%)
1. Adjective Ordering

- "green big truck" or "big green truck"?

- Used in translation, generation, etc.
- Not a syntactic issue but a semantic issue:
  - size precedes colour, etc.
Adjective Ordering

• As a classification problem:
  – Take adjectives in alphabetical order
  – Decision: is alphabetical order correct or not?

• Why not just most frequent order on web?
  – 87% for web order but 94% for classifier
Adjective Ordering Features

• Lex features: indicators for the adjectives
  – $\text{adj}_1$ indicated with +1, $\text{adj}_2$ indicated with -1
  – E.g. “big green”

$$\text{big} \quad \downarrow \quad \text{green}$$

$x_{\text{Lex}} = (\ldots, 0, 0, 0, 0, 0, 0, +1, 0, 0, 0, 0, \ldots, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, \ldots)$

Decision: $h_{\text{Lex}}(x_{\text{Lex}}) = w_{\text{Lex}} \cdot x_{\text{Lex}}$

$h_{\text{Lex}}(x_{\text{Lex}}) = w_{\text{big}} - w_{\text{green}}$
Adjective Ordering Features

$W_{\text{big}}$ $W_{\text{green}}$

big green truck
Adjective Ordering Features

$W_{\text{big}} - W_{\text{first}}$

first big storm
Adjective Ordering Features

$W_{first}$  $W_{big}$  $W_{young}$  $W_{green}$  $W_{Canadian}$
Adjective Ordering Features

- **N-gm features:**

  \[\text{Count}(“big green”) \quad \text{Count}(“green big”)
  \]
  \[\text{Count}(“big J.*”) \quad \text{Count}(“green J.*”)
  \]
  \[\text{Count}(“J.* big”) \quad \text{Count}(“J.* green”)…\]

\[x_{\text{N-gm}} = (29K, 200, 571K, 2.5M, …)\]
### Adjective Ordering Results

<table>
<thead>
<tr>
<th>System</th>
<th>Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malouf (2000)</td>
<td>91.5</td>
</tr>
<tr>
<td>web $c(a_1, a_2)$ vs. $c(a_2, a_1)$</td>
<td>87.1</td>
</tr>
<tr>
<td>SVM with N-GM features</td>
<td>90.0</td>
</tr>
<tr>
<td>SVM with LEX features</td>
<td>93.0</td>
</tr>
<tr>
<td>SVM with N-GM + LEX</td>
<td>93.7</td>
</tr>
</tbody>
</table>
In-Domain Learning Curve

Accuracy (%) vs Number of training examples

- N-GM+LEX
- N-GM
- LEX

93.7%
Out-of-Domain Learning Curve

![Graph showing the accuracy of different methods (N-GM+LEX, N-GM, LEX) with respect to the number of training examples. The x-axis represents the number of training examples, ranging from 100 to 1e5, and the y-axis represents the accuracy percentage, ranging from 60 to 100. The graph indicates that N-GM+LEX consistently outperforms N-GM and LEX, especially at larger numbers of training examples.]
2. Real-Word Spelling Correction

- Classifier predicts correct word in context:
  “Let me know weather you like it.”
  “weather” or “whether”
Spelling Correction

• Lex features:
  – Presence of particular words (and phrases) preceding or following the confusable word
Spelling Correction

- N-gm feats: Leverage multiple relevant contexts:
  [Bergsma et al., 2009]

Let me know _
  me know _ you
  know _ you like
  _ you like it

- Five 5-grams, four 4-grams, three 3-grams and two 2-grams span the confusable word
Spelling Correction

• N-gm features:
  – Count("let me know weather you")  5-grams
  – Count("me know weather you like")
  ... 
  – Count("let me know weather")  4-grams
  – Count("me know weather you")
  – Count("know weather you like")
  ... 
  – Count("let me know whether you")  5-grams
  ...
Spelling Correction Results

<table>
<thead>
<tr>
<th>System</th>
<th>Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>66.9</td>
</tr>
<tr>
<td>SVM with N-GM features</td>
<td>95.7</td>
</tr>
<tr>
<td>SVM with LEX features</td>
<td>95.2</td>
</tr>
<tr>
<td>SVM with N-GM + LEX</td>
<td>96.5</td>
</tr>
</tbody>
</table>
In-Domain Learning Curve

![Graph showing the learning curve for different models with increasing number of training examples. The graph plots Accuracy (%) on the y-axis against Number of training examples on the x-axis. The models include N-GM+LEX, N-GM, and LEX.](image)
## Cross-Domain Results

<table>
<thead>
<tr>
<th>Domain</th>
<th>N-gm + Lex</th>
<th>Lex</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-Domain</td>
<td>96.5</td>
<td>95.2</td>
</tr>
<tr>
<td>Literature</td>
<td>91.9</td>
<td>85.8</td>
</tr>
<tr>
<td>Biomedical</td>
<td>94.8</td>
<td>91.0</td>
</tr>
</tbody>
</table>
3. Noun Compound Bracketing

- “... bus driver”
  - female (bus driver)
  - *(female bus) driver
  - (school bus) driver

3-word case is a binary classification: **right** or **left** bracketing
Noun Compound Bracketing

• Lex features:
  – binary features for all words, pairs, and the triple, plus capitalization pattern

[Vadas & Curran, 2007]
Noun Compound Bracketing

• N-gm features, e.g. “female bus driver”
  – Count(“female bus”) → predicts left
  – Count(“female driver”) → predicts right
  – Count(“bus driver”) → predicts right
  – Count(“femalebus”)      [Nakov & Hearst, 2005]
  – Count(“busdriver”)      etc.
In-Domain Learning Curve

![Graph showing the learning curve for different methods: N-GM+LEX, N-GM, and LEX. The graph plots accuracy (%) on the y-axis and the number of labeled examples on the x-axis. The accuracy increases as the number of labeled examples increases for all methods.](image-url)
Out-of-Domain Results

• Without N-grams:
  – A Disaster!
Part 2 Conclusion

• It’s good to mix standard lexical features with N-gram count features (but be careful OOD)
• Domain sensitivity of NLP in general: a very big deal
Part 3: Parsing NPs with conjunctions

1) [dairy and meat] production
2) [sustainability] and [meat production]

yes: [dairy production] in (1)
no: [sustainability production] in (2)

• Our contributions: new semantic features from raw web text and a new approach to using bilingual data as soft supervision

[Bergsma, Yarowsky & Church, ACL 2011]
One Noun Phrase or Two: A Machine Learning Approach

• Classify as either one NP or two using a linear classifier: \( h(x) = \mathbf{w} \cdot x \)

\[ \mathbf{x}_{\text{Lex}} = (\ldots, \text{first-noun}=\textit{dairy}, \ldots \]
\[ \text{second-noun}=\textit{meat}, \ldots \]
\[ \text{first+second-noun}=\textit{dairy+meat}, \ldots ) \]
N-gram Features

[dairy and meat] production
• If there is only one NP, then it is implicitly talking about “dairy production”
  • Count(“dairy production”) in N-gram Data? [High]

sustainability and [meat production]
• If there is only one NP, then it is implicitly talking about “sustainability production”
  • Count(“sustainability production”) in N-gram Data? [Low]
Features for Explicit Paraphrases

Pattern: 3 of 1 and 2

↑Count(production of dairy and meat)

↓Count(production of sustainability and meat)

Pattern: 2 3 and 1

↓Count(meat production and dairy)

↑Count(meat production and sustainability)

New paraphrases extending ideas in [Nakov & Hearst, 2005]
Using Bilingual Data

• Bilingual data: a rich source of paraphrases
  *dairy and meat production* $\Leftrightarrow$ *producción láctea y cárnica*

• Build a classifier which uses *bilingual* features
  – Applicable when we know the translation of the NP
Bilingual “Paraphrase” Features

Pattern: 3 1 ... 2 (Spanish)

Count($producción láctea y cárnica$)

Pattern: 1 ... 3 2 (Italian)

Count($sostenibilità e la produzione di carne$)
Bilingual “Paraphrase” Features

Pattern: 1 - ... 2 3 (Finnish)

Count(maidon- ja lihantuotantoon) unseen
+ Features from Google Data

\[ h(x_m) \]

Training Examples

insurrection and regime change
North and South Carolina
business and computer science
the Bosporus and Dardanelles straits
pollution and transport safety
rockets and mortars attacks
the environment and air transport

+ Features from Translation Data

\[ h(x_b) \]

Training Examples

coal and steel
money
coal and steel
Training Examples

+ Features from Google Data

$h(x_m)$

Training Examples

+ Features from Translation Data

$h(x_b)^1$

Insurrection and regime change
North and South Carolina
business and computer science
the Bosporus and Dardanelles straits
pollution and transport safety
the environment and air transport
Training Examples
- Business and computer science
- The Bosporus and Dardanelles straits
- The environment and air transport

+ Features from Google Data

- Co-Training: [Yarowsky’95], [Blum & Mitchell’98]

- Insurrection and regime change
- North and South Carolina
- Pollution and transport safety

Training Examples
- Coal and steel money
- Rocket and mortar attacks

+ Features from Translation Data
Error rate (%) of co-trained classifiers

Bilingual View
Monolingual View

\[ h(x_b)^i \]

\[ h(x_m)^i \]

Co-training iteration
Error rate (%) on Penn Treebank (PTB)

- Broad-coverage Parsers
- Nakov & Hearst (2005)
- Pitler et al. (2010)
- New Supervised Monoclassifier
- Co-trained Monoclassifier

- 800 PTB training examples
- 800 PTB training examples
- 2 training examples

- $h(x_m)^N$
Conclusion

• Robust NLP needs to look beyond human-annotated data to exploit large corpora

• Size matters:
  – Most parsing systems trained on 1 million words
  – We use:
    • billions of words in bitexts (as soft supervision)
    • trillions of words of monolingual text (as features)
    • online images: hundreds of billions
      (×1000 words each ⇔ a 100 trillion words!)

[See our RANLP 2011, IJCAI 2011 papers]
Questions + Thanks

• Gold sponsors:

  - Google
  - The Center For Language and Speech Processing at the Johns Hopkins University
  - human language technology center of excellence
  - NSERC CRISNG

• Platinum sponsors (collaborators):
  - Kenneth Church (Johns Hopkins), Randy Goebel (Alberta), Dekang Lin (Google), Emily Pitler (Penn), Benjamin Van Durme (Johns Hopkins) and David Yarowsky (Johns Hopkins)