Designing and comparing G2P-type lemmatizers for a morphology-rich language

Steffen Eger,
Goethe University Frankfurt am Main,
Text Technology Lab
steeger@em.uni-frankfurt.de

Goals

- Compare performances of different lemmatization systems for Latin
  - For practical purposes: want to offer Latin preprocessing to the community
- Evaluate how character-level (G2P inspired) string transduction systems perform

Table of contents

Table of contents

Lemmatization

- Task of converting an **inflected form** to its **base form**
  - *playing* $\mapsto$ *play*
  - *gespielt* $\mapsto$ *spielen*
  - *amaveritis* $\mapsto$ *amo*
- Can be done with the help of, e.g., lexicons, but designing lexicons is costly (and boring)
- View the problem as a (machine learning) **string transduction** problem where we want to learn **character level** transformations for translating
  - $\mathbf{x} \mapsto \mathbf{y}$

Broader Context: Preprocessing tools for Latin

- Want to develop NLP tools for Latin as part of the Comphistsem project `www.comphistsem.org`
  - Lexicon (Collex.LA — Mehler et al. 2015): > 8 million word forms
  - Lemmatizers
  - Taggers (Eger, vor der Brück, Mehler, 2015)
  - Dependency parsers
  - See also: `https://prepro.hucompute.org/`

- Traditionally, lemmatization in machine learning is viewed as a problem of *suffix* (and *prefix*) transformation
  - Jursic et al. (2010), Gesmundo and Samardzic (2012)
- In contrast, we compare general purpose string transduction systems with such systems:
  - General purpose string transduction systems, particularly for G2P, have been well-explored
  - Prefix and suffix transformations may not always be sufficient/appropriate; e.g. u/v alternation in Latin or irregular forms
    - cf. *schafft* $\mapsto$ *schaffen*,
    - cf. *schläft* $\mapsto$ *schlafen*

Lemmatization (and related fields such as *inflection generation*) has attracted attention recently

- Durrett and DeNero (2013); Ahlberg, Forsberg, Hulden (2014)
  - paradigm induction from inflection tables
  - inflect an input base-form by matching it to a paradigm seen during training
- Nicolai, Cherry, Kondrak (2015):
  - View inflection generation as a character-level string transduction task (like this work)

Ahlberg, Forsberg, Hulden (2014):

- Paradigm for **schreiben, leihen**, etc.

$x_1 + \mathbf{e} + x_2 + x_3 + \mathbf{en}$	INFINITIVE
$x_1 + \mathbf{e} + x_2 + x_3 + \mathbf{end}$	PRESENT PARTICIPLE
$\mathbf{ge} + x_1 + x_2 + \mathbf{e} + x_3 + \mathbf{en}$	PAST PARTICIPLE
$x_1 + \mathbf{e} + x_2 + x_3 + \mathbf{e}$	PRESENT 1P SG
$x_1 + \mathbf{e} + x_2 + x_3 + \mathbf{st}$	PRESENT 2P SG
$x_1 + \mathbf{e} + x_2 + x_3 + \mathbf{t}$	PRESENT 3P SG

- At test time, match an input form to a paradigm, then generate arbitrary other forms from paradigm

Table of contents

Systems

- **Mate** (Bohnet, 2010): learns shortest edit scripts
- **LemmaGen** (Jursic et al., 2010): learns to transform word form suffixes via 'if-then' rules
- **LemmAsTagging** (Gesmundo and Samardzic, 2012): codes (densely) lemmatization as prefix and suffix transformations; can then lemmatize in context
- **Phonetisaurus** (Novak et al., 2012): Joint G2P n-gram model
- **AliSeTra**: Own discriminative model (in spirit similar to Jiampojamarn et al., 2010)

Systems

- **Mate** (Bohnet, 2010): learns shortest edit scripts
- **LemmaGen** (Jursic et al., 2010): learns to transform word form suffixes via 'if-then' rules
- **LemmAsTagging** (Gesmundo and Samardzic, 2012): codes (densely) lemmatization as prefix and suffix transformations; can then lemmatize in context
- **Phonetisaurus** (Novak et al., 2012): Joint G2P *n*-gram model
- **AliSeTra**: Own discriminative model (in spirit similar to Jiampojamarn et al., 2010)

Training data

All systems take pairs of strings (word form,lemma) as input

ingem**uistis**	ingem**isco**
exmact**auissetis**	exmact**o**
conr**ectvs**	conr**igeo**
emund**atarum**	emund**o**
superinte**xere**	superinte**go**
dispute**bant**	dispute**o**

prine**ipibv**s	prine**p**s
frag**i**	frag**um**
chyrogrilli**o**	chyrogrilli**us**
adversat**vm**	adversat**us**
erupturus	erupturus
sciotheric**orv**m	sciotheric**u**m

LemmAsTagging (Gesmundo and Samardzic, 2012)

- Code suffix and prefix transformations as 4-tuples
  - *gespielt* $\mapsto$ *spielen* $\Longrightarrow (2, \emptyset, 1, \mathrm{en})$
- Allows to view lemmatization as a classification/tagging problem
  - Can lemmatize in context

Phonetisaurus (Novak et al., 2012)

(1) Align training data

d	i	s	s	o	n	verat
d	i	s	s	o	n	o

(2) Train N-gram model on aligned data

(3) At decoding time, apply learned N-gram model

AliSeTra — Align, Segment, Transduce

(1) Align training data

d	i	s	s	o	n	verat
d	i	s	s	o	n	o

(2) Train discriminative model on aligned data (CRF, structured SVM)

(3) At decoding time, first *segment* input string, then apply the CRF

AliSeTra — Align, Segment, Transduce

(1) Align training data

d	i	s	s	o	n	verat
d	i	s	s	o	n	o

(2) Train discriminative model on aligned data (CRF, structured SVM)

(3) At decoding time, first *segment* input string, then apply the CRF:

- Test input: *computaris*

AliSeTra — Align, Segment, Transduce

(1) Align training data

d	i	s	s	o	n	verat
d	i	s	s	o	n	o

(2) Train discriminative model on aligned data (CRF, structured SVM)

(3) At decoding time, first *segment* input string, then apply the CRF:

- Test input: *c-o-m-p-u-t-aris*

AliSeTra — Align, Segment, Transduce

(1) Align training data

d	i	s	s	o	n	verat
d	i	s	s	o	n	o

(2) Train discriminative model on aligned data (CRF, structured SVM)

(3) At decoding time, first *segment* input string, then apply the CRF:

- Test input: *c-o-m-p-u-t-o*

AliSeTra — Align, Segment, Transduce

(1) Align training data

d	i	s	s	o	n	verat
d	i	s	s	o	n	o

(2) Train discriminative model on aligned data (CRF, structured SVM). **Features**: Context features, linear chain features, I use CRF++ (highly not recommended)
Additional features: Intra-subsequence-character features (AliSeTra++)

Running times

On training set of size 100,000

Mate	minutes to hours
LemmaGen	seconds
LAT	depends (days)
Phonetisaurus	minutes
AliSeTra	depends (days)

Table of contents

G2P results

	2,000	5,000	10,000
AliSeTra++	38.33	51.98	61.26
AliSeTra	36.64	52.43	62.13
Phonetisaurus	**44.60**	**57.62**	**66.67**
LemmaGen	2.29	4.42	6.82
-last-4-chars	15.30	22.33	36.82
Mate	0.39	0.76	1.00
-on-training	89.17	97.49	95.26

Table: Word accuracy in % as a function of training set size. G2P data.

Results on word lists

- Extract pairs (form,lemma) from our lexicon and train and test on them
- For different word classes (verbs, adjectives, nouns)
- Indicates the degree to which systems can learn regular morphological phenomena

Verbs

	Avg-InDomain	Avg-OutDomain
AliSeTra	87.89	81.78
AliSeTra++	**88.42**	**83.09**
Phonetisaurus	86.98	73.78
LemmaGen	78.23	76.91
Mate	66.10	64.36

Table: Word accuracy in % for different systems, **verbs**. Each system is trained on 10 random subsets of the training data of size 40,000 each. Average and simple majority vote results indicated. In bold: Statistically indistinguishable best performances.

Nouns

	Avg-InDomain	Avg-OutDomain
AliSeTra	**78.25**	74.11
AliSeTra++	77.76	**74.31**
Phonetisaurus	76.74	72.98
LemmaGen	75.37	72.74
Mate	72.90	70.26

Table: Word accuracy in % for different systems, **nouns**. Each system is trained on 10 random subsets of the training data of size 40,000 each. Average and simple majority vote results indicated. In bold: Statistically indistinguishable best performances.

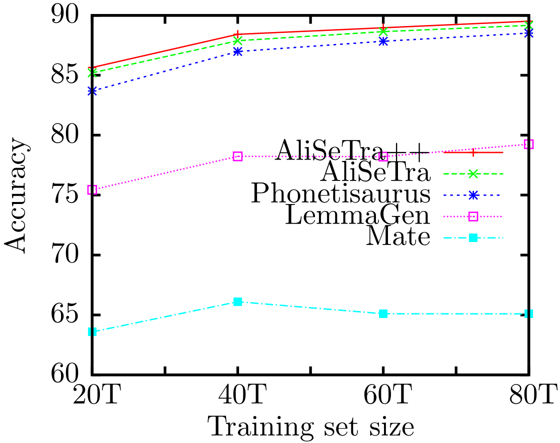

Figure: Word accuracy as a function of training set size. In-domain testing. Verbs

Errors

- Deponent verbs (-*or* vs. -*o*)
- Mix up of conjugation/declination classes
- Gender (-*us* vs. -*um*)
- Lexicon might act as a filtering device

Application: How learned lemmatizers can assist lexicon-based systems

	Token accuracy
TreeTagger	86.23%
TreeTagger+AliSeTra++	88.56%
TreeTagger+LemmaGen	89.37%

Table: TreeTagger lemma token accuracy on a subpart of the PL and accuracy values when the lemmatizer is complemented by our trained lemmatizers.

Evaluation on Text

- Evaluate on real text — different distribution of words (many irregular forms, repetition)
-

	Accuracy
Mate	93.62
LemmaGen	95.47
AliSeTra	95.15
Phonetisaurus	95.40
LaT	95.49

Conclusion

- Systems have different performances depending on evaluation scenario
- If lemmatization in text is the goal, systems perform roughly equally well
- Choosing a fast system may be the best choice

Conclusion

- Must look at *joint* lemmatization and tagging
  - But see: (to appear) 2015. Thomas Müller, Ryan Cotterell, Alex Fraser and Hinrich Schütze. Joint Lemmatization and Morphological Tagging with Lemming. EMNLP
- How can we combine predictions of the different systems (at substring level)?
  - Eger, Steffen. Multiple Many-To-Many Sequence Alignment For Combining String-Valued Variables: A G2P Experiment. In: ACL, 2015

Literature

- Eger,vor der Brück, Mehler. Lexicon-assisted tagging and lemmatization in Latin: A comparison of six taggers and two lemmatization methods. LaTeCH 2015, Beijing, China, 2015.
- Mehler, vor der Brück, Gleim, Geelhaar. Towards a network model of the coreness of texts: An experiment in classifying Latin texts using the TTLab Latin tagger. 2015.
- Ahlberg, Forsberg, Hulden. Semi-supervised learning of morphological paradigms and lexicons. EACL, 2014.
- Durret and DeNero. Supervised learning of complete morphological paradigms. NAACL-HLT, 2013.
- Nicolai, Cherry, Kondrak. Inflection generation as a generative string transduction task. NAACL, 2015.

Thank you!