ESSLLI 2013: Computational Morphology
Resource-light Approaches to Morphology

Jirka Hana & Anna Feldman
Overview

1. Linguistica
   - Intro
   - Signatures
   - Process
   - Evaluation & Problems

2. Yarowsky & Wicentowski 2000
   - Intro
   - Similarity measures
   - Combination
   - Resources
   - Problems

3. Schone & Jurafsky 2000
   - Algorithm
   - Candidate affixes
   - Computing semantic vectors
   - Subrules

4. Cucerzan & Yarowsky 2002

Jirka Hana & Anna Feldman
ESSLLI 2013: Computational Morphology
Linguistica

(Goldsmith 2001)

http://linguistica.uchicago.edu/

Learns signatures (paradigms) together with roots they combine with

Completely unsupervised: input = raw text (5K-500K tokens)

Assumes suffix-based morphology
Signatures

- Signatures are sets of suffixes that are used with a given set of stems.

| NULL.ed.ing | betray, betrayed, betraying |
| NULL.ed.ing.s | remain, remained, remaining, remains |
| NULL.s | cow, cows |
| e.ed.ing.es | notice, noticed, noticing, notices |

- Similar to but not the same as paradigms:
  - Includes both derivational and inflectional affixes;
  - Purely corpus based, thus often not complete
    - See NULL.ed.ing vs NULL.ed.ing.s above (the corpus contains remains but no betrays)

- Purely concatenative, so blow/blew would be analyzed as bl + ow/ew (if analyzed at all)
## Top English signatures

<table>
<thead>
<tr>
<th>Rank</th>
<th>Signature</th>
<th>#Stems</th>
<th>Rank</th>
<th>Signature</th>
<th>#Stems</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NULL.ed.ing</td>
<td>69</td>
<td>16</td>
<td>e.es.ing</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>e.ed.ing</td>
<td>35</td>
<td>17</td>
<td>NULL.ly.ness</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>NULL.s</td>
<td>253</td>
<td>18</td>
<td>NULL.ness</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>NULL.ed.s</td>
<td>30</td>
<td>19</td>
<td>e.ing</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>NULL.ed.ing.s</td>
<td>14</td>
<td>20</td>
<td>NULL.ly.s</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>’s.NULL.s</td>
<td>23</td>
<td>21</td>
<td>NULL.y</td>
<td>17</td>
</tr>
<tr>
<td>7</td>
<td>NULL.ly</td>
<td>105</td>
<td>22</td>
<td>NULL.er</td>
<td>16</td>
</tr>
<tr>
<td>8</td>
<td>NULL.ing.s</td>
<td>18</td>
<td>23</td>
<td>e.ed.es.ing</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>NULL.ed</td>
<td>89</td>
<td>24</td>
<td>NULL.ed.er.ing</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>NULL.ing</td>
<td>77</td>
<td>25</td>
<td>NULL.es</td>
<td>16</td>
</tr>
<tr>
<td>11</td>
<td>ed.ing</td>
<td>74</td>
<td>26</td>
<td>NULL.ful</td>
<td>13</td>
</tr>
<tr>
<td>12</td>
<td>’s.NULL</td>
<td>65</td>
<td>27</td>
<td>NULL.e</td>
<td>13</td>
</tr>
<tr>
<td>13</td>
<td>e.ed</td>
<td>44</td>
<td>28</td>
<td>ed.s</td>
<td>13</td>
</tr>
<tr>
<td>14</td>
<td>e.es</td>
<td>42</td>
<td>29</td>
<td>e.ed.es</td>
<td>5</td>
</tr>
<tr>
<td>15</td>
<td>NULL.er.est.ly</td>
<td>5</td>
<td>30</td>
<td>ed.es.ing</td>
<td>5</td>
</tr>
</tbody>
</table>
Process

1. A set of heuristics is used to generate candidate signatures (together with roots they combine with)

2. The MDL metrics is used to accept or reject them
Step 1: Candidate generation – Word segmentation

- Uses heuristics to generate a list of potential affixes:
  - Collect all word-tails up to length six,
  - For each tail $n_1, n_2 \ldots n_k$, compute the following metric (where $N_k$ is the total number of tail of length $k$):
    \[
    C(n_1,n_2\ldots n_k) \frac{\log C(n_1,n_2\ldots n_k)}{N_k} \frac{C(n_1)}{C(n_1)C(n_2)\ldots C(n_k)}
    \]
  - The first 100 top ranking candidates are chosen

- Other heuristics are possible

- Words in the corpus are segmented according to these candidates.

- For each stem collect the list associated suffixes (incl. NULL), i.e., the signature for that stem.

- All signatures associated only with one stem or only with one suffix are dropped.
Step 2: Candidate evaluation

- Not all suggested signatures are useful. They need to be evaluated.
- Use Minimum Description Length to filter them
Minimum description length (MDL)

- Criterion for selecting among models
- Developed by (Rissanen 1989); see also (Kazakov 1997, Marcken 1995)
- According to MDL, the best model is the one which gives the most compact description of the data, including the description of the model itself.
- In our case:
  - A grammar (the model) can be used to compress a corpus.
  - The better the morphological description is, the better the compression is.
- The size of the grammar and corpus is measured in bits.
Evaluation

- Applied to English, French, Italian, Spanish, and Latin.
- Identification of morpheme boundaries in 1000-word corpus
- Evaluated subjectively, because there is no gold standard
- Not always clear where the boundary should be: *abolition* vs. *abol-ish*; *Alexander*, *Alex-is*, *Johnson*; *alumn-i*
- English: precision = 85.9 %; recall = 90.4 %
Problems

- Analyzes only suffixes (easily generalizable to prefixes as well).
- Handling stem-internal changes would require significant overhaul.
- All phonological/graphemic changes accompanying inflection, must be factored into suffixes:
  - English: *hated* (*hate*+*ed*) analyzed as *hat-ed*
  - Russian: *plak-at’* ‘cry_{inf} and *plač-et* ‘cry_{pres.3pl}’ analyzed as *pla-kat’* / *pla-čet’*
- Considers only information contained in individual words and their frequencies. Ignores any contextual information (reflecting syntactical and semantical information).
Linguistica is a strictly concatenative and therefore, it is not suitable for discovering paradigms employing other morphological processes (interfixes, templates, metathesis, deletion, etc.).
Resource-light induction of inflectional paradigms (suffixal and irregular).

Tested on induction of English/Spanish present-past verb pairs

Forms of the same lexeme are discovered using a combination of four measures:

- expected frequency distributions,
- context similarity,
- phonemic/orthographic similarity,
- model of suffix and stem-change probabilities.
Process

1. Estimate a probabilistic alignment between inflected forms.
2. Train a supervised morphological analysis learner on a weighted subset of these aligned pairs.
3. Use the result of Step 2 as either a stand-alone analyzer or a probabilistic scoring component to iteratively refine the alignment in Step 1.
Two forms belong to the same lexeme, when their relative frequency fits the expected distribution.

* sing/sang – 1204/1427 – sing/singed – 1204/9 – singe/singed – 2/9 *

The distribution is approximated by the distribution of regular forms.
Frequency similarity

Two forms belong to the same lexeme, when their relative frequency fits the expected distribution.


The distribution is approximated by the distribution of regular forms.

Works for verbal tense, but sometimes one can expect multimodal distribution.

For example, for nouns, the distribution is different for count nouns, mass nouns, plurale-tantum nouns, currency names, proper nouns, ...

Jirka Hana & Anna Feldman

ESSLLI 2013: Computational Morphology
Context similarity

- Forms of the same lemma have similar selectional preferences
- Related verbs tend to occur with similar subjects/objects.
- Arguments identified by simple regular expressions.
- Neither recall nor precision is perfect, but with a large corpus this is tolerable.
Context similarity

- Forms of the same lemma have similar selectional preferences.
- Related verbs tend to occur with similar subjects/objects.
- Arguments identified by simple regular expressions.
- Neither recall nor precision is perfect, but with a large corpus this is tolerable.

- Works well for verbs, but other POS have much less strict subcategorization requirements.
- Some inflectional categories influence subcategorization, e.g., aspect in Slavic.
Form similarity

Form (phonemic/graphemic) similarity is measured by weighted Levenshtein measure (Levenshtein 1966).
Form similarity

- Form (phonemic/graphemic) similarity is measured by weighted Levenshtein measure (Levenshtein 1966).
- Levenshtein distance (edit distance)
  - Distance between two strings is the minimal number of character substitutions, insertion or deletions
  - Used in many different applications
  - Can be calculated by an efficient dynamic programming algorithm
  - Various modifications exist – additional operations, operations’ cost depend on the modified characters, etc.
Form similarity

- Form (phonemic/graphemic) similarity is measured by weighted Levenshtein measure (Levenshtein 1966).
- Levenshtein distance (edit distance)
  - Distance between two strings is the minimal number of character substitutions, insertion or deletions
  - Used in many different applications
  - Can be calculated by an efficient dynamic programming algorithm
  - Various modifications exist – additional operations, operations’ cost depend on the modified characters, etc.
- Edit cost operate on character clusters
- Four types of clusters are distinguished: V, V+, C, C+
In step $k+1$, a probabilistic generative model is trained on the basis of the analyzer obtained in step $k$.

\[
P(\text{form} \mid \text{root, suffix, pos}) = P(a \rightarrow b \mid \text{root, suffix, pos}) =
\]
\[
P(cb + s \mid ca, +s, \text{pos}) = P(a \rightarrow b \mid ca, +s, \text{pos}) =
\]
\[
\approx \lambda_1 P(a \rightarrow b \mid \text{last}_3(\text{root}), \text{suffix, pos})
\]
\[
+ (1 - \lambda_1) \lambda_2 P(a \rightarrow b \mid \text{last}_2(\text{root}), \text{suffix, pos})
\]
\[
+ (1 - \lambda_2) \lambda_3 P(a \rightarrow b \mid \text{last}_1(\text{root}), \text{suffix, pos})
\]
\[
+ (1 - \lambda_3) \lambda_4 P(a \rightarrow b \mid \text{suffix, pos})
\]
\[
+ (1 - \lambda_4) P(a \rightarrow b)
\]
Combination

- Of the four measures, no single model is sufficiently effective on its own.

**English present-past tense verb pairs:**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>9.8 %</td>
</tr>
<tr>
<td>Levenshtein</td>
<td>31.3%</td>
</tr>
<tr>
<td>Context</td>
<td>28.0 %</td>
</tr>
<tr>
<td>F+L+C</td>
<td>71.6 %</td>
</tr>
<tr>
<td>F+L+C+M</td>
<td>96.5%</td>
</tr>
<tr>
<td>F+L+C+M</td>
<td>99.2%</td>
</tr>
</tbody>
</table>

- Therefore, traditional classifier combination techniques are applied to merge scores of the four models.
**Required resources**

1. List of inflectional categories, each with canonical suffixes.
2. A large unannotated text corpus.
3. A list of the candidate noun, verb, and adjective base forms (typically obtainable from a dictionary).
4. A rough mechanism for identifying the candidate parts of speech of the remaining vocabulary, not based on morphological analysis.
5. A list of consonants and vowels.
6. Optionally, a list of common function words.
7. Optionally, various distance/similarity tables generated by the same algorithm on previously studied (related) languages - used as seed information.
Problems

- Suffix/tail based
  Generalized by (Wicentowski 2004), but no longer unsupervised.
- The “rough” mechanism for identifying POS relies on word-order templates. Good for English, not so much for Polish.
- Other problems mentioned above
Knowledge-free Induction of Morphology using LSA

- unsupervised
- input: a space-separated, unlabeled corpus of English (8M words)
- output: ”conflation sets” of morphologically related words
Their algorithm is divided into four parts:

1. Hypothesize candidate affixes
2. Identify pairs of candidate affixes which may be morphological variants, e.g. \((ed, ing)\) or \((s, NULL)\).
3. Collect contextual information about all word pairs which share these morphologically variant affixes, e.g. \((walked, walking)\) or \((walks, walk)\).
4. Determine “morphologically relatedness” for those word pairs with similar semantics (as defined by their ±50 word context).
5. Insert words into a trie and extract potential affixes by observing those places in the trie where branching occurs.
Hypothesize candidate affixes

Two words $w_1$ and $w_2$ are said to be $p$-similar if and only if:

a. the first $p$ characters of $w_1$ are the same as the first $p$ characters of $w_2$

b. the $p + 1$ characters of $w_1$ and $w_2$ are not the same

Ex. walks and walking are 4-similar, as are walk and walks.

c. These pairs are not 5-similar, by rule (a), and not 3-similar, by rule (b).
Selecting candidate pairs

- Identify all pairs of affixes which descend from the same node (e.g. “s”, NULL) and call these pairs rules.
- Two words which share the same stem and affix rule form a PPMV (pair of potential morphological variants).
- For example, (“car”, “cars”) The ruleset of a rule is the set of all PPMVs that have that rule in common. Here, the ruleset of (“s”, NULL) would be the set “cars/car”, “cares/care” The algorithm finds the ruleset for each rule.
Computing semantic vectors

- Decide which of the rulesets that have been generated contain pairs of words which are semantically related.
- S & J don’t compute cosine scores directly on each vector in the matrix; rather, they first apply singular value decomposition (SVD) to the matrix (aka Latent Semantic Analysis or LSA; Landauer et al. 1988)
- LSA: The matrix is projected (compressed) into a lower $k$-dimensional subspace such that the $k$ dimensions of this new subspace are the $k$ most informative dimensions.
- This results in a matrix of “semantic vectors”.

Comparing semantic vectors

- Determine if a pair of words in a PPMV are semantically related by defining a normalized cosine score, or NCS
- Compute the NCS between two semantic vectors
## Sample normalized cosine scores (NCSs)

<table>
<thead>
<tr>
<th>PPMV</th>
<th>cos</th>
</tr>
</thead>
<tbody>
<tr>
<td>ally/allies</td>
<td>6.5</td>
</tr>
<tr>
<td>car/cars</td>
<td>5.6</td>
</tr>
<tr>
<td>dirty/dirt</td>
<td>2.4</td>
</tr>
<tr>
<td>rating/rate</td>
<td>0.97</td>
</tr>
<tr>
<td>car/cares</td>
<td>-0.14</td>
</tr>
<tr>
<td>car/caring</td>
<td>-0.71</td>
</tr>
<tr>
<td>car/cared</td>
<td>-0.96</td>
</tr>
<tr>
<td>ally/all</td>
<td>-1.3</td>
</tr>
</tbody>
</table>

A score over 2.0 would be rare for a random event.
Ruleset-level Statistics

- Determine if a rule is valid (e.g., ("s", NULL) vs. "e", "age")
- So, compute the NCS for PPMVs of a particular rule
- The NCS scores for invalid PPMVs should be distributed normally
- Calculate \(\Pr(\text{true})\), the probability that a particular ruleset is valid (=non-random)
Consider the rule (“es”, NULL)

- This rule pairs together “car/cares” which have a low NCS.
- This rule is sometimes valid (“church/churches”, “mash/mashes”, “miss”, “misses”)
- The problem is that we have to decide whether the rule (“es”, NULL) is valid based on members of the ruleset and there will be a lot of incorrect (“es”, NULL) matches (“hat/hates”, “cap/capes”, “sit/sites”)...

...So how can we remedy this?
Based on the intuition that these rules are phonological variations of other rules, we might expect to find that the ("es", NULL) rule applies in only specific cases.

If so, there should be specific environments where we’d find that there were higher than average NCS scores:

<table>
<thead>
<tr>
<th>Rule/Subrule</th>
<th>Average</th>
<th>Std Dev</th>
<th># instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>(&quot;es&quot;, NULL)</td>
<td>1.62</td>
<td>2.43</td>
<td>173</td>
</tr>
<tr>
<td>(&quot;ches&quot;, &quot;ch&quot;)</td>
<td>2.20</td>
<td>1.66</td>
<td>32</td>
</tr>
<tr>
<td>(&quot;shes&quot;, &quot;sh&quot;)</td>
<td>2.39</td>
<td>1.52</td>
<td>15</td>
</tr>
<tr>
<td>(&quot;res&quot;, &quot;r&quot;)</td>
<td>-0.69</td>
<td>0.47</td>
<td>6</td>
</tr>
<tr>
<td>(&quot;tes&quot;, &quot;t&quot;)</td>
<td>-0.58</td>
<td>0.93</td>
<td>11</td>
</tr>
</tbody>
</table>
S&J set a threshold (T₅) for determining whether or not to believe that a particular PPMV in a rule set is non-random.

<table>
<thead>
<tr>
<th></th>
<th>(Goldsmith) Linguistica</th>
<th>S&amp;J T₅=0.5</th>
<th>S&amp;J T₅=0.7</th>
<th>S&amp;J T₅=0.85</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>83.0%</td>
<td>85.0%</td>
<td>90.0%</td>
<td>92.6%</td>
</tr>
<tr>
<td>Recall</td>
<td>80.4%</td>
<td>81.8%</td>
<td>79.3%</td>
<td>76.6%</td>
</tr>
<tr>
<td>F-Score</td>
<td>81.6%</td>
<td>83.4%</td>
<td>84.3%</td>
<td>83.9%</td>
</tr>
</tbody>
</table>
Schone & Jurafsky 2001: Knowledge Free Induction of Inflectional Morphology

- Extends the Schone and Jurafsky (2000) work
- Includes additional measures because of the shortcomings of semantics alone.
- (“reusability”, “use”) is labeled as a morphological variant but is discarded since the words are not semantically similar enough.
- (“as”, “a”) is deemed acceptable because, since they appear so frequently, neither has much semantic information, so, in that respect, they are semantically very similar.
- Introduction of bad rules: “ho-/∅” ≠ “pi-/∅” for “hog/pig” which have very similar semantics [81 unique pairs].
Bootstrapping a Multilingual Part-of-speech Tagger in One Person-day

- Bootstrap a fine-grained, broad-coverage POS tagger in a new language using only one person-day of data acquisition effort.

- Resources:
  1. an online or hard-copy pocket-sized bilingual dictionary
  2. a basic library reference grammar
  3. access to an existing monolingual text corpus in the language

- Induce initial lexical POS distributions from English translations in a bilingual dictionary without POS tags.

- Handle irregular, regular and semi-regular morphology through a robust generative model using weighted Levenshtein alignments.

- Induce grammatical gender via global modeling of context window feature agreement

- Interactively train context and lexical prior models for fine-grained POS tag spaces.
English Natural Gender Seeds

Female:
- woman
- queen
- aunt
- ...

Romanian Translations
- femeie
- regina
- matușa

Male:
- man
- king
- uncle
- ...

Context models

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>un</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>o</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>acest</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>... mare</td>
<td>0.6</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Suffix models

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>-e</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>-ie</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>-ge</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>-i</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>-t</td>
<td>0.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Test data: Romanian Nouns

<table>
<thead>
<tr>
<th>(blood)</th>
<th>(M) sânge</th>
<th>(F) plăcere</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.3</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>0.9</td>
<td>0.1</td>
</tr>
</tbody>
</table>

- splits words into morphemes in a hierarchical fashion
- more suitable for agglutinative languages with a large number of morphemes per word
- an HMM is used to add a simple morphotactic model.

- Add a set of 0-10,000 correctly segmented words
- Optimize separate weights for unlabeled and labeled data by using a heldout of 500 correctly segmented words
Table: Results of kohonen:etal:2010 with various size of training data

<table>
<thead>
<tr>
<th>labeled data size</th>
<th>kohonen:etal:2010</th>
<th>Morfessor</th>
<th>soa</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500</td>
<td>600</td>
<td>800</td>
</tr>
<tr>
<td>English</td>
<td>61.1</td>
<td>65.2</td>
<td>65.6</td>
</tr>
<tr>
<td>Finnish</td>
<td>49.1</td>
<td>52.7</td>
<td>54.9</td>
</tr>
</tbody>
</table>
Paramor (monson:2009) is a system for unsupervised acquisition of paradigms from a list of words.

It learns paradigms and a lexicon in several steps.

1. Consider all possible segmentations of words into candidate stems and endings.
2. Creates schemes (partial paradigms with the associated stems) by joining endings that share a large number of associated stems.
3. Similar schemes (as measured by cosine similarity) are merged.
4. Schemes proposing frequent morpheme boundaries not consistent with boundaries proposed by the character entropy measure are