High-Performance, Language-Independent Morphological Segmentation

Sajib Dasgupta and Vincent Ng
Human Language Technology Research Institute
University of Texas at Dallas

Morphology

- Segmentation of words into prefixes, suffixes and roots.

- Example

  unfriendliness

  = un + friend + ly + ness

Goal: Unsupervised Morphology

- **Input:** Unannotated text corpus

Word	Frequency
aback	157
abacus	6
abacuses	3
abalone	77
abandon	2781
abandoned	4696
abandoning	1082
abandonment	378
abandonments	23
abandons	117
…….	…….

- **Output:** Word segmentation

Word	Segmentation
aback	aback
abacus	abacus
abacuses	abacus+es
abalone	abalone
abandon	abandon
abandoned	abandon+ed
abandoning	abandon+ing
abandonment	abandon+ment
abandonments	abandon+ment+s
abandons	abandon+s
…….	…….

Why Unsupervised Morphology?

- Advantages
  - No linguistic knowledge input
  - Language Independent
  - Resource Scarce Languages
    - **Only thing we need is a text corpus!!!**
- We tested on 4 different languages
  - English, Finnish, Turkish, Bengali

Plan for the Talk

- **Basic system**
  - Motivated by **Keshava and Pitler**, the best-performing algorithm in PASCAL-MorphoChallenge, 2005
- **Extensions**
  - Address two problems with the basic system

Problem 1: Over-segmentation

- validate = valid + ate ✓
- candidate = candid + ate ✗
- devalue = de + value ✓
- decrease = de + crease ✗

- Biggest source of errors
- Tough to solve!!

Problem 2: Orthographic Change

- Consider
  - deni**a**l = den**y** (deni) + al
  - stab**il**ity = stab**le** (stabil) + ity
- How to handle **spelling changes**?

Basic System

1. Induce prefixes, suffixes and roots
2. Segment the words using the induced prefixes, suffixes and roots

Inducing Prefixes and Suffixes

- Assumptions (Keshava and Pitler, 2006)
  - ***xy* and *x* in vocabulary ⇒ *y* is a suffix**
  - ***xy* and *y* in vocabulary ⇒ *x* is a prefix**

- Too simplistic!
  - <diverge, diver> ⇒ "ge" is a suffix (**Wrong!**)

- **Solution:**
  - Score the affixes and remove low-scoring affixes.

Scoring the Affixes

- **Scoring metric**
  - $score\ (x) = frequency\ (x) * length\ (x)$

# of different words *x* attaches to

# of characters in *x*

- Retain only affixes whose score > *k*, where *k* is the threshold.

Setting the threshold *k*

- *k* is **dependent on vocabulary size**
- In a larger vocabulary, same affix attaches to larger number of words.
- So, we need to set larger *k* for larger vocabulary system

- For example,
  - English: 50
  - Finnish: 300

  (Finnish vocabulary is almost 6 times larger than that of English)

Basic System

- Induce prefixes, suffixes and **roots**
- Segment the words using the induced morphemes

Inducing Roots

- For each word *w* in the vocabulary, we consider *w* as a root if it is **not divisible** i.e.
  - *w* cannot be segmented as *p*+*r* or *r*+*x*,

    where *p* is a prefix, *x* is a suffix and *r* is a word in the vocabulary

Plan for the Talk

- Basic system
  - Segmentation using automatically induced morphemes
- Two extensions to the basic system
  - **Handling over-segmentation (addresses Problem 1)**
  - Handling orthographic changes (addresses Problem 2)

Over-Segmentation

- “affectionate” = “affection” + “ate”? **Correct attachment**
- “candidate” = “candid” + “ate”? **Incorrect attachment**

- We propose 2 methods:
  - **Relative Word-Root Frequency**
  - Suffix Level Similarity

Relative Frequency

Our hypothesis:

- If a word w can be segmented as $r+a$ or $a+r$, where r is a root and a is an affix, then

  ***correct-attachment* (*w*, *r*, *a*)   *freq* (*w*) < *freq* (*r*)**

- Inflectional or derivational form of a root word occurs less frequently than the root itself
- Examples
  - freq (reopen) < freq (open)   => reopen = re+open
  - **freq (candidate) > freq (candid) => candidate≠candid+ate**

6380

119

How correct is this hypothesis?

- In other words, does all the inflectional or derivational forms of the root words occur less frequently than the root?

- We randomly chose 387 English words which can be segmented into Prefix+Root or Root+Suffix
- For each word we check **Word-Root Frequency Ratio (WRFR)**

	Root+Suffix	Prefix+Root	Overall
#of words	344	34	378
WRFR<1	70.1%	88.2%	71.7%

Relaxing the Hypothesis

- To increase the accuracy of the hypothesis we relax it as follows:

Original Hypothesis:

***correct-attachment* (*w*, *r*, *a*)   WRFR (*w*, *r*) < 1**

Relaxed Hypothesis:

correct-attachment* (*w*, *r*, *a*)   WRFR (*w*, *r*) < *t

- We set the threshold ***t*** to 10 for suffixes and 2 for prefixes

Over Segmentation

- We propose 2 methods:
  - Relative Word-Root Frequency
  - **Suffix Level Similarity**

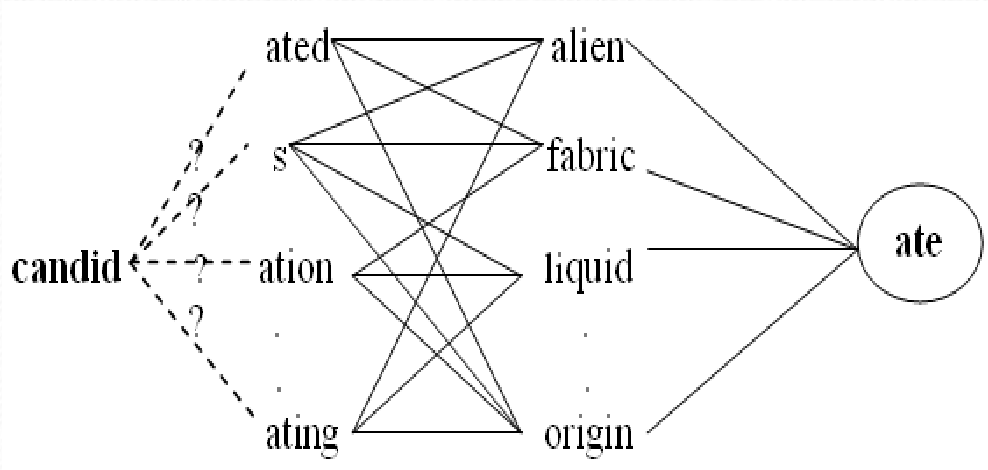
Suffix Level Similarity

Our hypothesis:

- If *w* attaches to a suffix *x*, then *w* should attach to suffixes similar to *x*. i.e.

$$\boldsymbol{w + x \Rightarrow w + \text{Similar}\ (x)}$$

- "candid" + "ate" = "candidate" ?



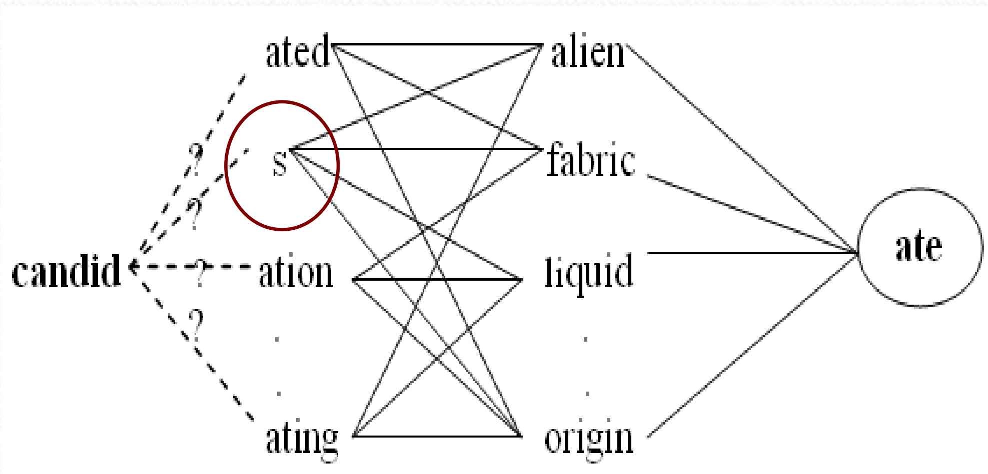
Suffix Level Similarity

Our hypothesis:

- If *w* attaches to a suffix *x*, then *w* should attach to suffixes similar to *x*. i.e.

$$\boldsymbol{w + x \Rightarrow w} + \textbf{Similar}\ (\boldsymbol{x})$$

- “candid” + “ate” = “candidate” ?



Computing similarity between two suffixes

- How to give **more weight** to “ated” than “s”?
- Similarity Metric:

$$Sim(\overset{\boxed{\text{ate}}}{x}, \underset{\boxed{\text{s}}}{y}) = P(y \mid x) * P(x \mid y) = \frac{n}{n_1} * \frac{n}{n_2},$$

where n_1 is the number of words that combine with *x*

n_2 is the number of words that combine with y

n is the number of words that combine with both *x* and *y*

Suffix Level Similarity

- How to use suffix level similarity to check whether r + $\boldsymbol{x}$ is correct?
  - We get 10 most similar suffixes of x according to *Sim*.
  - We scale each suffix's *Sim* value linearly between 1-10.
  - Given 10 similar suffixes $\boldsymbol{x_1}$, $\boldsymbol{x_2}$, … , $\boldsymbol{x_{10}}$ we check

$$\sum_{1}^{10} f_i w_i > t,$$

where f_i is 1 if x_i attaches to r.

w_i is the scaled similarity between suffix x and x_i.

t is a predefined threshold ($t > 0$)

Suffix Level Similarity

- It's **too strict** for words that do not attach to many suffixes.
- We decided to combine WRFR and suffix level similarity to detect incorrect attachments:

$$-\text{WRFR} + \beta * (\textit{suffix level similarity}) < 0,$$

where β is set to be 0.15 for all 4 languages we considered

Plan for the Talk

- Basic system
  - Segmentation using automatically induced morphemes
- Two extensions to the basic system
  - Handling over-segmentation (addresses Problem 1)
  - **Handling orthographic changes (addresses Problem 2)**

Handling Orthographic Changes

- **Goal:** Segment words like “denial” into “deny”+“al”

- **Challenge:**
  - System does not have any **knowledge** of language-specific orthographic rules like

    y:i / _ + al

Handling Orthographic Changes

- Can we **generate** the orthographic change rules **automatically**?

Handling Orthographic Changes

- Can we **generate** the orthographic change rules **automatically**? **Yes, but ….**
- Considering the complexity of the task, we will focus on
  - Change at the morpheme boundary only
  - Change of edit distance 1 (i.e. 1 character insertion or deletion or replacement)
- Our orthographic induction algorithm consists of 3 steps

Step1: Inducing Candidate Allomorphs

- If
  - $\alpha A\beta$ is a word in the vocabulary (e.g. "denial"),
  - β is an induced suffix (e.g. "al"),
  - αB is an induced root (e.g. "deny"),
  - The attachment of β to αB is correct according to relative frequency,

  then αA (e.g. "deni") is an allomorph of αB (e.g. "deny")

- The allomorphs generated from at least **2 different suffixes** are called candidate allomorphs
- Now we have a list of **<candidate allomorph, root,**

Step 2: Inducing Orthographic Rules

- Each <candidate allomorph, root, suffix> tuple is associated with an orthographic rule:
  - From <denial, deny, al> we learn **y:i / _ + al**
  - From <social, sock, al> we learn **k:i / _ + al** **wrong!**
    - So, "social" = "sock" + "al" ?
- We have to **filter** out erroneous rules

Step 3: Filtering the Rules

- **Goal:** For each suffix *x*, remove the low-frequency rules
- Frequency of a rule *r*,

  # of different <allomorph, root, *x*> generate the rule *r*
- **Remove** *r* if it is generated by less than 15% of the tuples.

Step 3: Filtering the Rules

- **Goal:** Filter "morphologically undesirable" rules
- If there are two rules like *A*:*B* / _ + *x* and *A*:*C* / _ + *x*, then the rules are morphologically undesirable, because *A* changes to *B* and *C* under the same environment *x*.

- **To filter morphologically undesirable rules**,
  1. We define the strength of a rule as follows:

$$strength(A:B) = \frac{frequency(A:B)}{\sum_{@} frequency(A:@)}$$

  2. We then keep only those rules *r* whose

frequency* (*r*) * *strength* (*r*) > *t

Setting the threshold *t*

- *t* is **dependent on vocabulary size**

- For example,
  - English: 4
  - Finnish: 25

  (Finnish vocabulary is almost 6 times larger than that of English)

Evaluation

- Results for English and Bengali
- PASCAL Morpho-Challenge results
  - English, Finnish, Turkish

Experimental Setup: Corpora

- English:
  - **WSJ** and **BLLIP**
- Bengali:
  - 5 years of news articles from **Prothom Alo**

Experimental Setup: Test Set Creation

- English:
  - 5000 words from our corpus
  - Correct segmentation given by **CELEX**
- Bengali:
  - 4191 words from our corpus
  - Correct segmentation provided by two linguists

Experimental Setup: Evaluation Metrics

- **Exact accuracy**

- **F-score**

Results

	English				Bengali			
	Acc	**P**	**R**	**F**	**Acc**	**P**	**R**	**F**
Linguistica	68.9	84.8	75.7	80.0	36.3	58.2	63.3	60.6
Morphessor	64.9	69.6	85.3	76.6	56.5	89.7	67.4	76.9
Basic system	68.1	79.4	82.8	81.1	57.7	79.6	81.2	80.4
Relative frequency	74.0	86.4	82.5	84.4	63.2	85.6	79.9	82.7
Suffix level similarity	74.9	88.6	82.3	85.3	66.1	89.7	78.8	83.9
Allomorph detection	78.3	88.3	86.4	87.4	68.3	89.3	81.3	85.1

Results

	English				Bengali			
	Acc	P	R	F	Acc	P	R	F
Linguistica	68.9	84.8	75.7	**80.0**	36.3	58.2	63.3	**60.6**
Morphessor	64.9	69.6	85.3	**76.6**	56.5	89.7	67.4	**76.9**
Basic system	68.1	79.4	82.8	81.1	57.7	79.6	81.2	80.4
Relative frequency	74.0	86.4	82.5	84.4	63.2	85.6	79.9	82.7
Suffix level similarity	74.9	88.6	82.3	85.3	66.1	89.7	78.8	83.9
Allomorph detection	78.3	88.3	86.4	87.4	68.3	89.3	81.3	85.1

Results

	English				Bengali			
	Acc	P	R	F	Acc	P	R	F
Linguistica	68.9	84.8	75.7	**80.0**	36.3	58.2	63.3	60.6
Morphessor	64.9	69.6	85.3	76.6	56.5	89.7	67.4	**76.9**
Basic system	68.1	79.4	82.8	**81.1**	57.7	79.6	81.2	**80.4**
Relative frequency	74.0	86.4	82.5	84.4	63.2	85.6	79.9	82.7
Suffix level similarity	74.9	88.6	82.3	85.3	66.1	89.7	78.8	83.9
Allomorph detection	78.3	88.3	86.4	87.4	68.3	89.3	81.3	85.1

Results

	English				**Bengali**			
	Acc	P	R	F	Acc	P	R	F
Linguistica	68.9	84.8	75.7	80.0	36.3	58.2	63.3	60.6
Morphessor	64.9	69.6	85.3	76.6	56.5	89.7	67.4	76.9
Basic system	68.1	79.4	82.8	**81.1**	57.7	79.6	81.2	**80.4**
Relative frequency	74.0	86.4	82.5	**84.4**	63.2	85.6	79.9	**82.7**
Suffix level similarity	74.9	88.6	82.3	85.3	66.1	89.7	78.8	83.9
Allomorph detection	78.3	88.3	86.4	87.4	68.3	89.3	81.3	85.1

Results

	English				Bengali			
	Acc	P	R	F	Acc	P	R	F
Linguistica	68.9	84.8	75.7	80.0	36.3	58.2	63.3	60.6
Morphessor	64.9	69.6	85.3	76.6	56.5	89.7	67.4	76.9
Basic system	68.1	79.4	82.8	81.1	57.7	79.6	81.2	80.4
Relative frequency	74.0	86.4	82.5	**84.4**	63.2	85.6	79.9	**82.7**
Suffix level similarity	74.9	88.6	82.3	**85.3**	66.1	89.7	78.8	**83.9**
Allomorph detection	78.3	88.3	86.4	87.4	68.3	89.3	81.3	85.1

Results

	English				Bengali			
	Acc	P	R	F	Acc	P	R	F
Linguistica	68.9	84.8	75.7	80.0	36.3	58.2	63.3	60.6
Morphessor	64.9	69.6	85.3	76.6	56.5	89.7	67.4	76.9
Basic system	68.1	79.4	82.8	81.1	57.7	79.6	81.2	80.4
Relative frequency	74.0	86.4	82.5	84.4	63.2	85.6	79.9	82.7
Suffix level similarity	74.9	88.6	82.3	**85.3**	66.1	89.7	78.8	**83.9**
Allomorph detection	78.3	88.3	86.4	**87.4**	68.3	89.3	81.3	**85.1**

Results

	English				Bengali			
	Acc	P	R	F	Acc	P	R	F
Linguistica	**68.9**	84.8	75.7	80.0	36.3	58.2	63.3	60.6
Morphessor	64.9	69.6	85.3	76.6	**56.5**	89.7	67.4	76.9
Basic system	68.1	79.4	82.8	81.1	57.7	79.6	81.2	80.4
Relative frequency	74.0	86.4	82.5	84.4	63.2	85.6	79.9	82.7
Suffix level similarity	74.9	88.6	82.3	85.3	66.1	89.7	78.8	83.9
Allomorph detection	**78.3**	88.3	86.4	87.4	**68.3**	89.3	81.3	85.1

PASCAL Morpho-Challenge Results

- Three languages
  - English
  - Finnish
  - Turkish

Results (F-scores)

	English	Finnish	Turkish
Winner	76.8	64.7	65.3
Our System	79.4	65.2	66.2
Morphessor	66.2	66.4	70.1

Winner for English: Keshava and Pitler's (2006) algorithm

Winner for Finnish and Turkish: Bernhard's (2006) algorithm

Creutz's remark on the PASCAL Challenge:

None of the participants performs well on all 3 languages

Results (F-scores)

	English	Finnish	Turkish
Winner	76.8	64.7	65.3
Our System	79.4	65.2	66.2
Morphessor	66.2	66.4	70.1

Our system outperforms the winners!

- **Robustness across different languages**

Results (F-scores)

	English	Finnish	Turkish
Winner	76.8	64.7	65.3
Our System	79.4	65.2	66.2
Morphessor	66.2	66.4	70.1

Morphessor slightly outperforms our system for Finnish and Turkish, but what about English?

Results (F-scores)

	English	Finnish	Turkish
Winner	76.8	64.7	65.3
Our System	79.4	65.2	66.2
Morphessor	66.2	66.4	70.1

Morphessor performs poorly for English

Conclusion

- Our system shows robust performance across different languages
  - Outperforms Linguistica and Morphessor for English and Bengali
  - Compares favorably to the winners of the PASCAL datasets

Thank you!