Computational Grammar

Week 7: Syntactic Parsing of Tweets

Rob van der Goot
r.van.der.goot@rug.nl

25-02-2019

Outline

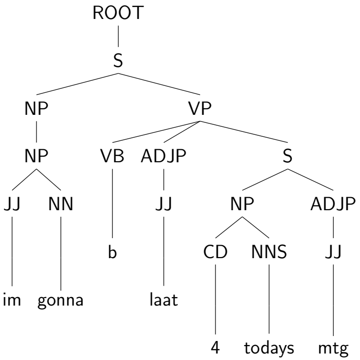
Problem


●●●○○ vodafone
9:41 AM
50 %
Chats
Gertjan
online
Hey GJ, im gonna b laat 4 todays mtg

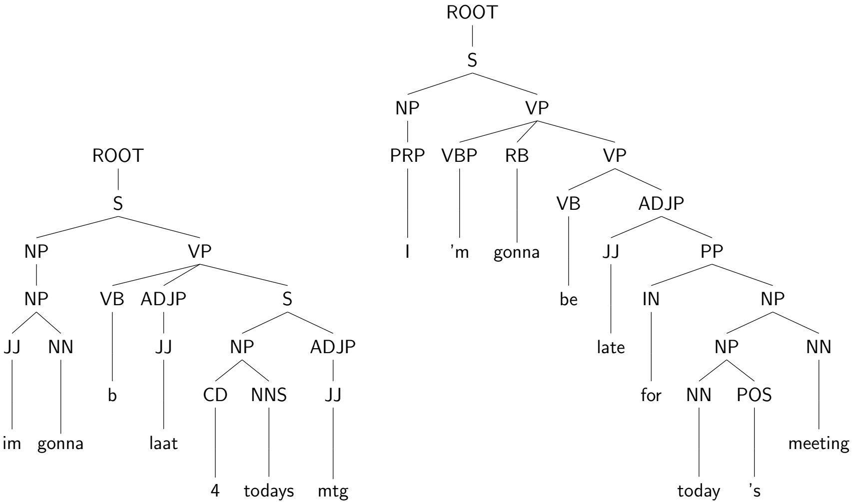
Problem


●●●○○ vodafone
9:41 AM
50 %
Chats
Gertjan
online
Hey GJ, im gonna b laat 4 todays mtg
aight m8

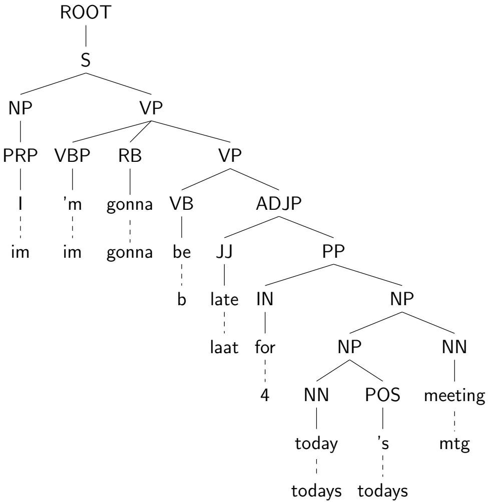
Problem


ROOT
S
NP
VP
NP
VB
ADJP
S
JJ
NN
JJ
NP
ADJP
b
CD
NNS
JJ
im
gonna
laat
4
todays
mtg

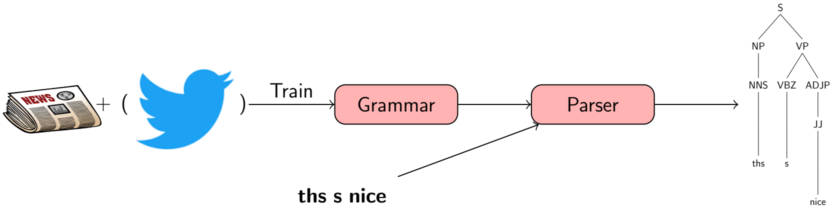
Problem


ROOT
S
NP
VP
NP
VB
ADJP
S
JJ
NN
JJ
NP
ADJP
b
CD
NNS
JJ
im
gonna
laat
4
todays
mtg
ROOT
S
NP
VP
PRP
VBP
RB
VP
VB
ADJP
I
'm
gonna
JJ
PP
be
IN
NP
late
NP
NN
for
NN
POS
meeting
today
's

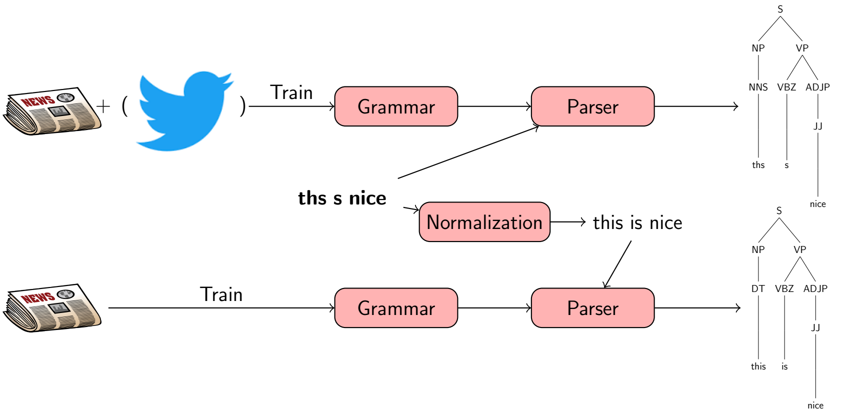
Idea


ROOT
S
NP
VP
PRP
VBP
RB
VP
I
'm
gonna
VB
ADJP
im
im
gonna
be
JJ
PP
b
late
IN
NP
laat
for
NP
NN
4
NN
POS
meeting
today
's
mtg
todays
todays

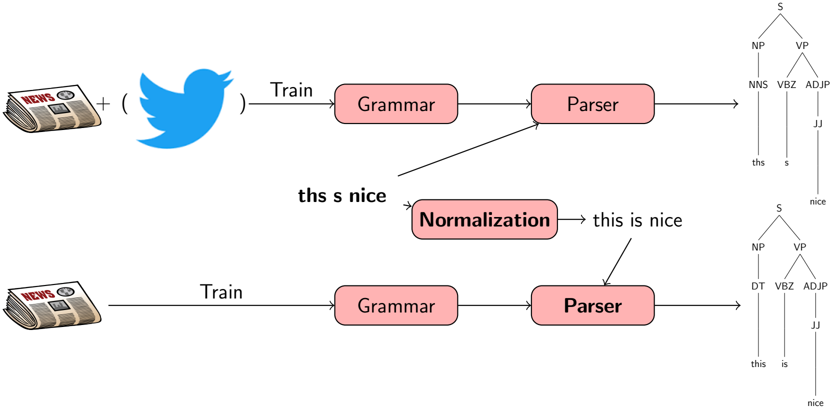
Idea


+ (
)
Train
Grammar
Parser
ths s nice
S
NP
VP
NNS
VBZ
ADJP
JJ
ths
s
nice

Idea


Train
Grammar
Parser
ths s nice
Normalization
this is nice
Train
Grammar
Parser
S
NP
VP
NNS
VBZ
ADJP
JJ
ths
s
nice
S
NP
VP
DT
VBZ
ADJP
JJ
this
is
nice

Idea



+ ( )
Train
Grammar
Parser
S
NP
VP
NNS
VBZ
ADJP
JJ
ths
s
nice
ths s nice
Normalization
this is nice
Train
Grammar
Parser
S
NP
VP
DT
VBZ
ADJP
JJ
this
is
nice

Task

Lexical normalization

- No word reordering
- But can include multi-word replacements

Task

Datasets:

Corpus	Words	Lang.	%normed	1-N	Caps
GhentNorm	12,901	NL	4.8	+	+
TweetNorm	13,542	ES	6.3	+	+
LexNorm1.2	10,576	EN	11.6	−	−
LiLiu	40,560	EN	10.5	−	+
LexNorm2015	73,806	EN	9.1	+	−
Janes-Norm	75,276	SL	15.0	−	+
ReLDI-hr	89,052	HR	9.0	−	+
ReLDI-sr	91,738	SR	8.0	−	+

Task

nee	!	:-D	kzal	nog	es	vriendenlijk	doen	lol
nee	!	:-D	ik zal	nog	eens	vriendelijk	doen	lol

tgaat	goed	,	vdg	rustig	aaan	.
Het gaat	goed	,	vandaag	rustig	aan	.

social	ppl	r	anoying
social	people	are	annoying

aaah	buenoo	esqe	digo	pa	qe	madrugara	este	jajaja
ah	bueno	es que	digo	para	qué	madrugará	este	jajaja

nekomu	je	sarkazm	detektor	crknu
nekomu	je	sarkazem	detektor	crknil

Task

Other data used:

- Aspell dictionaries
- Wikipedia dumps
- Tweets (for South Slavic languages web crawl data)

Task

Mo'Noise

- ~~Detect anomalies~~
- Generate normalization candidates (add original word)
- Rank normalization candidates

Task

original word	mostt	social	ppl	r	troublesome
candidates	mostt	social	ppl	r	troublesome
	most	socials	pol	ri	trouble some
	misty	media	people	rnt	bothersome
	mosttt	socially	pple	ra	troubles

Table: Example of Candidate Generation

Task

Generation:

- Original word
- Aspell
- Word embeddings
- Lookup list
- word.*
- split

Task

Aspell

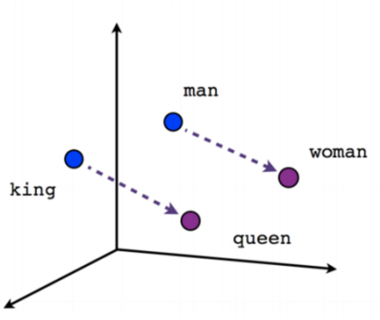
- Based on edit distances (character/phonetic)
- Available for 92 languages

Task

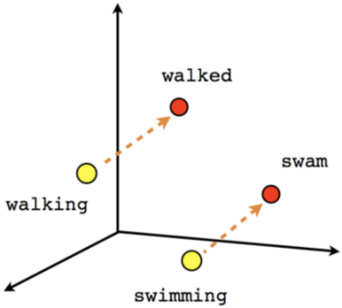
Word embeddings:

- word2vec
- Place words in N-dimensional space
- Based on co-occurences (context)

Task

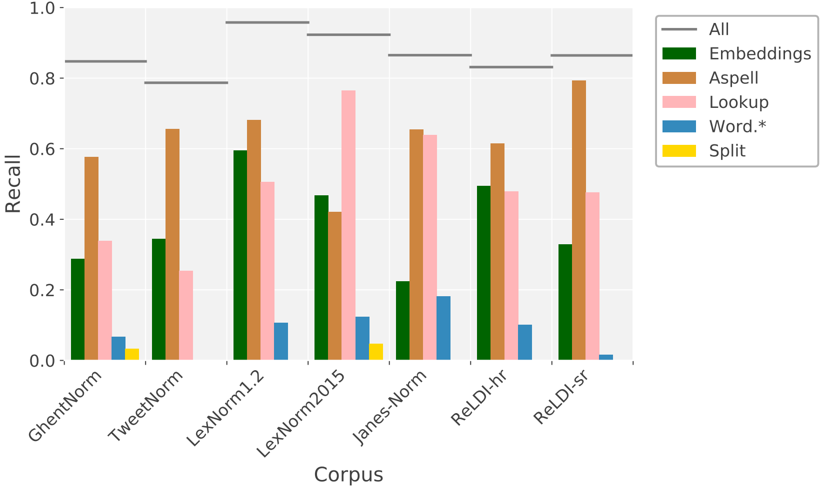


Male-Female



Verb tense

Task



1.0
0.8
0.6
0.4
0.2
0.0
Recall
GhentNorm
TweetNorm
LexNorm1.2
LexNorm2015
Janes-Norm
ReLDI-hr
ReLDI-sr
Corpus
All
Embeddings
Aspell
Lookup
Word.*
Split

Task

Not found:

GhentNorm		LexNorm1.2		LexNorm2015	
neeneenee	nee nee nee	sowi	sorry	trynna	trying to
zijt	bent	neb	nebraska	skepta	sunglasses
bij	die	mo'd	mowed	satnite	saturday night
bwoaja	ja	sumwer	somewhere	tbf	to be fair
jana's	jana 's	thuur	thursday	wada	water

Task

Ranking:

Candidate	Feat1	Feat2	Feat3	...	Gold label
ppl	1.0	0.01	0.42	...	0
pol	0.0	0.00	0.03	...	0
people	0.0	0.24	0.12	...	1
pple	0.0	0.05	0.08	...	0

Task

Features:

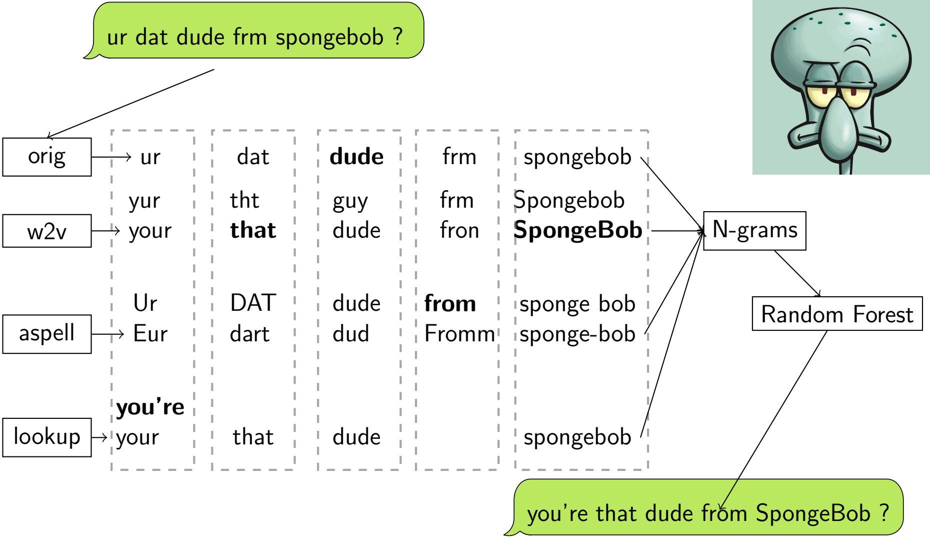
- From generation modules
- N-gram probabilities (based on Wikipedia/Twitter data)
- Dictionary lookup (1/0)
- Character order
- Length
- ContainsAlpha
- OrigWord

Task

Classifier to predict whether gold label $= 1$:

- Random forest classifier
- Rank based on confidence score

Task


ur dat dude frm spongebob ?
orig
ur
dat
dude
frm
spongebob
w2v
yur
your
tht
that
guy
dude
frm
fron
Spongebob
SpongeBob
aspell
Ur
Eur
DAT
dart
dude
dud
from
Fromm
sponge bob
sponge-bob
lookup
you're
your
that
dude
spongebob
N-grams
Random Forest
you're that dude from SpongeBob ?

Task

Comparison to previous work:

Corpus	Prev. state-of-the-art	Metric	Prev.	MoNoise
LexNorm1.2	Li and Liu (2015)	Accuracy	87.58	87.63
LexNorm2015	Jin (2015)	F1	84.21	86.61
GhentNorm	Schulz et al. (2016)	WER	3.2	1.36
TweetNorm	Porta and Sancho (2013)	OOV-Precision	63.4	70.57
Janes L1	Ljubešic et al. (2016)	CER	0.38	0.55
Janes L3	Ljubešic et al. (2016)	CER	1.58	2.38

Task

At least 7 different evaluation metrics!

- F1: unclear, what to do with words which are normalized wrongly?
- BLEU: but word order is known
- WER: but word order is known
- Accuracy over OOVs: detection is not included
- Precision over OOVs: detection is not included
- CER: some words are much more important (lol)
- Accuracy: clear

Task

Accuracy:

- For one corpus, clear
- For multiple corpora: is a score of 96 good?

Task

Accuracy:

- For one corpus, clear
- For multiple corpora: is a score of 96 good?
- So normalize for number of replacements (size of problem)

Task

Baseline:

- leave-as-is
- identity
- copy

$$Acurracy_{baseline} = \frac{notnormalizedwords}{allwords}$$

Task

$$ERR = \frac{Accuracy_{system} - Accuracy_{baseline}}{1.0 - Accuracy_{baseline}} \tag{1}$$

Task

- Easy to interpret: shows percentage of problem solved
- Compare across corpora
- Evaluate the complete normalization task (for more detail, complementary methods can be used)

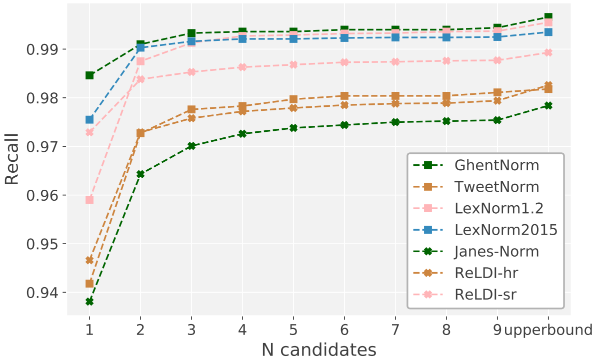
Task

Corpus	ERR	Precision	Recall
GhentNorm	44.62	86.84	50.77
TweetNorm	35.86	90.05	37.09
LexNorm1.2	60.61	78.03	79.12
LexNorm2015	76.15	91.98	80.58
Janes-Norm	67.15	89.62	70.81
ReLDI-hr	51.73	92.17	54.23
ReLDI-sr	57.48	86.43	60.78

Table: Results of MoNoise on the test data.

Task

Results of ranking:

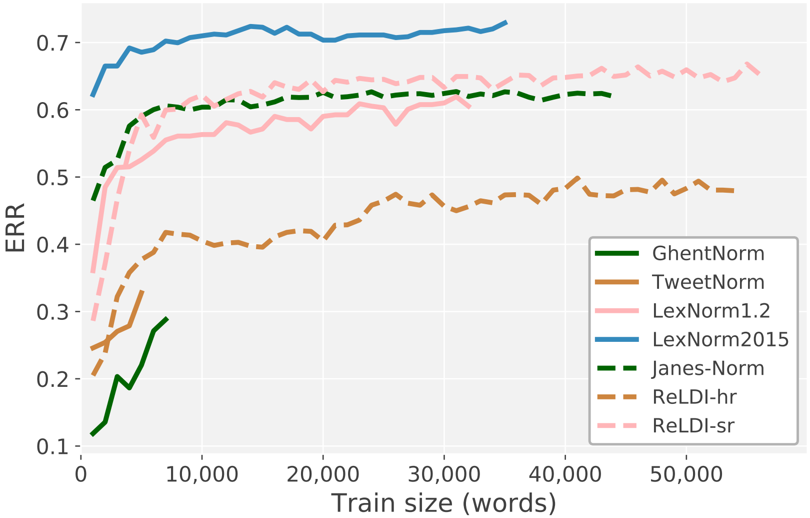


Task

Errors (rough average over all datasets):

- 25%: Normalized wrong word
- 65%: Too conservative (correct word second, original word kept)
- 9%: Not found
- 1%: Ranked wrong

Task



ERR
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
10,000
20,000
30,000
40,000
50,000
Train size (words)
GhentNorm
TweetNorm
LexNorm1.2
LexNorm2015
Janes-Norm
ReLDI-hr
ReLDI-sr

Task

www.let.rug.nl/rob/monoise

Task

Conclusion:

- Modular system is sensible: state-of-the-art for multiple languages
- The generation modules cover almost all cases
- N-gram probabilities are good features
- Bottleneck: decide when to normalize
- Evaluation: many metrics are used, but ERR is better

Outline

Constituency Parsing

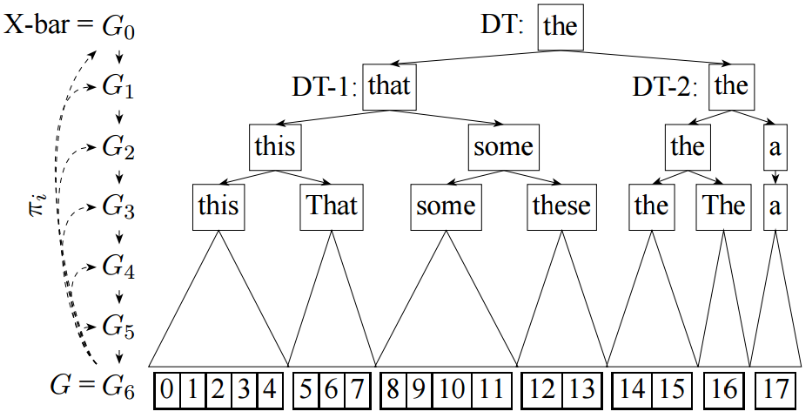
Dataset:

- Jennifer Foster, Ozlem Cetinoglu, Joachim Wagner, Joseph Le Roux, Joakim Nivre, Deirdre Hogan and Josef van Genabith, 2011. From News to Comment: Resources and Benchmarks for Parsing the Language of Web 2.0.
- 519 tweets (250-269)
- Constituency trees (EWT)
- Less noisy compared to normalization corpora

Constituency Parsing

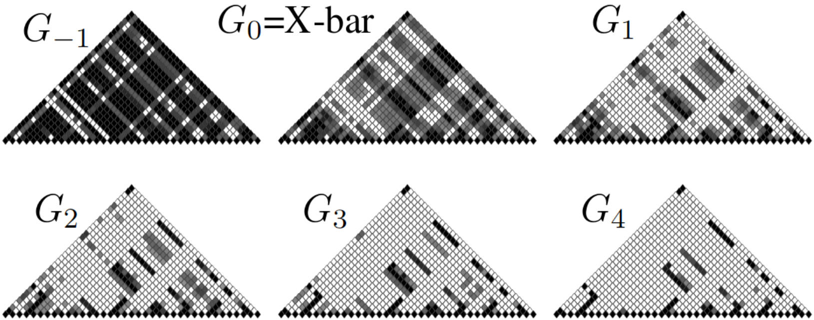
- Berkeley parser (CYK, PCFG-LA)
- Reaches ¿90% F1 on WSJ
- Trained on EWT and WSJ

Constituency Parsing



taken from Petrov and Klein (2007)

Constituency Parsing



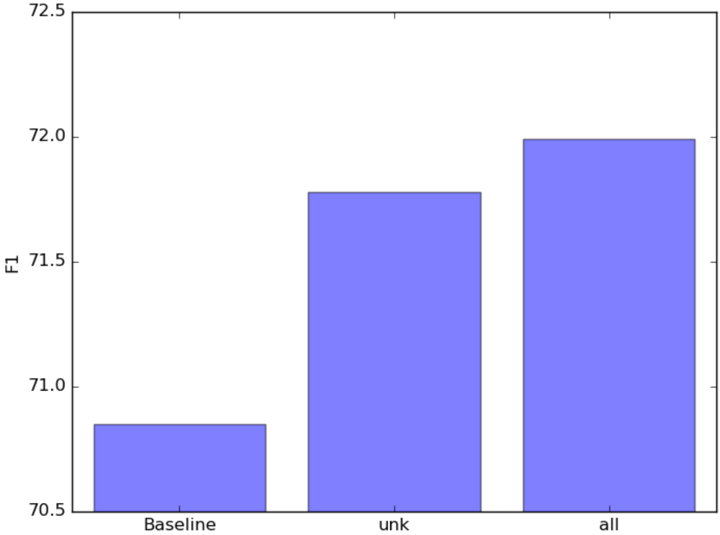
taken from Petrov and Klein (2007)

Constituency Parsing

Two strategies:

- UNK: Only attempt to normalize unknown words (not in training treebank)
- ALL: Attempt to normalize all words

Constituency Parsing



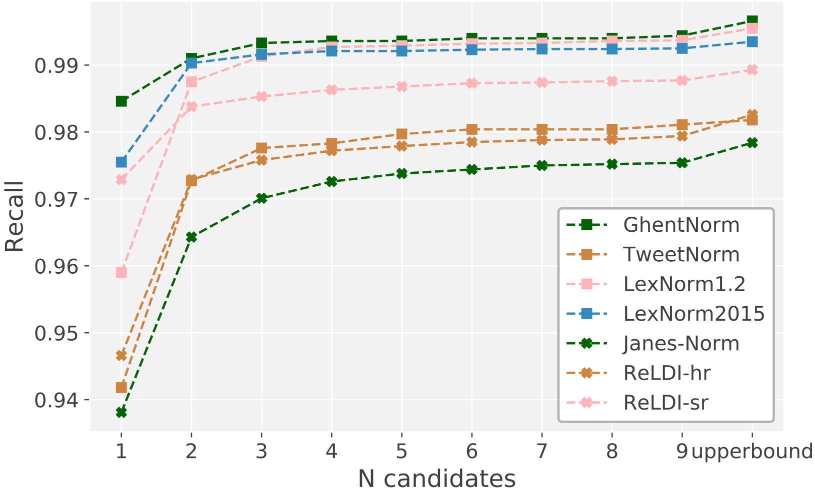
Constituency Parsing

- Nice improvement,
- but:

Constituency Parsing

- Nice improvement,
- but:
- Normalization is not perfect
- Information is lost

Constituency Parsing



Parsing as Intersection

- Bar-hilel (1961)
- "The intersection of a context-free language with a regular language is again a context-free language"

Parsing as Intersection

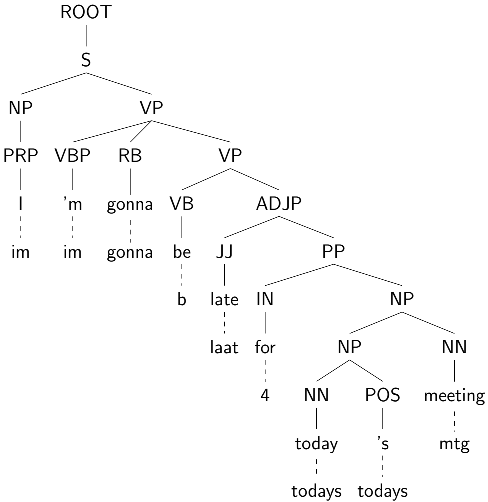
- Bar-hilel (1961)
- "The intersection of a context-free language with a regular language is again a context-free language"
- Ability to find optimal parse tree over a word graph

Parsing as Intersection

In practice:

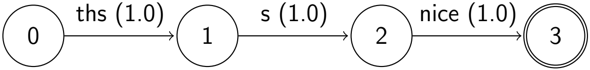
- Treat words as constituents

Parsing as Intersection

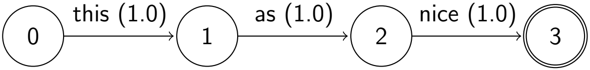


Parsing as Intersection

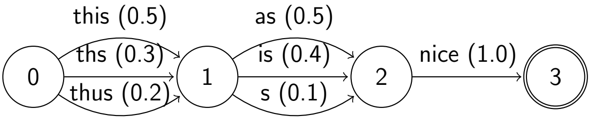
Raw sent:



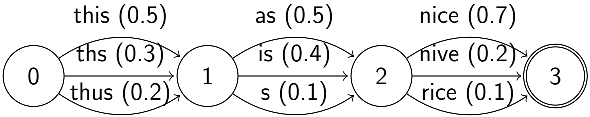
Best norm:



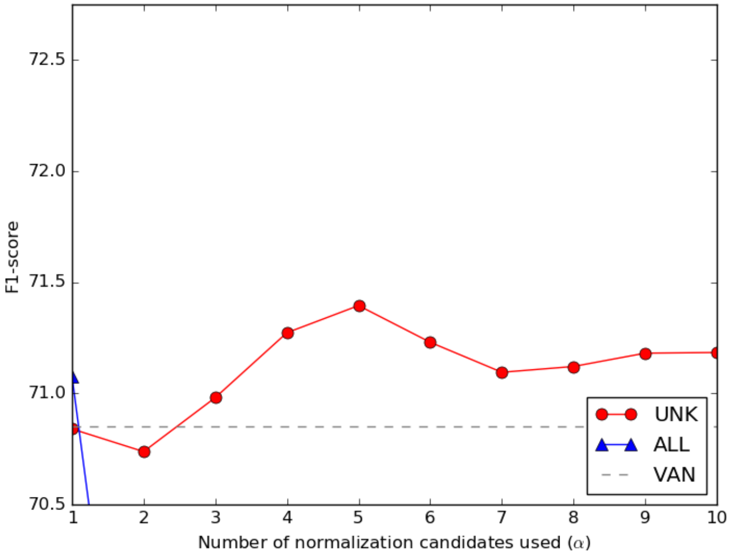
UNK:



ALL:



Parsing as Intersection



Parsing as Intersection

Adjust normalization weight:

$$P_{chart} = (1 + \beta^2) * \frac{P_{norm} * P_{pos}}{(\beta^2 * P_{norm}) + P_{pos}} \quad (2)$$

Parsing as Intersection

Emperically:

$$\beta = 2 \quad (3)$$

Parsing as Intersection

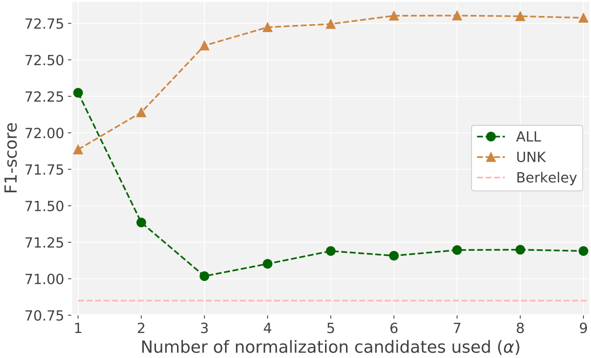
Emperically:

$$\beta = 2 \tag{3}$$

Normalization gets a higher weight than POS tagger

Evaluation

Development data:



Evaluation

Test data:

Parser	dev	test
Stanford parser	66.05	61.95
Berkeley parser	70.85	66.52
Best norm. seq.	72.03	67.06
Integrated norm.	73.14*	67.36*
Gold POS tags	74.98	71.80

Conclusion

- Normalization improves performance of PCFG-LA parser for tweets
- Integrating normalization leads to further improvement

Outline

New Treebank

	format	noisy	size
tweebank	Dependency adapted	+-	929
Denoised web treebank	CoNLL-2008	+	500
EWT	UD	-	16,622
Foster	ptb (constituency)	-	1,000
Foreebank	ptb (constituency)	-	1,000

New Treebank

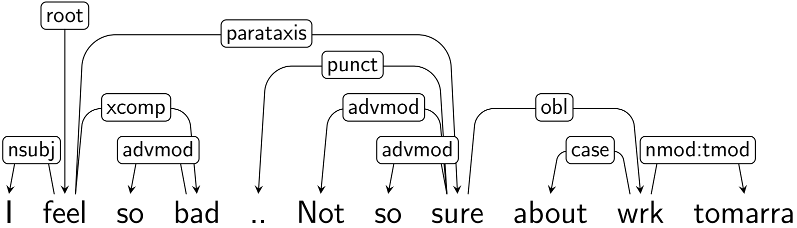
Why?

- Manually corrected train data
- Gold normalization available
- Data should be non-canonical
- UD format

New Treebank

- Pre-filtered to contain non-standard words
- Data from Li and Liu (2015): Owoputi and LexNorm
- 600 Tweets / 10,000 words
- UD2.1 format

New Treebank



New Treebank

Experimental setup:

- Train: English Web Treebank
- Dev: Owoputi
- Test: Lexnorm

New Treebank

Made simultaneously:

- Tweebank 2.0: Liu et al. (2018)
- UD-TwitterAAE: Blodgett et al. (2018)

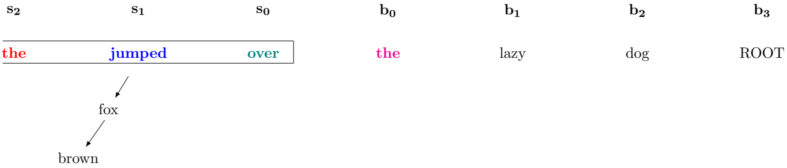
Neural Network parser

Neural networks:

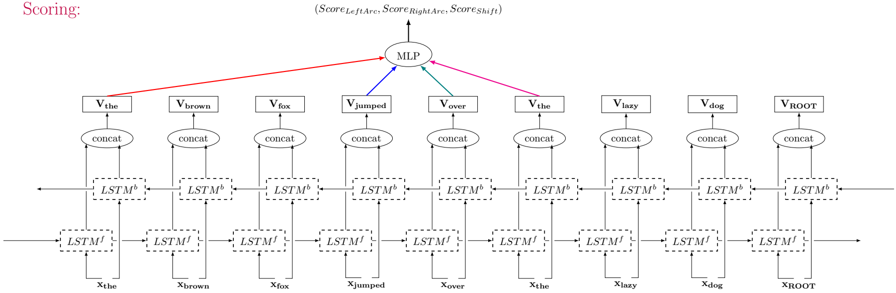
- No manual feature engineering
- Optimizes N features per word
- Words can be represented with a vector of floats

Neural Network parser

Configuration:



Scoring:



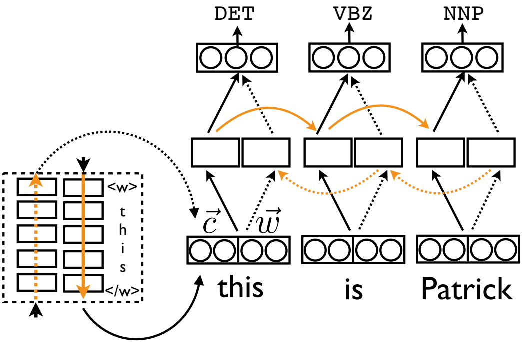
Taken from Kiperwasser and Goldberg (2016)

Neural Network parser

UUparser 2.0 (de Lhoneux et al., 2017)

- Performs well
- Relatively easy to adapt
- No POS tags
- Characters + external embeddings

Neural Network parser

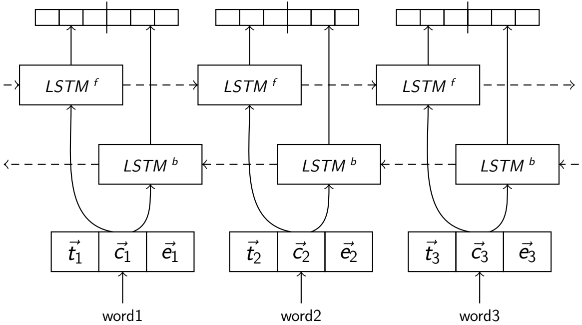


Neural Network parser

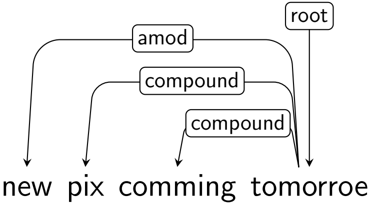
External embeddings:

- trained using word2vec
- 760,744,676 tweets

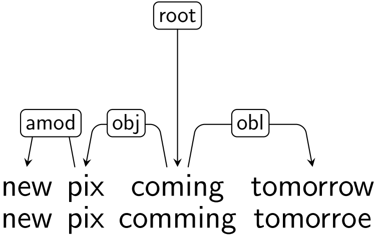
Neural Network parser



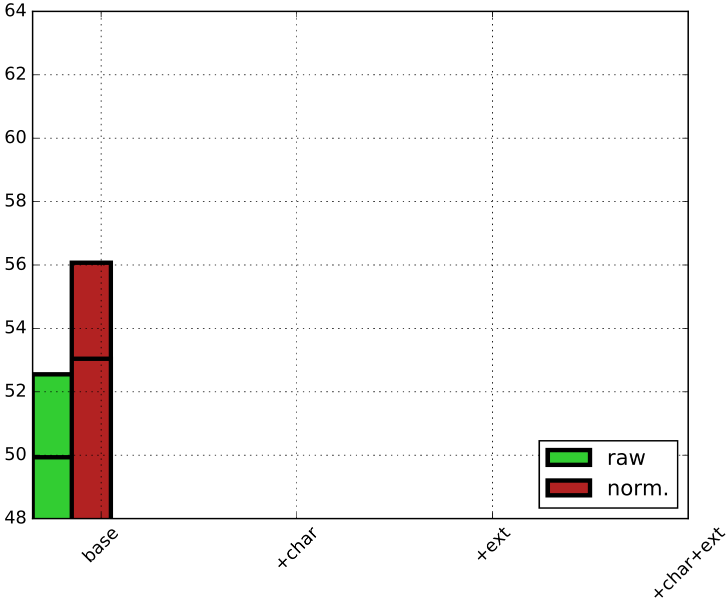
Use Normalization as Pre-processing



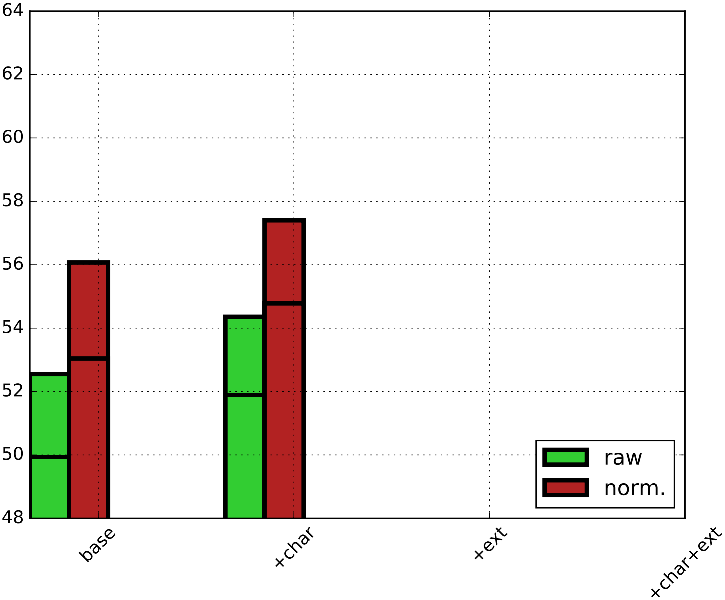
Use Normalization as Pre-processing



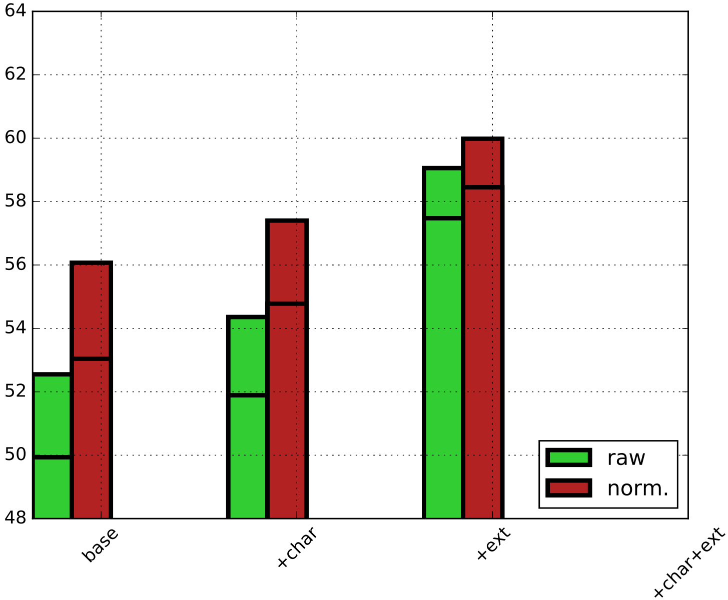
Use Normalization as Pre-processing



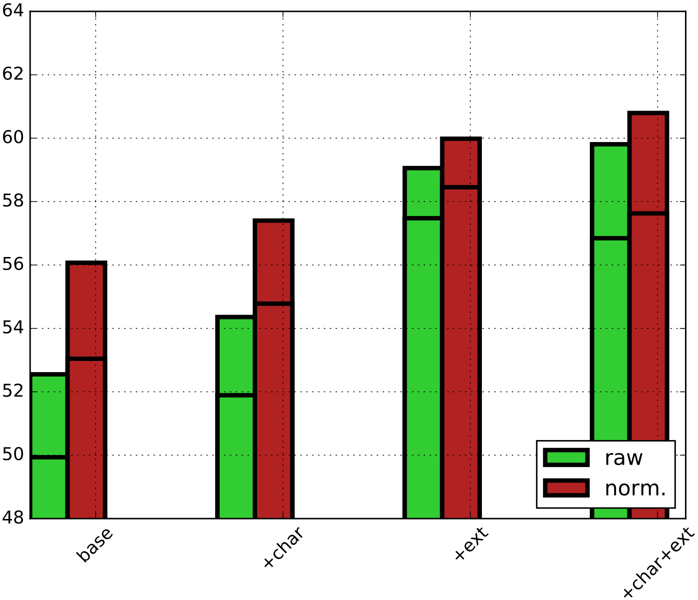
Use Normalization as Pre-processing



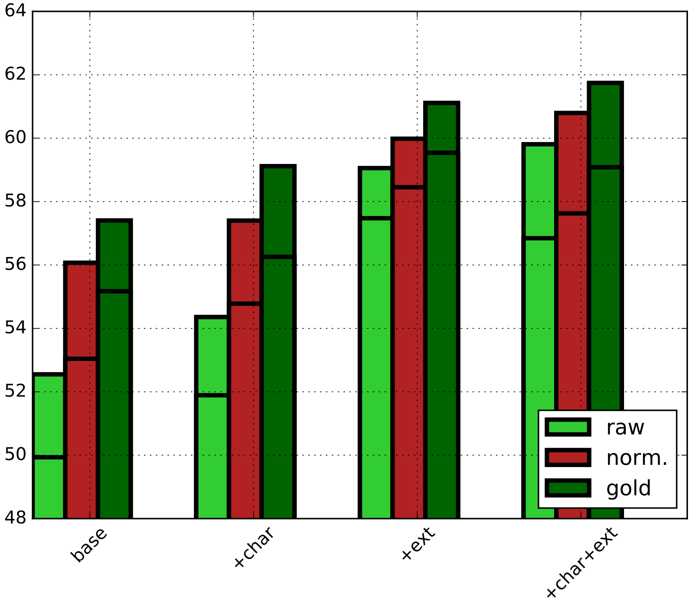
Use Normalization as Pre-processing



Use Normalization as Pre-processing



Use Normalization as Pre-processing



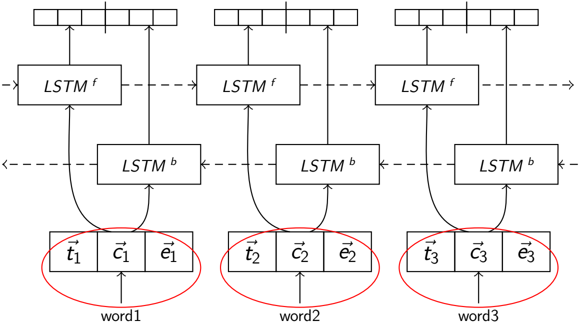
Integrate Normalization

new pix comming tomorroe

Integrate Normalization

new		pix		comming		tomoroe	
new	0.9466	pix	0.7944	coming	0.5684	tomorrow	0.5451
news	0.0315	selfies	0.0882	comming	0.4314	tomoroe	0.3946
knew	0.0111	pictures	0.0559	combing	0.0002	tomorrow's	0.0191
now	0.0063	photos	0.0449	comping	<0.0001	Tagore	0.0174
newt	0.0045	pic	0.0165	common	<0.0001	tomorrows	0.0173

Integrate Normalization



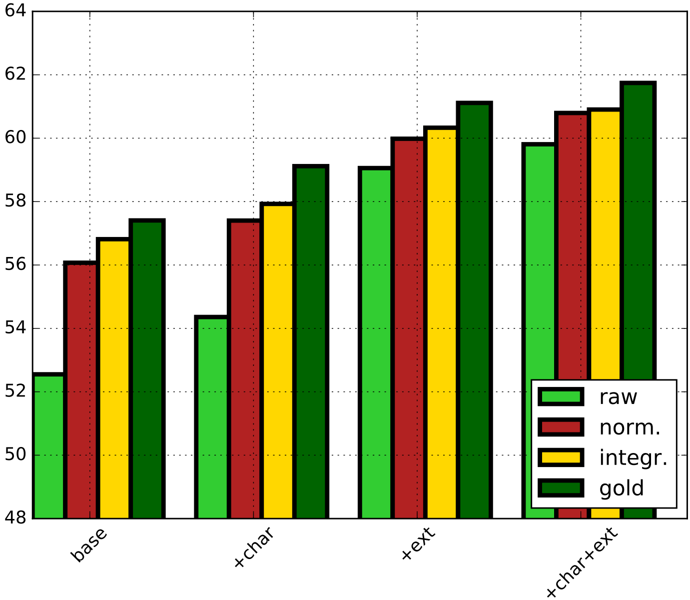
Integrate Normalization

$$\vec{w}_i = \sum_{j=0}^{n} P_{ij} * \vec{n}_{ij}$$

Integrate Normalization

$$\vec{w}_1 = (\vec{new} * 0.9466) + (\vec{news} * 0.0315) + (\vec{knew} * 0.0111) + (\vec{now} * 0.0063) + (\vec{newt} * 0.0045)$$

Integrate Normalization



Integrate Normalization

Test data:

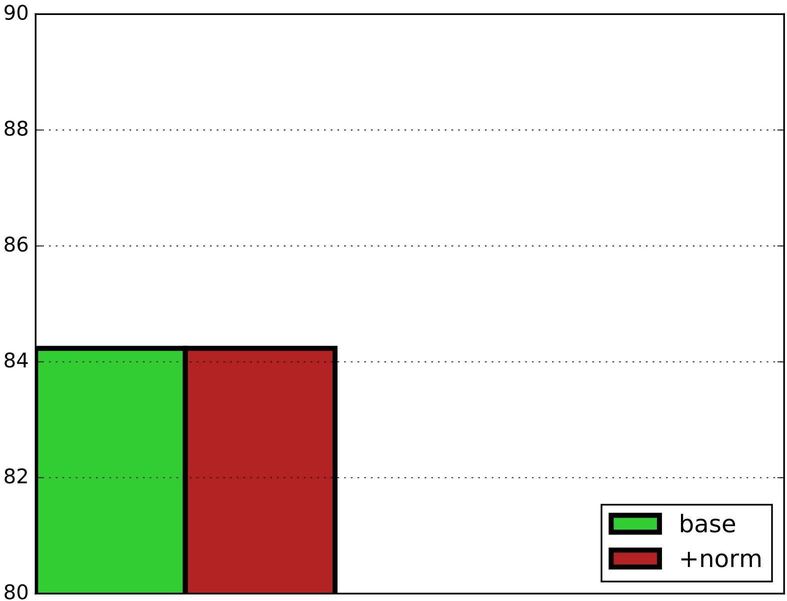
Model	UAS	LAS
raw	70.47	60.16
normalization-		
direct	71.03*	61.83*
integrated	71.15	62.30*
gold	71.45	63.16*

Table: *indicates statistical significance compared to previous entry.

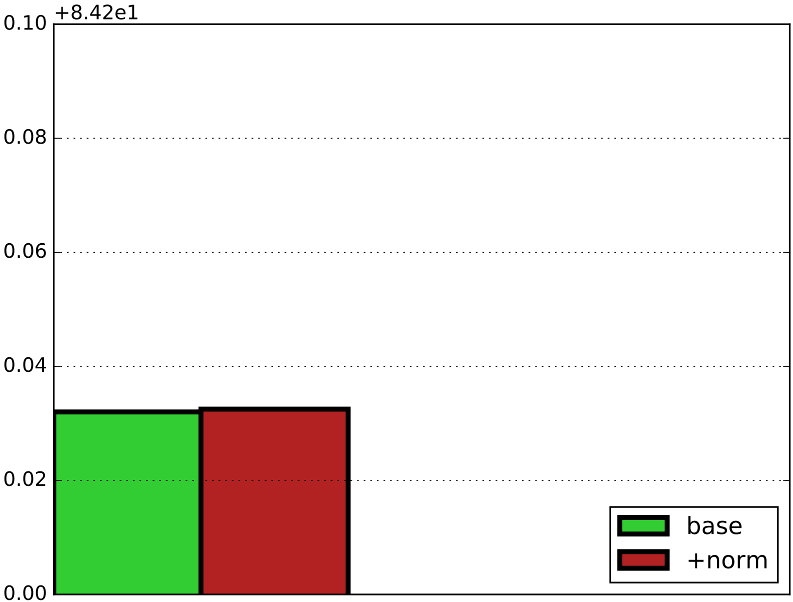
Integrate Normalization

But what about in-domain performance?

Integrate Normalization



Integrate Normalization



Integrate Normalization

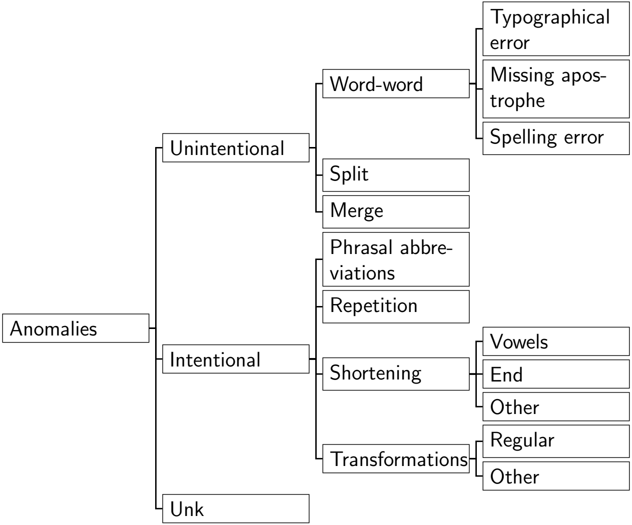
Conclusions:

- Normalization is still helpful on top of character and external embeddings
- Integrating normalization leads to a small but consistent/significant improvement
- Performance +-60% from using gold normalization
- New dataset is publicly available, provides a nice benchmark for domain adaptation

Outline

Future/Current work

Analysis of effect of different types of replacements on parsing:



Future/Current work

Independent UD annotation:

	F1 score
Tokens	97.64
Sentences	100.00
Words	97.52
UPOS	90.31
UAS	76.23
LAS	69.40

Future/Current work

Ma. theses:

- Lexical normalization and POS tagging for Dutch
- Predicting normalization categories (cross-corpus & cross-language)
- Distant supervision for normalization (* 2)

Future/Current work

Bibliography

Su Lin Blodgett, Johnny Wei, and Brendan O'Connor. Twitter universal dependency parsing for African-American and mainstream American English. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Melbourne, Australia, 2018. Association for Computational Linguistics.

Miryam de Lhoneux, Yan Shao, Ali Basirat, Eliyahu Kiperwasser, Sara Stymne, Yoav Goldberg, and Joakim Nivre. From raw text to universal dependencies - look, no tags! In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 207–217, Vancouver, Canada, August 2017. Association for Computational Linguistics.

Ning Jin. NCSU-SAS-Ning: Candidate generation and feature engineering for supervised lexical normalization. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 87–92, Beijing, China, July 2015. Association for Computational Linguistics.