Morphology in Machine Translation

Joachim Daiber

Institute for Logic, Language and Computation
University of Amsterdam

Today's lecture

- Introduction

—

- Part 1: Morphology induction
- Part 2: Morphology and syntax

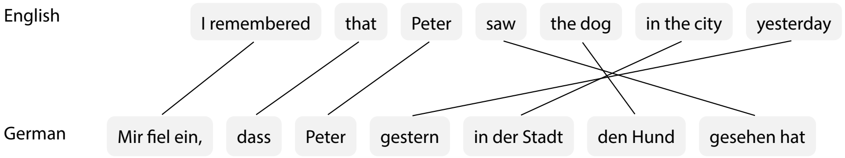
—

- Soft enforcement of agreement constraints in syntactic MT
  Georgi, Tom and Maartje

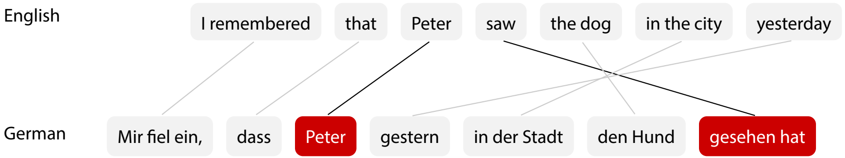
Translation into morphologically rich languages

English	I remembered	that	Peter	saw	the dog	in the city	yesterday
German	Mir fiel ein,	dass	Peter	gestern	in der Stadt	den Hund	gesehen hat

Translation into morphologically rich languages



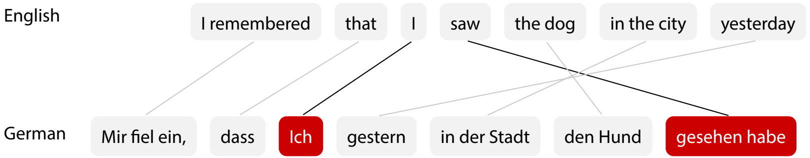
Translation into morphologically rich languages



Challenges:

- Morphological agreement over long distances

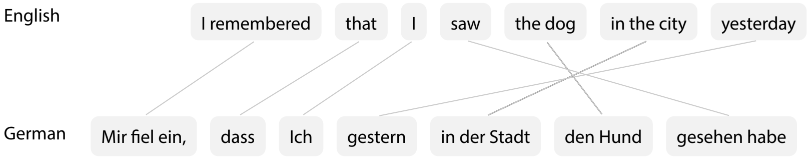
Translation into morphologically rich languages



Challenges:

- Morphological agreement over long distances

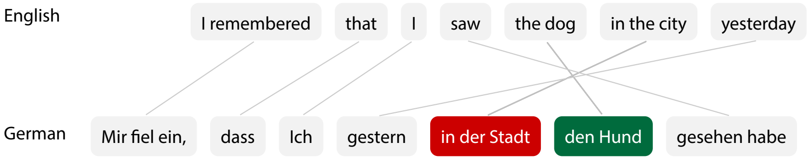
Translation into morphologically rich languages



Challenges:

- Morphological agreement over long distances

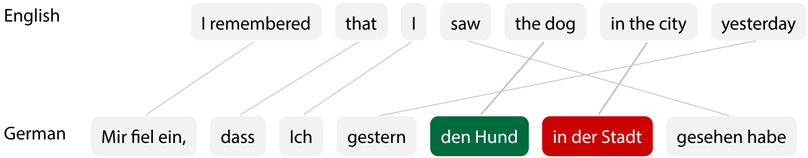
Translation into morphologically rich languages



Challenges:

- Morphological agreement over long distances
- Relatively freer word order

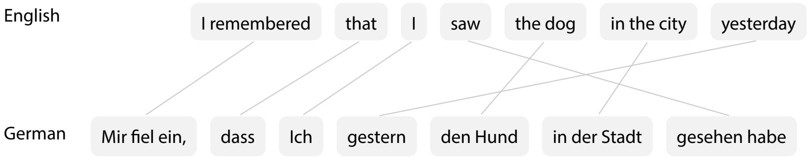
Translation into morphologically rich languages



Challenges:

- Morphological agreement over long distances
- Relatively freer word order

Translation into morphologically rich languages



Challenges:

- Morphological agreement over long distances
- Relatively freer word order
- Data sparsity

Translation into morphologically rich languages

- Established methods often do not work well
- One example: Source-side reordering

Morphology Induction

Morphology induction from word embeddings

Paper: Unsupervised morphology induction using word embeddings.
Radu Soricut and Franz Och, NAACL 2015.

Question: Can we induce representations of morphology from representations of words?

Word embeddings

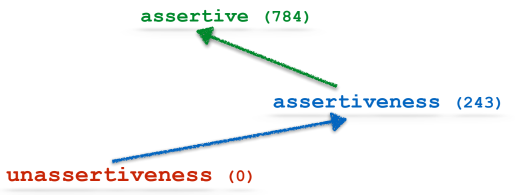
- vocabulary V, embedding function e: $V \to \mathbf{R}^n$
- vector space encodes semantic similarity
  - e(car) ≃ e(automobile), e(car) ≠ e(seahorse)
- vector space encodes compositionality
  - semantic: e(king) - e(man) + e(woman) ≃ e(queen)
  - syntactic: e(cars) - e(car) + e(fireman) ≃ e(firemen)
- vector space encodes syntactic/semantic transformations
  - anti+ ≃ e(anticoruption) - e(corruption)

Slides by Radu Soricut

Morphology induction from word embeddings

Q: What do we want?

A: We want *high-quality* embeddings for all words (even ones outside *V*)

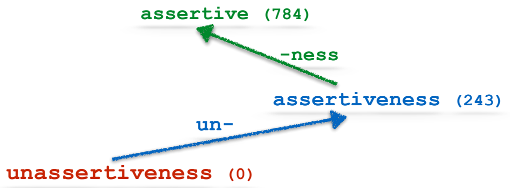


Slides by Radu Soricut

Morphology induction from word embeddings

Q: What do we want?

A: We want *morphology-based transformations* that can accurately analyze words (even ones unseen at training time)

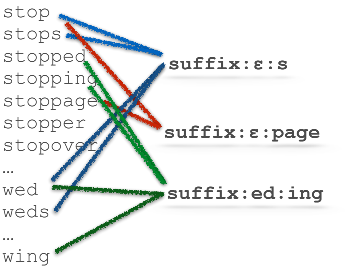


Slides by Radu Soricut

Algorithm: Steps

Steps:

1. From *V*, extract candidates for morphological rules (prefix & suffix only)

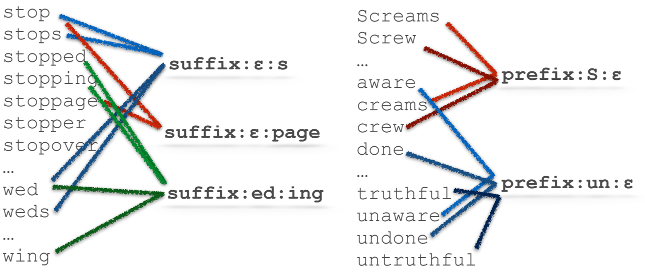


Slides by Radu Soricut

Algorithm: Steps

Steps:

1. From *V*, extract candidates for morphological rules (prefix & suffix only)



Slides by Radu Soricut

Algorithm: Steps

Steps:

2. Query against embedding space: *morphology does not shift meaning*

suffix:ed:ing

adored adorned affected …
blamed blitzed blogged …
stayed stepped stopped …
weaned wed wedged whirled

rank(blamed → blaming) = 1
rank(stopped → stopping)= 2
rank(wed → wing) = 28609

prefix:ε:S

aura aux ave …
canned cans car care …
crape cream creams …
miles mitten mothers …

rank(care → Scare) = 57778
rank(cream → Scream)= 9434
rank(miles → Smiles)= 18800

Slides by Radu Soricut

Algorithm: Steps

Steps:

2. Query against embedding space: *morphology does not shift meaning*

prefix:un:ε

unabated unable unabridged…
unaware unbalance unbeaten…
undoing undone undoubted…
untrusted untrustworthy…

rank(unaware → aware) = 1
rank(undone → done) = 129

Slides by Radu Soricut

Algorithm: Steps

Steps:

2. Query against embedding space: ~~*morphology does not shift meaning*~~

morphology shifts meaning consistently

prefix:un:ε

unabated unable unabridged…
unaware unbalance unbeaten…
undoing undone undoubted…
untrusted untrustworthy…

rank(unaware → aware) = 0
rank(undone → done) = 129

↑un-

clear - unclear
delivered - undelivered
truthful - untruthful

rank(undone + ↑un- → done) = 4

Slides by Radu Soricut

Algorithm: Steps

Steps:

3. Extract candidate rules using embedding-based stats

	Candidate Rule	Direction	#Correct	#Total	Acc10
Bad	**suffix:h:a**	↑Teh	1	449	0.4%
	suffix:o:es	↑Tono	7	688	1.0%
	prefix:D:W	↑Daring	9	675	1.3%
	...				
Good	**prefix:un:ε**	↑undelivered	166	994	23.3%
	suffix:ed:ing	↑procured	2138	4714	56.2%
	...				
	suffix:ating:ate	↑formulating	255	395	74.7%
	suffix:sed:zed	↑victimised	153	186	90.9%

Slides by Radu Soricut

Algorithm: Steps

Steps:

4. Use rules to extract lexicalized, weighted morphological transformations

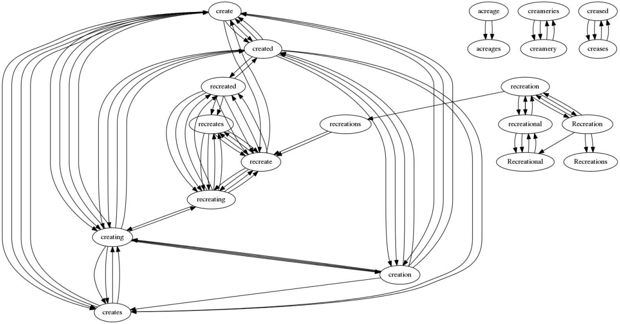
Start	Rule + Direction = Transformation	End	Cosine	Rank
…				
recreations	suffix:ions:e + ↑investigations	recreate	0.69	1
recreations	suffix:tions:te + ↑investigations	recreate	0.70	1
recreations	suffix:ions:ed + ↑delineations	recreated	0.51	29
recreations	suffix:ions:ing + ↑reconstruction	recreating	0.72	1
…				
unaware	prefix:un:ε + ↑uncivilized	aware	0.77	1
unaware	prefix:un:ε + ↑undelivered	aware	0.63	7

Slides by Radu Soricut

Algorithm: Output

Output (I): labeled, weighted, cyclic, directed multigraph $G^{V}{}_{Morph}$

- words are nodes, morphological transformations are (weighted) edges

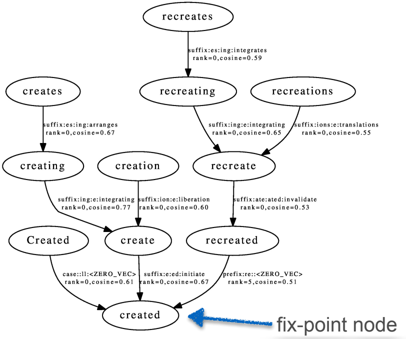


Slides by Radu Soricut

Algorithm: Output

Output (II): labeled, weighted, acyclic, directed graph D^{V}_{Morph}

- words are nodes, morphological mappings are weighted edges

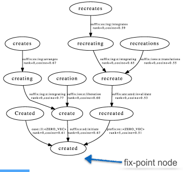


Slides by Radu Soricut

Application

Analyze words outside *V*

1. Train time: extract and count all paths ending in a "fix-point" from the directed acyclic graph D^V_{Morph}
   - each path is called a "rule sequence"



rule sequence	count
`suffix:s:ε`	3119
`suffix:ed:ε`	687
`suffix:ing:ed`	412
`prefix:un:ε`	207
`suffix:ness:ε`	162
`suffix:ness:ly`	25
`suffix:y:ier,suffix:er:ness`	10
`prefix:un:ε,suffix:ed:ing`	5

Application

Analyze words outside V

2. Run time: apply each rule sequence in descending order of counts
   - if rule fires, check that result has count > 0 and in-degree > 0
   - stop at first winner

rule sequence	count
suffix:s:ε	3119
suffix:ed:ε	687
suffix:ing:ed	412
prefix:un:ε	207
suffix:ness:ε	162
suffix:ness:ly	25
suffix:y:ier,suffix:er:ness	10
prefix:un:ε,suffix:ed:ing	5

unassertiveness(0) → suffix:s:ε → unassertivenes(0)

unassertiveness(0) → prefix:un:ε → assertiveness(243)

unassertiveness = assertiveness + ↑un+

Evaluation

Training Setup

	Language	Train Set	|Tokens|	|V|	$\|G^V{}_{Morph}\|$	$\|D^V{}_{Morph}\|$
Small	EN	Wiki-EN	1.1b	1.2m	780k	75,823
	DE	WMT-DE	1.2b	2.9m	3.7m	169,017
Large	EN	News-EN	120b	1.0m	2.9m	98,268
	DE	News-DE	20b	1.8m	6.7m	351,980

Slides by Radu Soricut

Evaluation

Evaluation on similarity datasets (RG-DE, RW-EN)

size: 2034 pairs

impossibilities	unattainableness	8.8
deregulating	liberation	8.0
baseness	unworthiness	4.0
transmigrating	born	1.1

Language	Train Set	|Tokens|	|V|	$\|G^V_{Morph}\|$	$\|D^V_{Morph}\|$
EN	Wiki-EN	1.1b	1.2m	780k	75,823
DE	WMT-DE	1.2b	2.9m	3.7m	169,017
EN	News-EN	120b	1.0m	2.9m	98,268
DE	News-DE	20b	1.8m	6.7m	351,980

	RW-EN Testset			
	|Unembedded|		Spearman ρ	
System	Wiki-EN	News-EN	Wiki-EN	News-EN
SkipGram	78	177	35.8	44.7
SkipGram+Morph	1	0	41.8	52.0

+9 +7

Slides by Radu Soricut

Evaluation

Evaluation on similarity datasets (RG-DE, RW-EN)

size: 65 pairs

Edelstein	Juwel	3.8
Autogramm	Unterschrift	3.5
Irrenhaus	Friedhof	0.3
Kraftfahrzeug	Magier	0.0

Language	Train Set	|Tokens|	|V|	$\|G^V_{Morph}\|$	$\|D^V_{Morph}\|$
EN	Wiki-EN	1.1b	1.2m	780k	75,823
DE	WMT-DE	1.2b	2.9m	3.7m	169,017
EN	News-EN	120b	1.0m	2.9m	98,268
DE	News-DE	20b	1.8m	6.7m	351,980

	RW-EN Testset			
	|Unembedded|		Spearman ρ	
System	Wiki-EN	News-EN	Wiki-EN	News-EN
SkipGram	80	177	35.8	44.7
SkipGram+Morph	1	0	41.8	52.0

+9 +7

	RG-DE Testset			
	|Unembedded|		Spearman ρ	
System	WMT-DE	News-DE	WMT-DE	News-DE
SkipGram	0	20	62.4	62.1
SkipGram+Morph	0	0	64.1	69.1

+0 +7

Slides by Radu Soricut

Conclusions

1. Method for inducing morphological transformations between words
   - from scratch, unsupervised, language agnostic
2. Provides morphology-based structure over embedding spaces
3. Provides high-quality embeddings for out-of-vocabulary and low-count morphological variants

Slides by Radu Soricut

Compounds

Compound induction from word embeddings

Paper: Splitting Compounds by Semantic Analogy

Joachim Daiber, Lautaro Quiroz, Roger Wechsler and Stella Frank, DMTW 2015.

Question: Can we learn to split compounds using those sub-word representations?

Compounds in MT

Compound words...

- ... make life hard for standard NLP applications, incl. MT
- ... are often modeled with shallow information (e.g. Moses frequency-based splitter)

Question: Can we use distributional semantics to do deeper processing of compounds in a simple way?

Splitting compounds for SMT

- Koehn and Knight (2003) showed PBMT systems can better deal with compounds if they are split into their meaningful parts
- Difficulty: many possible splits, we need to choose the correct ones

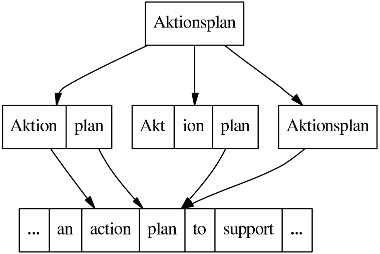


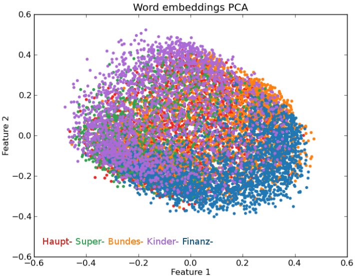
Figure: Compound splitting example from Koehn and Knight (2003).

Compounds and the semantic vector space

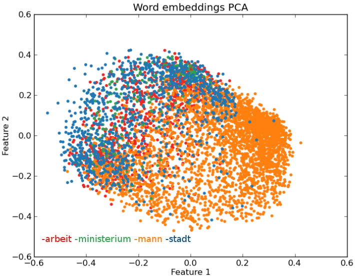
Semantic vector space

- Word embeddings saw surge of successful applications recently
- Basic idea: ”You shall know a word by the company it keeps”
  - Words are mapped to vectors of real numbers in low dimensional space
  - These vectors are estimated on large amounts of text data using a neural network

Compounds and the semantic vector space



(a) Compounds with same modifier.



(b) Compounds with the same head.

The analogy test

- We model compounds based on their modifiers
- Potential compound splits are judged by how similar they are to a set of prototypical compounds for each modifier

Analogy test: *Mauszeiger* is to *Zeiger* what *Mausklick* is to *Klick*?
(mouse pointer) (pointer) (mouse click) (click)

Extracting potential compound splits

For all words in the vocabulary:

- Extract all possible string prefixes ≥ 4:
  Bundespräsident $\rightarrow$ *Bund, Bunde, Bundes, ...*
- Judge each Modifier+Compound pair by how well it explains others

Judging potential compound splits

All potential compounds with prefix *Maus*

Maus|kostüm
Maus|zeiger
Maus|stämme
Maus|klick
Maus|hirn
Maus|tasten
Maus|ersatz
Maus|mutanten
Maus|knopf
Maus|steuerung
Maus|bewegung
Maus|gene
Maus|klicks
Maus|hirns
Maus|zeiger
Maus|hirnen
Maus|bedienung

...
(up to 500)

Judging potential compound splits

All potential compounds $\times$ All potential compounds

Maus|kostüm
Maus|zeiger
Maus|stämme
Maus|klick
Maus|hirn
Maus|tasten
Maus|ersatz
Maus|mutanten
Maus|knopf
Maus|steuerung
Maus|bewegung
Maus|gene
Maus|klicks
Maus|hirns
Maus|zeiger
Maus|hirnen
Maus|bedienung
...
(up to 500)

Maus|kostüm
Maus|zeiger
Maus|stämme
Maus|klick
Maus|hirn
Maus|tasten
Maus|ersatz
Maus|mutanten
Maus|knopf
Maus|steuerung
Maus|bewegung
Maus|gene
Maus|klicks
Maus|hirns
Maus|zeiger
Maus|hirnen
Maus|bedienung
...
(up to 500)

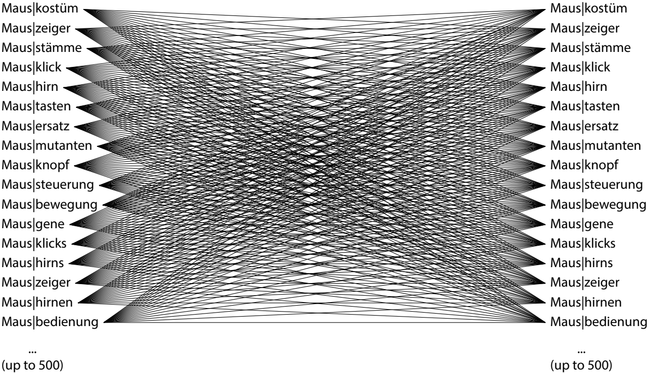
Judging potential compound splits

All potential compounds $\times$ All potential compounds

Maus|kostüm	Maus|kostüm
Maus|zeiger	Maus|zeiger
Maus|stämme	Maus|stämme
Maus|klick	Maus|klick
Maus|hirn	Maus|hirn
Maus|tasten	Maus|tasten
Maus|ersatz	Maus|ersatz
Maus|mutanten	Maus|mutanten
Maus|knopf	Maus|knopf
Maus|steuerung	Maus|steuerung
Maus|bewegung	Maus|bewegung
Maus|gene	Maus|gene
Maus|klicks	Maus|klicks
Maus|hirns	Maus|hirns
Maus|zeiger	Maus|zeiger
Maus|hirnen	Maus|hirnen
Maus|bedienung	Maus|bedienung
...	...
(up to 500)	(up to 500)

Judging potential compound splits

All potential compounds $\times$ All potential compounds



Judging potential compound splits

Maus|zeiger	Maus|zeiger
Maus|stämme	Maus|stämme
Maus|klick	Maus|klick
Maus|hirn	Maus|hirn

Perform analogy test: *Mauszeiger* is to *Zeiger* what *Mausklick* is to *Klick*?
(mouse pointer) (pointer) (mouse click) (click)

Computational considerations

- **Analogy test is expensive!**
- True and predicted vectors:
  - $v_{\text{Mausklick}}$
  - $\hat{v}_{\text{Mausklick}} = \text{Mauszeiger} - \text{Zeiger} + \text{Klick}$
- Two evaluation functions: $\textsc{Rank}$ and $\textsc{Cosine}$

Computational considerations

- Exact but slow implementation:

$$\textsc{Rank}(v_{cmpd}, \hat{v}_{cmpd}) = \textsc{Rank of } v_{cmpd} \textsc{ in } \operatorname*{arg\,sort}_{w \in V} \left[\textsc{Cosine}\left(v_w, \hat{v}_{cmpd}\right) \right]$$

- Approximate but fast implementation:
  - Approximate k-nearest neighbor search
  - We use the Spotify Annoy library (C++) to perform the search

- *Maus|zeiger* explains *Maus|klick* **IFF**

$$\textsc{Rank}(v_{\mathrm{cmpd}}, \hat{v}_{\mathrm{cmpd}}) < 100 \quad \textbf{AND} \quad \textsc{Cosine}(v_{\mathrm{cmpd}}, \hat{v}_{\mathrm{cmpd}}) > 0.5$$

Prototypes

Compounds that are good examples of a compound modifier.

- These are best at explaining other similar modifier+compound pairs
- We call this set the modifier's ***prototypes***

Extracting prototypes

Maus|kostüm
Maus|zeiger
Maus|stämme
Maus|klick
Maus|hirn
Maus|tasten
Maus|ersatz
Maus|mutanten
Maus|knopf
Maus|steuerung
Maus|bewegung
Maus|gene
Maus|klicks
Maus|hirns
Maus|zeiger
Maus|hirnen
Maus|bedienung
...
(up to 500)

Maus|kostüm
Maus|zeiger
Maus|stämme
Maus|klick
Maus|hirn
Maus|tasten
Maus|ersatz
Maus|mutanten
Maus|knopf
Maus|steuerung
Maus|bewegung
Maus|gene
Maus|klicks
Maus|hirns
Maus|zeiger
Maus|hirnen
Maus|bedienung
...
(up to 500)

Extracting prototypes

Maus|kostüm	Maus|kostüm
Maus|zeiger	Maus|zeiger
Maus|stämme	Maus|stämme
Maus|klick	Maus|klick
Maus|hirn	Maus|hirn
Maus|tasten	Maus|tasten
Maus|ersatz	Maus|ersatz
Maus|mutanten	Maus|mutanten
Maus|knopf	Maus|knopf
Maus|steuerung	Maus|steuerung
Maus|bewegung	Maus|bewegung
Maus|gene	Maus|gene
Maus|klicks	Maus|klicks
Maus|hirns	Maus|hirns
Maus|zeiger	Maus|zeiger
Maus|hirnen	Maus|hirnen
Maus|bedienung	Maus|bedienung
...	...
(up to 500)	(up to 500)

Extracting prototypes

Maus|kostüm	Maus|kostüm
Maus|zeiger	Maus|zeiger
Maus|stämme	Maus|stämme
Maus|klick	Maus|klick
Maus|hirn	Maus|hirn
Maus|tasten	Maus|tasten
Maus|ersatz	Maus|ersatz
Maus|mutanten	Maus|mutanten
Maus|knopf	Maus|knopf
Maus|steuerung	Maus|steuerung
Maus|bewegung	Maus|bewegung
Maus|gene	Maus|gene
Maus|klicks	Maus|klicks
Maus|hirns	Maus|hirns
Maus|zeiger	Maus|zeiger
Maus|hirnen	Maus|hirnen
Maus|bedienung	Maus|bedienung
...	...
(up to 500)	(up to 500)

Extracting prototypes

Maus|kostüm	Maus|kostüm
Maus|zeiger	Maus|zeiger
Maus|stämme	Maus|stämme
Maus|klick	Maus|klick
Maus|hirn	Maus|hirn
Maus|tasten	Maus|tasten
Maus|ersatz	Maus|ersatz
Maus|mutanten	Maus|mutanten
Maus|knopf	Maus|knopf
Maus|steuerung	Maus|steuerung
Maus|bewegung	Maus|bewegung
Maus|gene	Maus|gene
Maus|klicks	Maus|klicks
Maus|hirns	Maus|hirns
Maus|zeiger	Maus|zeiger
Maus|hirnen	Maus|hirnen
Maus|bedienung	Maus|bedienung
...	...
(up to 500)	(up to 500)

Extracting prototypes

Maus|kostüm	Maus|kostüm
Maus|zeiger	Maus|zeiger
Maus|stämme	Maus|stämme
Maus|klick	Maus|klick
Maus|hirn	Maus|hirn
Maus|tasten	Maus|tasten
Maus|ersatz	Maus|ersatz
Maus|mutanten	Maus|mutanten
Maus|knopf	Maus|knopf
Maus|steuerung	Maus|steuerung
Maus|bewegung	Maus|bewegung
Maus|gene	Maus|gene
Maus|klicks	Maus|klicks
Maus|hirns	Maus|hirns
Maus|zeiger	Maus|zeiger
Maus|hirnen	Maus|hirnen
Maus|bedienung	Maus|bedienung
...	...
(up to 500)	(up to 500)

Extracted prototypes for *Maus-*

Prototype	Evidence words
$v_{\text{-Zeiger}}$	-Bewegung -Klicks -Klick -Tasten -Zeiger
$v_{\text{-Stämme}}$	-Mutanten -Gene -Hirnen -Stämme
$v_{\text{-Kostüm}}$	-Knopf -Hirn -Hirns -Kostüm
$v_{\text{-Steuerung}}$	-Ersatz -Bedienung -Steuerung

Compound splitting: *Mausmutation*

Mausmutation

- We start from the left...

Compound splitting: *Mausmutation*

Mausmutation

$\longrightarrow$

- Do I know the modifier *Mau*? No!

Compound splitting: *Mausmutation*

Mausmutation

$\longrightarrow$

- Do I know the modifier *Maus*? Yes!

Compound splitting: *Mausmutation*

Mausmutation

$\longrightarrow$

- Do I know the modifier *Maus*? Yes!

  Prototypes:

  - -Zeiger
  - -Stämme
  - -Kostüm
  - -Steuerung

Compound splitting: *Mausmutation*

<u>Maus</u>mutation

$\longrightarrow$

- Do I know the modifier *Maus*? Yes!

  Prototypes:

  - ~~-Zeiger~~
  - -Stämme ✓ $\rightarrow$ ***Mausmutation*** is to ***Mutation*** what ***Mausstämme*** is to ***Stämme***.
  - ~~-Kostüm~~
  - ~~-Steuerung~~

Compound splitting: *Mausmutation*

<u>Mausm</u>utation

$\longrightarrow$

- Do I know the modifier *Mausm*? No!

Compound splitting: *Mausmutation*

Mausmutation

- And so on...

Compound splitting: *Mausmutation*

Maus|mutation

- The prototype with the highest score will be our split!
- Recurse...

Compound splitting: *Plantage*

Plantage

- Let's try another example...

Compound splitting: *Plantage*

Plantage

$\longrightarrow$

- Do I know the modifier *Plan*? Yes!

Compound splitting: *Plantage*

Plantage

$\longrightarrow$

- Do I know the modifier *Plan*? Yes! Prototypes:
  - -Feststellung
  - -Wert
  - -Fertiger
  - ...

Compound splitting: *Plantage*

Plantage

$\longrightarrow$

- Do I know the modifier *Plan*? Yes! Prototypes:
  - ~~-Feststellung~~
  - ~~-Wert~~
  - ~~-Fertiger~~
  - ...

Compound splitting: *Plantage*

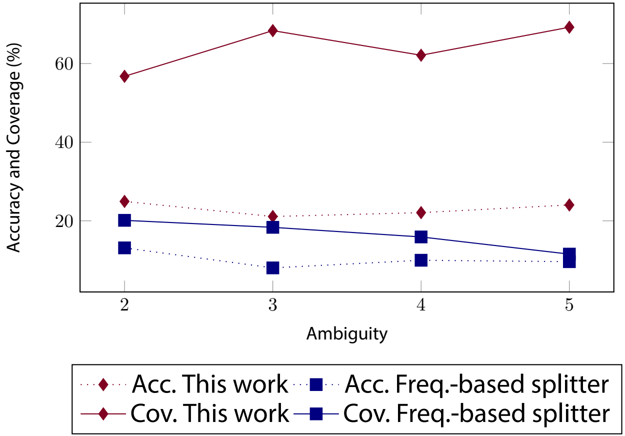
Plantage

- No compound split!

Intrinsic evaluation

- Evaluation on human-annotated dataset (Henrich and Hinrichs, 2011)
  - ~50k compounds
  - only binary splits
- Baseline: Frequency-based Moses compound splitter (Koehn and Knight, 2003)
- We evaluate:
  - Accuracy: $\frac{|\text{correct splits}|}{|\text{compounds}|}$
  - Coverage: $\frac{|\text{compounds split}|}{|\text{compounds}|}$

Intrinsic evaluation



Machine translation experiments (German to English)

	(a) No comp. splitting			(b) Rare: c(w) < 20			(c) All words		
	Splits	BLEU	MTR	Splits	BLEU	MTR	Splits	BLEU	MTR
Moses splitter	0	17.6	25.5	231	17.6	25.7	244	17.9	25.8[A]
This work				744	**18.2**[ABC]	**26.1**[ABC]	1616	17.7	26.3[A]

[A] Stat. sign. against (a) at $p < 0.05$ [B] Stat. sign. against Moses splitter at same c(w) at $p < 0.05$
[C] Stat. sign. against best Moses splitter (c) at $p < 0.05$

Conclusion

- Regularities in semantic vector space can be used to model composition of compounds
- We can extract modifiers and prototypes (Soricut and Och, 2015)
- Compound splitting algorithm:
  - Good intrinsic performance on gold standard
  - Improved translation quality (standard PBMT setup)
  - Especially adept at splitting highly ambiguous compounds

Morphology and Syntax

Joint modeling of morphology and syntax

Paper: A Joint Dependency Model of Morphological and Syntactic Structure for Statistical Machine Translation.

Rico Sennrich and Barry Haddow, EMNLP 2015.

Joint modeling of morphology and syntax

- Languages may differ in degree of morphological synthesis
- Syntactic structure in one language $\rightarrow$ morph. structure in another
- Flat structure is not enough!
  - hierarchical structure between morphemes
  - morphosyntactic constraints
  - selectional preferences
- **Hence:** dependency representation of compounds and particle verbs

Compounds

- Head-final in Germanic languages
- Head determines:
  - agreement in phrase
  - selectional preferences for verbs

Compounds

sie erheben <u>eine</u> Hand|gepäck|<u>gebühr</u>

they charge <u>a</u> carry-on bag <u>fee</u>

Agreement: case, number, gender

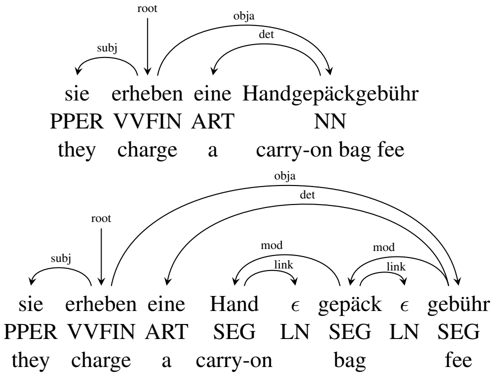
Particle verbs

function/postion	English/German example
finite (main)	he **walks away** quickly er **geht** schnell **weg**
finite (sub.)	[...] because he **walks away** quickly [...] weil er schnell **weggeht**
bare infinitive	he can **walk away** quickly er kann schnell **weggehen**
to/zu-infinitive	he promises **to walk away** quickly er verspricht, schnell **wegzugehen**

Compound representation

- Split compounds and verbs using finite state morphology + statistical corpus evidence
- Noun and adjective compounds
- Compound representation
  - left-branching
  - head of compound $\rightarrow$ head in dep. tree
  - bigram dependency LM can enforce agreement

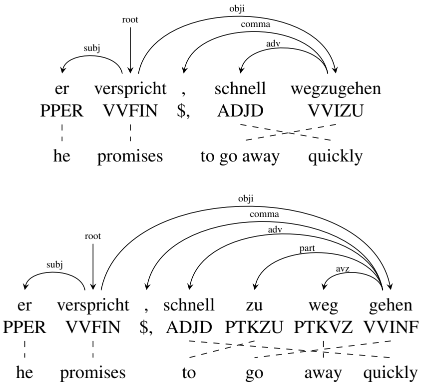
Compound representation



Particle verb representation

- Representation abstracts away from surface realization
- Verb particle reordered to be closest pre modifer to verb
- Dependency links allow enforcement of agreement
- Reduces data sparsity

Particle verb representation



Some technicalities

- Dependencies are converted into constituents
- Dependency language model
- The model should
  - produce new words
  - memorize observed words

$\rightarrow$ compounds need to be constituent

$\rightarrow$ some binarization required

Translation

- String-to-tree model
- Restoring the target sentence:
  - start from tree output
  - merge compounds: concatenate
  - merge particle verbs: apply simple rules
- Experiments English$\rightarrow$German
- Compounds split if occurred $<$ 5 times

Results

system	newstest2014	newstest2015
baseline	20.7	22.0
+split compounds	21.3	22.4
+particle verbs	21.4	22.8
head binarization	20.9	22.7
+split compounds	22.0	23.4
+particle verbs	22.1	23.8
full system	22.6	24.4

- Head binarization matters
- Examples:
  - Staub|sauger|roboter
  - Gravitation|s|wellen
  - NPD|-|Verbot|s|verfahren

Conclusion

- Both particle verb and compound processing helps
- But: particle verbs are rarer!

Question: Does the new representation help in agreement?

- Test 200 rare compound
- Artificially introduce agreement errors
- Original representation accuracy (dep. LM): 55%
- New representation accuracy (dep. LM): 96.5%

Thank You!

Any questions?

References

Henrich, V. and Hinrichs, E. W. (2011). Determining immediate constituents of compounds in GermaNet. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing 2011*.

Koehn, P. and Knight, K. (2003). Empirical methods for compound splitting. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 187–193. Association for Computational Linguistics.

Soricut, R. and Och, F. (2015). Unsupervised morphology induction using word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1627–1637, Denver, Colorado. Association for Computational Linguistics.