A Language-Independent Unsupervised Model for Morphological Segmentation

Vera Demberg

School of Informatics
University of Edinburgh

June 27
ACL 2007, Prague

Why Analyse Words Morphologically?

Motivation

- Decrease data sparseness
- Smaller lexica
- Find relations between words

Applications

- Machine Translation [Goldwater and McClosky, 2005]
- Speech Recognition [Kurimo et al., 2006, Puurula and Kurimo, 2007]
- Text-to-Speech Systems [Möbius, 2001, Sproat, 1996, Taylor, 2005]

Unsupervised?

- Less domain-dependent
- Lower development cost
- Good generalizability to (related?) languages

Why Analyse Words Morphologically?

Motivation

- Decrease data sparseness
- Smaller lexica
- Find relations between words

Applications

- Machine Translation [Goldwater and McClosky, 2005]
- Speech Recognition [Kurimo et al., 2006, Puurula and Kurimo, 2007]
- Text-to-Speech Systems [Möbius, 2001, Sproat, 1996, Taylor, 2005]

Unsupervised?

- Less domain-dependent
- Lower development cost
- Good generalizability to (related?) languages

Why Analyse Words Morphologically?

Motivation

- Decrease data sparseness
- Smaller lexica
- Find relations between words

Applications

- Machine Translation [Goldwater and McClosky, 2005]
- Speech Recognition [Kurimo et al., 2006, Puurula and Kurimo, 2007]
- Text-to-Speech Systems [Möbius, 2001, Sproat, 1996, Taylor, 2005]

Unsupervised?

- Less domain-dependent
- Lower development cost
- Good generalizability to (related?) languages

Overview

Overview

Morphology

Concatenative Processes

e.g. Wortzerlegungen $\longrightarrow$ Wort+zer+leg+ung+en (‘word segmentations’)

- Compounding: Wort+zerlegungen
- Suffixation: zerleg+ung+en
  - inflectional: zerlegung+en
  - derivational: zerleg+ung
- Prefixation: zer+leg
- Others: infixation, circumfixation, reduplication

Non-concatenative Processes

e.g. Wörter $\longrightarrow$ Wort+er (‘words’)

- Ablauting: o $\longrightarrow$ ö
- Others: umlauting, vowel harmony, deletion, insertion

Previous Approaches

Unsupervised algorithms for **concatenative** phenomena

- Letter Successor Variety / Conditional Entropy between letters [Harris, 1955, Hafer and Weiss, 1974, Déjean, 1998, Monson et al., 2004, Bordag, 2006, Bernhard, 2006, Keshava and Pitler, 2006]
- Minimum Description Length [Goldsmith, 2001, Creutz and Lagus, 2006]

Unsupervised algorithms addressing **word-internal variation**

- Phonological Relationships between Related Words [Neuvel and Fulop, 2002, Yarowsky and Wicentowski, 2000]
- Algorithms that take into account syntax and semantics [Schone and Jurafsky, 2000, Yarowsky and Wicentowski, 2000, Jacquemin, 1997]

Comparative Evaluation of Unsupervised Morphologies

Evaluation Results of Morpho Challenge 2005 (F-score):

System	Finnish	Turkish	English
Bordag, 2006	48.3%	57.0%	61.7%
Morfessor 1.0	54.2%	51.3%	66.0%
Morf. Categories-ML	66.4%	**70.7%**	66.2%
Bernhard, 2006	64.7%	65.3%	66.4%
Morf. Categories-ML	**67.0%**	69.2%	69.0%
RePortS	–	–	**76.8%**

RePortS algorithm:

- Very good results for English
- Highly efficient
- Very simple

Overview

1. Background
2. Algorithm
   - Data Structure
   - Identifying Morphemes
   - Segmenting Words
3. Learning Stem Variation
4. Evaluation
   - Evaluation of Modifications
   - Evaluation on G2P task

The RePortS Algorithm [Keshava and Pitler, 2006]

Three steps + added improvements

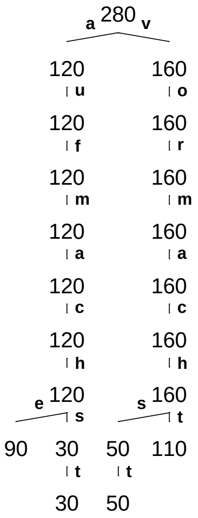
1. Building up data structure
2. Finding affixes
   Finding word stems
3. Segmenting words
   Segmentation ranking with n-gram language model

Step 1: Data Structure

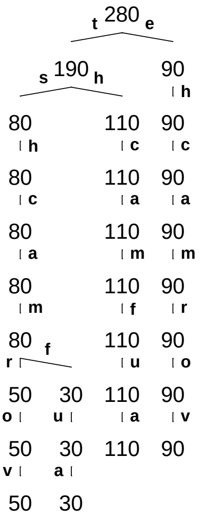
(a) corpus

type	count
⋮	
aufmacht	90
aufmachst	30
vormache	110
vormachst	50
⋮	

(b) forward tree



(c) backward tree

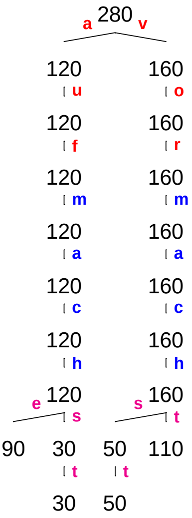


Step 1: Data Structure

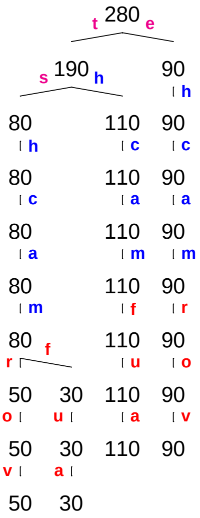
(a) corpus

type	count
⋮	
aufmacht	90
aufmachst	30
vormache	110
vormachst	50
⋮	

(b) forward tree



(c) backward tree



Step 1: Data Structure

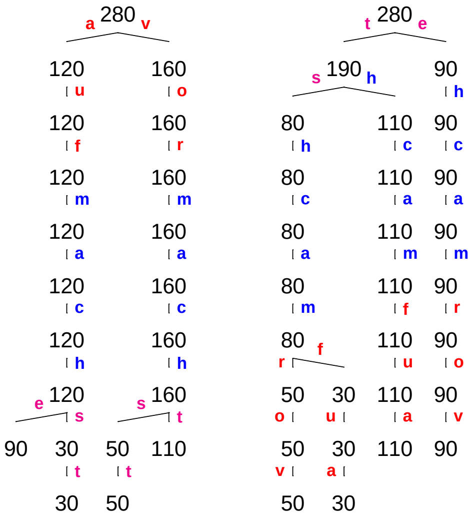
(a) corpus

type	count
⋮	
aufmacht	90
aufmachst	30
vormache	110
vormachst	50
⋮	

$$P_f(h|aufmac) = 120/120 = 1$$
$$P_f(s|aufmach) = 30/120 < 1$$
$$P_b(m|achst) = 80/80 = 1$$
$$P_b(r|machst) = 50/80 < 1$$

(b) forward tree

(c) backward tree



Step 2: Finding Affixes – Original Method

- **Q:** Is there a morpheme boundary between '*A*' and '*B*' in word '$\alpha AB\beta$'?

  example 'working': $\underbrace{wor}_{\alpha}\underbrace{k}_{A}\underbrace{i}_{B}\underbrace{ng}_{\beta}$

- **Algorithm**

find suffix $B\beta$	find prefix αA
1. αA in corpus	1. βB in corpus
2. $P_f(A\|\alpha) \approx 1$	2. $P_b(B\|\beta) \approx 1$
3. $P_f(B\|\alpha A) < 1$	3. $P_b(A\|B\beta) < 1$

Ranking algorithm
`if (cond. satisfied)` *reward!* `else` *punish!*

- **Implicit assumptions**
  - All stems are valid words in the corpus
  - Affixes occur at the beginning or end of words only
  - Affixation does not change stems

  $\longrightarrow$ specific to English

Step 2: Finding Affixes – Original Method

- **Q:** Is there a morpheme boundary between 'A' and 'B' in word '$\alpha AB\beta$'?

  example 'working': $\underbrace{\textit{wor}}_{\alpha}\underbrace{\textit{k}}_{A}\underbrace{\textit{i}}_{B}\underbrace{\textit{ng}}_{\beta}$

- **Algorithm**

find suffix $B\beta$	find prefix αA
1. αA in corpus	1. βB in corpus
2. $P_f(A\|\alpha) \approx 1$	2. $P_b(B\|\beta) \approx 1$
3. $P_f(B\|\alpha A) < 1$	3. $P_b(A\|B\beta) < 1$

Ranking algorithm
`if (cond. satisfied)` *reward!* `else` *punish!*

- **Implicit assumptions**
  - All stems are valid words in the corpus
  - Affixes occur at the beginning or end of words only
  - Affixation does not change stems

  $\longrightarrow$ specific to English

Main Issue: Low Recall for German / Turkish / Finnish

- Remember: assumed that stems coincide with words from corpus
- Does not hold for other languages
- e.g., *abhol* not a German word

German corpus:
⋮
abholst
abholen
abholt
abhole
Abholung
⋮

Idea:

- Find list of stems
- Change first condition from 'αA in corpus' to 'αA in stem list'

Main Issue: Low Recall for German / Turkish / Finnish

- Remember: assumed that stems coincide with words from corpus
- Does not hold for other languages
- e.g., *abhol* not a German word

German corpus:
⋮
abholst
abholen
abholt
abhole
Abholung
⋮

Idea:

- Find list of stems
- Change first condition from 'αA in corpus' to 'αA in stem list'

The RePortS Algorithm [Keshava and Pitler, 2006]

Three steps + added improvements

1. Building up data structure
2. Finding affixes
   Finding word stems
3. Segmenting words
   Segmentation ranking with n-gram language model

New Additional Step: Finding Word Stems

German corpus:
Studentenausweis
Studentenausschuß
Studentenausschüsse
Eingreiftruppe
eingreifst
eingreift
runterschlucken
runterschaute

stem candidate	suffix list
studentenaus	{schuß weise weis schusses schüsse schuss}
geschäftsflug	{hafen zeugen häfen zeuge hafens verkehr verkehrs}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st}
exekutier	{t en ten te ung e ter er end est et st tet}
runtersch	{lucken iebt ubsen icken aute}

New Additional Step: Finding Word Stems

German corpus:
Studentenausweis
Studentenausschuß
Studentenausschüsse
Eingreiftruppe
eingreifst
eingreift
runterschlucken
runterschaute

- Algorithm 'studentenaus' to be a stem
- With suffixes:
  'weis', 'schüsse', 'schuß'
- Enter 'studentenaus' as a stem candidate, and list all suffixes it occurred with

stem candidate	suffix list
studentenaus	{schuß weise weis schusses schüsse schuss}
geschäftsflug	{hafen zeugen häfen zeuge hafens verkehr verkehrs}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st}
exekutier	{t en ten te ung e ter er end est et st tet}
runtersch	{lucken iebt ubsen icken aute}

New Additional Step: Finding Word Stems

German corpus:
Studentenausweis
Studentenausschuß
Studentenausschüsse
Eingreiftruppe
eingreifst
eingreift
runterschlucken
runterschaute

- Suffixes are ordered into two different groups:
  1. Compounds (if suffix occurs in corpus independently)
  2. Inflectional suffix (otherwise)

stem candidate	suffix list
studentenaus	{schuß weise weis schusses schüsse schuss}
geschäftsflug	{hafen zeugen häfen zeuge hafens verkehr verkehrs}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st}
exekutier	{t en ten te ung e ter er end est et st tet}
runtersch	{lucken iebt ubsen icken aute}

New Additional Step: Finding Word Stems

German corpus:
Studentenausweis
Studentenausschuß
Studentenausschüsse
Eingreiftruppe
eingreifst
eingreift
runterschlucken
runterschaute

- How to ensure quality of inflectional suffixes?
- Idea: linguistically motivated suffixes occur with many other stem candidates as well
- Otherwise they are probably artifacts

stem candidate	suffix list
studentenaus	{schuß weise weis schusses schüsse schuss}
geschäftsflug	{hafen zeugen häfen zeuge hafens verkehr verkehrs}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st} good inflectional suffixes
exekutier	{t en ten te ung e ter er end est et st tet}
runtersch	{lucken iebt ubsen icken aute} rubbish!

Summary – Finding Morphemes

Output of morpheme acquisition step

- List of prefixes (same as with original algorithm)
- List of suffixes (empty with original algorithm due to first condition "stem must be word in corpus")
- List of stems (new)

Example segmentation of word *Abholung*

- Original algorithm: Abholung $\longrightarrow$ Ab+holung
  ('*Ab*' in prefix list)
- Modified algorithm: Abholung $\longrightarrow$ Ab+hol+ung
  ('*Ab*' in prefix list, '*ung*' in suffix list)

Step 3: Segmenting Words – Original Method

Segmentation strategy

- Peel off the affix with lowest transitional probability P_{trans} (if exists $P_{trans} < 1$)
- Do this iteratively for prefixes and suffixes

Issues

- No affix context taken into account
- Allows for morphotactically impossible segmentations (e.g. *sen+s+ation*)
- Cannot segment beyond an unknown morpheme (e.g. *Mäß+ig+ung+s+ge+löb+nis*)

The RePortS Algorithm [Keshava and Pitler, 2006]

Three steps + added improvements

1. Building up data structure
2. Finding affixes
   Finding word stems
3. Segmenting words
   Segmentation ranking with n-gram language model

New Method: Context-Sensitive Segmentation

1. **Generate all possible segmentations**
   - Locally most probable suffix not necessarily globally best solution
2. **Heuristic pruning**
   - Prefer analyses without unknown segments
     (e.g. access+ible instead of acce+s+s+ible)
3. **Ranking using language model**
   - Bi-gram model trained on simple segmentations (bootstrapping)
   - Biased towards properties of first-round segmentations (as in original algorithm)

Overview

Stem Variation Detection Method

Task

- Detect the relation between ‘Wort’ – ‘Wörter’, ‘panic’ – ‘panicked’

Observation

- Items in suffix list are often inflectional variants
- High precision word clusters
- But: language-specific for compounding languages

stem candidate	suffix list
studentenaus	{schuß weise weis schusses schüsse schuss}
geschäftsflug	{hafen zeugen häfen zeuge hafens verkehr verkehrs}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st}

Edit Distance

- Calculate edit-distance between all items in each suffix list
  e.g. `edit-dist(hafen, häfen)` = 2
- Resulting pattern: a $\longrightarrow$ ä

Stem Variation Detection Method

Task

- Detect the relation between ‘Wort’ – ‘Wörter’, ‘panic’ – ‘panicked’

Observation

- Items in suffix list are often inflectional variants
- High precision word clusters
- But: language-specific for compounding languages

stem candidate	suffix list
studentenaus	{schuß weise weis schusses schüsse schuss}
geschäftsflug	{hafen zeugen häfen zeuge hafens verkehr verkehrs}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st}

Edit Distance

- Calculate edit-distance between all items in each suffix list
  e.g. `edit-dist(hafen, häfen)` = 2
- Resulting pattern: a $\longrightarrow$ ä

Stem Variation Detection Method

Task

- Detect the relation between ‘Wort’ – ‘Wörter’, ‘panic’ – ‘panicked’

Observation

- Items in suffix list are often inflectional variants
- High precision word clusters
- But: language-specific for compounding languages

stem candidate	suffix list
studentenaus	{schuß weise weis schusses schüsse schuss}
geschäftsflug	{hafen zeugen häfen zeuge hafens verkehr verkehrs}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st}

Edit Distance

- Calculate edit-distance between all items in each suffix list
  e.g. `edit-dist(hafen, häfen)` = 2
- Resulting pattern: a $\longrightarrow$ ä

Stem Variation Detection Method

Task

- Detect the relation between ‘Wort’ – ‘Wörter’, ‘panic’ – ‘panicked’

Observation

- Items in suffix list are often inflectional variants
- High precision word clusters
- But: language-specific for compounding languages

stem candidate	suffix list
studentenaus	{schuß weise weis schusses schüsse schuss}
geschäftsflug	{hafen zeugen häfen zeuge hafens verkehr verkehrs}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st}

Edit Distance

- Calculate edit-distance between all items in each suffix list
  e.g. `edit-dist(hafen, häfen)` = 2
- Resulting pattern: a $\longrightarrow$ ä

Stem Variation Patterns

Accept highly frequent patterns

freq.	pattern	examples
1682	a $\longrightarrow$ ä..e	sack-säcke, brach-bräche, stark-stärke
344	a $\longrightarrow$ ä	sahen-sähen, garten-gärten
321	u $\longrightarrow$ ü..e	flug-flüge, bund-bünde
289	ä $\longrightarrow$ a..s	verträge-vertrages, pässe-passes
189	o $\longrightarrow$ ö..e	chor-chöre, strom-ströme, ?röhre-rohr
175	t $\longrightarrow$ en	setzt-setzen, bringt-bringen
168	a $\longrightarrow$ u	laden-luden, *damm-dumm
160	ß$\longrightarrow$ ss	läßt-lässt, mißbrauch-missbrauch
[. . .]		
136	a $\longrightarrow$ en	firma-firmen, thema-themen
[. . .]		
2	ß$\longrightarrow$ g	*fließen-fliegen, *laßt-lagt
2	um $\longrightarrow$ o	*studiums-studios

Stem Variation Patterns

Accept highly frequent patterns

freq.	pattern	examples
1682	a $\longrightarrow$ ä..e	sack-säcke, brach-bräche, stark-stärke
344	a $\longrightarrow$ ä	sahen-sähen, garten-gärten
321	u $\longrightarrow$ ü..e	flug-flüge, bund-bünde
289	ä $\longrightarrow$ a..s	verträge-vertrages, pässe-passes
189	o $\longrightarrow$ ö..e	chor-chöre, strom-ströme, ?röhre-rohr
175	t $\longrightarrow$ en	setzt-setzen, bringt-bringen
168	a $\longrightarrow$ u	laden-luden, *damm-dumm
160	ß $\longrightarrow$ ss	läßt-lässt, mißbrauch-missbrauch
[. . .]		
136	a $\longrightarrow$ en	firma-firmen, thema-themen
[. . .]		
2	ß $\longrightarrow$ g	*fließen-fliegen, *laßt-lagt
2	um $\longrightarrow$ o	*studiums-studios

Integration of Stem Variation Component

Applications for stem variation information

- Word segmentation
  e.g. Spr+ung, Spr+ünge, spr+ingen, spr+ang, spr+änge
  (generate equivalence classes for transitional probabilities)
- Lemmatization
  (identify semantically related words)

Implementation

- Use patterns to generate letter equivalence sets
- e.g. pattern ‘a$\rightarrow$ä’ generates equivalence class $\{$a,ä$\}$

Results

- 2% more recall without loss in precision (German)

Overview

Data sets

We evaluated the algorithm on four different languages:

Language	Data set size	Evaluation on:
German	240m tokens	250k words from CELEX
English	24m tokens	MorphoChallenge test set
Turkish	16m tokens	MorphoChallenge test set
Finnish	32m tokens	MorphoChallenge test set

Evaluation of Effect of Versions

Language	alg. version	F-Measure	Precision	Recall
German	original	59.2%	71.1%	50.7%
	+stems	68.4%	68.1%	68.6%
	+n-gram seg.	**68.9%**	73.7%	64.6%
English	original*	**76.8%**	76.2%	77.4%
	+stems	67.6%	62.9%	73.1%
	+n-gram seg.	75.1%	74.4%	75.9%
Turkish	original	54.2%	72.9%	43.1%
	+stems	61.8%	65.9%	58.2%
	+n-gram seg.	**64.2%**	65.2%	63.3%
Finnish	original	47.1%	84.5%	32.6%
	+stems	56.6%	74.1%	45.8%
	+n-gram seg.	**58.9%**	76.1%	48.1%
	max-split*	**61.3%**	66.3%	56.9%

Evaluation of Effect of Versions

Language	alg. version	F-Measure	Precision	Recall
German	original	59.2%	71.1%	50.7%
	+stems	68.4%	68.1%	68.6%
	+n-gram seg.	**68.9%**	73.7%	64.6%
English	original*	**76.8%**	76.2%	77.4%
	+stems	67.6%	62.9%	73.1%
	+n-gram seg.	75.1%	74.4%	75.9%
Turkish	original	54.2%	72.9%	43.1%
	+stems	61.8%	65.9%	58.2%
	+n-gram seg.	**64.2%**	65.2%	63.3%
Finnish	original	47.1%	84.5%	32.6%
	+stems	56.6%	74.1%	45.8%
	+n-gram seg.	**58.9%**	76.1%	48.1%
	max-split*	**61.3%**	66.3%	56.9%

Evaluation of Effect of Versions

Language	alg. version	F-Measure	Precision	Recall
German	original	59.2%	71.1%	50.7%
	+stems	68.4%	68.1%	68.6%
	+n-gram seg.	**68.9%**	73.7%	64.6%
English	original*	**76.8%**	76.2%	77.4%
	+stems	67.6%	62.9%	73.1%
	+n-gram seg.	75.1%	74.4%	75.9%
Turkish	original	54.2%	72.9%	43.1%
	+stems	61.8%	65.9%	58.2%
	+n-gram seg.	**64.2%**	65.2%	63.3%
Finnish	original	47.1%	84.5%	32.6%
	+stems	56.6%	74.1%	45.8%
	+n-gram seg.	**58.9%**	76.1%	48.1%
	max-split*	**61.3%**	66.3%	56.9%

Evaluation of Effect of Versions

Language	alg. version	F-Measure	Precision	Recall
German	original	59.2%	71.1%	50.7%
	+stems	68.4%	68.1%	68.6%
	+n-gram seg.	**68.9%**	73.7%	64.6%
English	original*	**76.8%**	76.2%	77.4%
	+stems	67.6%	62.9%	73.1%
	+n-gram seg.	75.1%	74.4%	75.9%
Turkish	original	54.2%	72.9%	43.1%
	+stems	61.8%	65.9%	58.2%
	+n-gram seg.	**64.2%**	65.2%	63.3%
Finnish	original	47.1%	84.5%	32.6%
	+stems	56.6%	74.1%	45.8%
	+n-gram seg.	**58.9%**	76.1%	48.1%
	max-split*	**61.3%**	66.3%	56.9%

Evaluation of Effect of Versions

Language	alg. version	F-Measure	Precision	Recall
German	original	59.2%	71.1%	50.7%
	+stems	68.4%	68.1%	68.6%
	+n-gram seg.	**68.9%**	73.7%	64.6%
English	original*	**76.8%**	76.2%	77.4%
	+stems	67.6%	62.9%	73.1%
	+n-gram seg.	75.1%	74.4%	75.9%
Turkish	original	54.2%	72.9%	43.1%
	+stems	61.8%	65.9%	58.2%
	+n-gram seg.	**64.2%**	65.2%	63.3%
Finnish	original	47.1%	84.5%	32.6%
	+stems	56.6%	74.1%	45.8%
	+n-gram seg.	**58.9%**	76.1%	48.1%
	max-split*	**61.3%**	66.3%	56.9%

Task-based Evaluations: Grapheme-to-Phoneme Conversion

Pronunciation of words is sensitive to morphological boundaries

- English example: *loophole*
  /luːfəʊl/ vs. /luːphəʊl/
- *Sternanisöl*
  /ʃtɛrnʔaniːsʔøːl/ vs. /ʃtɛrnaːnizœːl/
- *Röschen*
  /rœʃən/ vs. /rœːsçən/
- *vertikal* vs. *vertickern*
  /v/ vs. /f/
- *Weihungen* vs. *Gen*
  /ə/ vs. /eː/

Task-based Evaluation – Results

morphology	F-Measure (CELEX)	PER (dec.tree)
CELEX	100%	2.64%
ETI	79.5%	2.78%
SMOR	83.0%	3.00%
RePortS-ngram	68.8%	3.45%
no morphology	–	3.63%
orig. RePortS	59.2%	3.83%
Bernhard, 2006	63.5%	3.88%
RePortS-stem	68.4%	3.98%
Morfessor 1.0	52.6%	4.10%
Bordag, 2006	64.1%	4.38%

- Trained decision tree for g2p on morphological segmentations
- CELEX manual annotation used as gold standard
- Rule-based systems worked best
- **RePortS n-gram** only unsupervised system that improves g2p conversion with respect to no-morphology–baseline

Conclusions

Conclusions

- Improved over Reports
- Best Performance on German, English
- Good performance across the board
- Simple and efficient method
  - Training on 240 m tokens: 5 min
  - Running 250 k test words: 3 min (stems), 8 min (n-gram)
- Stem variation method improves recall

Future Work

- More sophisticated language model for segmentation
- Application of method to morphological tasks other than segmentation

Conclusions

Conclusions

- Improved over Reports
- Best Performance on German, English
- Good performance across the board
- Simple and efficient method
  - Training on 240 m tokens: 5 min
  - Running 250 k test words: 3 min (stems), 8 min (n-gram)
- Stem variation method improves recall

Future Work

- More sophisticated language model for segmentation
- Application of method to morphological tasks other than segmentation

Acknowledgments

Thank you:

- **Emily Pitler**, **Samarth Keshava**: for sharing the code of their algorithm
- **Stefan Bordag**, **Delphine Bernhard**: for running their algorithms on my German data
- **Matti Varjokallio**: for evaluating my segmentations on the MorphoChallenge data (English, Turkish, Finnish)
- **Christoph Zwirello**, **Gregor Möhler**: for training the decision tree on the new morphological segmentation
- **Frank Keller** and **reviewers**: for valuable comments

... and thanks to you for your attention!

Bernhard, D. (2006).
Unsupervised morphological segmentation based on segment predictability and word segments alignment.
In *Proceedings of 2nd Pascal Challenges Workshop*, pages 19–24, Venice, Italy.

Bordag, S. (2006).
Two-step approach to unsupervised morpheme segmentation.
In *Proceedings of 2nd Pascal Challenges Workshop*, pages 25–29, Venice, Italy.

Creutz, M. and Lagus, K. (2006).
Unsupervised models for morpheme segmentation and morphology learning.
In *ACM Transaction on Speech and Language Processing*.

Déjean, H. (1998).
Morphemes as necessary concepts for structures: Discovery from untagged corpora.
In *Workshop on paradigms and Grounding in Natural Language Learning*, pages 295–299, Adelaide, Australia.

Demberg, V. (2007).
Phonological constraints and morphological preprocessing for grapheme-to-phoneme conversion.
In *Proc. of ACL-2007*.

Goldsmith, J. (2001).
Unsupervised learning of the morphology of a natural language.
computational Linguistics, 27(2):153–198.

Goldwater, S. and McClosky, D. (2005).
Improving statistical mt through morphological analysis.
In *Proc. of EMNLP*.

Hafer, M. A. and Weiss, S. F. (1974).
Word segmentation by letter successor varieties.
Information Storage and Retrieval 10, pages 371–385.

Harris, Z. (1955).
From phoneme to morpheme.
Language 31, pages 190–222.

Jacquemin, C. (1997).
Guessing morphology from terms and corpora.
In *Research and Development in Information Retrieval*, pages 156–165.

Keshava, S. and Pitler, E. (2006).
A simpler, intuitive approach to morpheme induction.
In *Proceedings of 2nd Pascal Challenges Workshop*, pages 31–35, Venice, Italy.

Kurimo, M., Creutz, M., Varjokallio, M., Arisoy, E., and Saraclar, M. (2006).
Unsupervsied segmentation of words into morphemes – Challenge 2005: An introduction and evaluation report.
In *Proc. of 2nd Pascal Challenges Workshop*, Italy.

Möbius, B. (2001).
German and Multilingual Speech Synthesis.
phonetic AIMS, Arbeitspapiere des Instituts für Maschinelle Spachverarbeitung.

Monson, C., Lavie, A., Carbonell, J., and Levin, L. (2004).
Unsupervised induction of natural language morphology inflection classes.
In *Proceedings of the Seventh Meeting of ACL-SIGPHON*, pages 52–61, Barcelona, Spain.

Monz, C. and de Rijke, M. (2002).

Shallow morphological analysis in monolingual information retrieval for Dutch, German, and Italian.
In *Proceedings CLEF 2001, LNCS 2406*.

Neuvel, S. and Fulop, S. (2002).
Unsupervised learning of morphology without morphemes.
In *Proc. of the Wshp on Morphological and Phonological Learning, ACL Pub.*

Puurula, A. and Kurimo, M. (2007).
Vocabulary decomposition for estonian open vocabulary speech recognition.
In *Proc. of ACL-2007*.

Schone, P. and Jurafsky, D. (2000).
Knowledge-free induction of morphology using latent semantic analysis.
In *Proceedings of the Computational Natural Language Learning Conference*, Lisbon.

Sproat, R. (1996).
Multilingual text analysis for text-to-speech synthesis.
In *Proc. ICSLP '96*, Philadelphia, PA.

Taylor, P. (2005).
Hidden Markov models for grapheme to phoneme conversion.
In *INTERSPEECH*, pages 1973–1976, Lisbon, Portugal.

Yarowsky, D. and Wicentowski, R. (2000).
Minimally supervised morphological analysis by multimodal alignment.
In *Proceedings of ACL 2000*, Hong Kong.

Morphology

Concatenative Processes

- Prefixation: *un-do, re-open*
- Suffixation: *work, work-ing, work-ed, work-s*
- Compounding: *loop-hole*
- Circumfixation: *ge-mach-t ‘done’, ge-sproch-en ‘said’* (German)
- Infixation: *sulat ‘write’, s-um-ulat ‘wrote’, s-in-ulat ‘was written’* (Tagalog)
- Reduplication: *mejr ‘to sleep’, mej-mejr ‘sleeping’, mej-mej-mejr ‘still sleeping’* (Pingelapese)

Non-concatenative Processes

- Ablauting: *sing, sang, sung*
- Umlauting: *Garten, Gärten* ‘garden’
- Vowel harmony: *ev – evler ‘house’, kitap – kitaplar ‘book’* (Turkish)
- Deletion / insertion: *care, caring; panic, panicked*

Finding Word Stems (2)

1. **Create a list of candidate stems**

studentenaus	{schuß weise weis schusses schüsse schuss}
eingreif	{truppe werte trupps mandat trupp kräfte verband ...}
	+{en t e er est et st}
runtersch	{lucken iebt ubsen icken aute}

2. **Assess the stem candidates**
   - Accept all candidates with corpus words only
   - Rank by average frequency of non-corpus words (generated affixes)
3. **Define threshold for ranked list**