MorphoDiTa and NameTag
Current State and Future Plans

Milan Straka

ÚFAL Seminar Sedlec-Prčice

15th September 2014
Please do not hesitate to ask questions.
Questions

Please do not hesitate to ask questions.
Please do not hesitate to ask questions.
MorphoDiTa

- **Morphological Dictionary and Tagger**
- implementation of morphological dictionary and POS tagger, performing morphological analysis, morphological generation, POS+lemma tagging and UTF-8 tokenization
**MorphoDiTa**

- **Morphological Dictionary and Tagger**
- implementation of morphological dictionary and POS tagger, performing morphological analysis, morphological generation, POS+lemma tagging and UTF-8 tokenization
MorphoDiTa Goals

Goals

- to have a system
  - providing Czech morphology and POS tagging
    - with clear licences
    - usable in multiple programming languages
    - reasonably efficient in terms of speed, memory complexity and model sizes
  - originally, the plan was to use existing systems
    - failed because of several reasons (unmaintained code, lack of features, inefficiency, etc.)
  - after deciding to develop a new system, further goal arose
    - support morphology of as many languages as possible
Goals

- to have a system
  - providing Czech morphology and POS tagging
  - with clear licences
    - usable in multiple programming languages
    - reasonably efficient in terms of speed, memory complexity and model sizes
- originally, the plan was to use existing systems
  - failed because of several reasons (unmaintained code, lack of features, inefficiency, etc.)
- after deciding to develop a new system, further goal arose
  - support morphology of as many languages as possible
MorphoDiTa Goals

Goals

- to have a system
  - providing Czech morphology and POS tagging
  - with clear licences
  - usable in multiple programming languages
    - reasonably efficient in terms of speed, memory complexity and model sizes
- originally, the plan was to use existing systems
  - failed because of several reasons (unmaintained code, lack of features, inefficiency, etc.)
- after deciding to develop a new system, further goal arose
  - support morphology of as many languages as possible
MorphoDiTa Goals

Goals

- to have a system
  - providing Czech morphology and POS tagging
  - with clear licences
  - usable in multiple programming languages
  - reasonably efficient in terms of speed, memory complexity and model sizes
- originally, the plan was to use existing systems
  - failed because of several reasons (unmaintained code, lack of features, inefficiency, etc.)
- after deciding to develop a new system, further goal arose
  - support morphology of as many languages as possible
MorphoDiTa Goals

Goals

- to have a system
  - providing Czech morphology and POS tagging
  - with clear licences
  - usable in multiple programming languages
  - reasonably efficient in terms of speed, memory complexity and model sizes

- originally, the plan was to use existing systems
  - failed because of several reasons (unmaintained code, lack of features, inefficiency, etc.)
  - after deciding to develop a new system, further goal arose
    - support morphology of as many languages as possible
MorphoDiTa Goals

Goals

- to have a system
  - providing Czech morphology and POS tagging
  - with clear licences
  - usable in multiple programming languages
  - reasonably efficient in terms of speed, memory complexity and model sizes
- originally, the plan was to use existing systems
  - failed because of several reasons (unmaintained code, lack of features, inefficiency, etc.)
- after deciding to develop a new system, further goal arose
  - support morphology of as many languages as possible
MorphoDiTa Goals

MorphoDiTa Models

MorphoDiTa and NameTag

Milan Straka

MorphoDiTa and NameTag
Goals

- to have a system
  - providing Czech morphology and POS tagging
  - with clear licences
  - usable in multiple programming languages
  - reasonably efficient in terms of speed, memory complexity and model sizes
- originally, the plan was to use existing systems
  - failed because of several reasons (unmaintained code, lack of features, inefficiency, etc.)
- after deciding to develop a new system, further goal arose
  - support morphology of as many languages as possible
Morphological Dictionary

- use flat **form** - **lemma** - **tag** triplets on input
- create a binary representation that allows fast analysis and generation and is reasonably compact (should gracefully handle a gigaword)
  - dictionary compression exercise, no linguistics here
- provide guessers for out-of-dictionary words
  - currently two kinds based on prefixes/suffixes
Morphological Dictionary

- use flat form - lemma - tag triplets on input
- create a binary representation that allows fast analysis and generation and is reasonably compact (should gracefully handle a gigaword)
  - dictionary compression exercise, no linguistics here
- provide guessers for out-of-dictionary words
  - currently two kinds based on prefixes/suffixes
Morphological Dictionary

- use flat form - lemma - tag triplets on input
- create a binary representation that allows fast analysis and generation and is reasonably compact (should gracefully handle a gigaword)
  - dictionary compression exercise, no linguistics here
- provide guessers for out-of-dictionary words
  - currently two kinds based on prefixes/suffixes
MorphoDiTa Features

**POS Tagger**

- reimplementation of Morče and Featurama
  - averaged perceptron algorithm with Viterbi decoding
  - manual feature specification
    - easily changed (CRF, ANN, SEARN, etc.)
- uses MorphoDiTa for morphological analysis
- external morphological analysis can be used
- allows custom model training
MorphoDiTa Features

POS Tagger

- reimplementations of Morče and Featurama
  - averaged perceptron algorithm with Viterbi decoding
  - manual feature specification
  - easily changed (CRF, ANN, SEARN, etc.)
- uses MorphoDiTa for morphological analysis
- external morphological analysis can be used
- allows custom model training
MorphoDiTa Features

POS Tagger

- reimplementedation of Morče and Featurama
  - averaged perceptron algorithm with Viterbi decoding
  - manual feature specification
  - easily changed (CRF, ANN, SEARN, etc.)
- uses MorphoDiTa for morphological analysis
  - external morphological analysis can be used
  - allows custom model training
POS Tagger

- reimplementation of Morče and Featurama
  - averaged perceptron algorithm with Viterbi decoding
  - manual feature specification
  - easily changed (CRF, ANN, SEARN, etc.)
- uses MorphoDiTa for morphological analysis
- external morphological analysis can be used
- allows custom model training
MorphoDiTa Features

POS Tagger

- reimplementations of Morče and Featurama
  - averaged perceptron algorithm with Viterbi decoding
  - manual feature specification
  - easily changed (CRF, ANN, SEARN, etc.)
- uses MorphoDiTa for morphological analysis
- external morphological analysis can be used
- allows custom model training
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - would anyone need BSD or some other?
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  
  - would anyone need BSD or some other?

- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)

- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - would anyone need BSD or some other?
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - *would anyone need BSD or some other?*
- library for using the models, binaries for creating them
  - precompiled binaries+library for Linux/Windows/OS X
  - library language bindings for
    - Java (precompiled in the package)
    - Perl (available on CPAN as a standalone package)
    - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
  - Weblicht integration coming soon
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - *would anyone need BSD or some other?*
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - would anyone need BSD or some other?
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  
  *would anyone need BSD or some other?*

- library for using the models, binaries for creating them

- precompiled binaries+library for Linux/Windows/OS X

- library language bindings for
  
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)

- web service running on LINDAT/CLARIN
  
  - Weblicht integration coming soon
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - would anyone need BSD or some other?
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
    - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - *would anyone need BSD or some other?*
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon

MorphoDiTa and NameTag

Milan Straka
MorphoDiTa and NameTag
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - *would anyone need BSD or some other?*
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
  - Weblicht integration coming soon
MorphoDiTa Features

MorphoDiTa Implementation

- implemented in C++11
- available under LGPL licence
  - *would anyone need BSD or some other?*
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
  - Weblicht integration coming soon
Czech Model

Czech Morphological Model

- uses the morphological dictionary developed by prof. Hajič and others
  - recently released under ♥♥♥ CC BY-NC-SA ♥♥♥
  - therefore available also under CC BY-NC-SA licence
- PDT tag set (15 positions) and CoNLL-2009 tag set
- contains guessers for morphological analysis
  - statistical guesser by prof. Hajič
  - prefix guesser compiled by J. Hlaváčová
Czech Model

Czech Morphological Model
- uses the morphological dictionary developed by prof. Hajič and others
  - recently released under ♥♥♥ CC BY-NC-SA ♥♥♥
- therefore available also under CC BY-NC-SA licence
- PDT tag set (15 positions) and CoNLL-2009 tag set
- contains guessers for morphological analysis
  - statistical guesser by prof. Hajič
  - prefix guesser compiled by J. Hlaváčová
Czech Model

Czech Morphological Model

- uses the morphological dictionary developed by prof. Hajič and others
  - recently released under CC BY-NC-SA licence
- therefore available also under CC BY-NC-SA licence
- PDT tag set (15 positions) and CoNLL-2009 tag set
- contains guessers for morphological analysis
  - statistical guesser by prof. Hajič
  - prefix guesser compiled by J. Hlaváčová

MorphoDiTa and NameTag

Milan Straka
Czech Morphological Model

- uses the morphological dictionary developed by prof. Hajič and others
  - recently released under ♥♥♥ CC BY-NC-SA ♥♥♥
  - therefore available also under CC BY-NC-SA licence
- PDT tag set (15 positions) and CoNLL-2009 tag set
  - contains guessers for morphological analysis
    - statistical guesser by prof. Hajič
    - prefix guesser compiled by J. Hlaváčová
Czech Model

Czech Morphological Model

- uses the morphological dictionary developed by prof. Hajič and others
  - recently released under ♥♥♥ CC BY-NC-SA ♥♥♥
- therefore available also under CC BY-NC-SA licence
- PDT tag set (15 positions) and CoNLL-2009 tag set
- contains guessers for morphological analysis
  - statistical guesser by prof. Hajič
  - prefix guesser compiled by J. Hlaváčová
Czech Model

Czech Morphological Model

- uses the morphological dictionary developed by prof. Hajič and others
  - recently released under ♥♥♥ CC BY-NC-SA ♥♥♥
- therefore available also under CC BY-NC-SA licence
- PDT tag set (15 positions) and CoNLL-2009 tag set
- contains guessers for morphological analysis
  - statistical guesser by prof. Hajič
  - prefix guesser compiled by J. Hlaváčová
Czech Model

Czech Morphological Model

- uses the morphological dictionary developed by prof. Hajič and others
  - recently released under ♥♥♥ CC BY-NC-SA ♥♥♥
- therefore available also under CC BY-NC-SA licence
- PDT tag set (15 positions) and CoNLL-2009 tag set
- contains guessers for morphological analysis
  - statistical guesser by prof. Hajič
  - prefix guesser compiled by J. Hlaváčová
Czech POS Tagger Model

- reimplementation of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
  - slightly improved feature set
    - better handling of lemmatization

- trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - `pos_only`: only two first tag letters
  - `no_dia`: no diacritical marks on input text
Czech Model

Czech POS Tagger Model

- reimplementation of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
    - slightly improved feature set
      - better handling of lemmatization
  - trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - pos_only: only two first tag letters
  - no_dia: no diacritical marks on input text
Czech Model

Czech POS Tagger Model

- reimplementation of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
  - slightly improved feature set
    - better handling of lemmatization
- trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - pos_only: only two first tag letters
  - no_dia: no diacritical marks on input text
Czech POS Tagger Model

- reimplementation of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
  - slightly improved feature set
    - better handling of lemmatization
- trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - pos_only: only two first tag letters
  - no_dia: no diacritical marks on input text
Czech Model

Czech POS Tagger Model

- reimplementation of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
  - slightly improved feature set
    - better handling of lemmatization
- trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - pos_only: only two first tag letters
  - no_dia: no diacritical marks on input text
Czech Model

Czech POS Tagger Model

- reimplementation of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
  - slightly improved feature set
    - better handling of lemmatization
- trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - pos_only: only two first tag letters
  - no_dia: no diacritical marks on input text
Czech Model

Czech POS Tagger Model

- reimplemention of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
  - slightly improved feature set
    - better handling of lemmatization

- trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - pos_only: only two first tag letters
  - no_dia: no diacritical marks on input text
Czech POS Tagger Model

- reimplementation of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
  - slightly improved feature set
    - better handling of lemmatization
- trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - pos_only: only two first tag letters
  - no_dia: no diacritical marks on input text
Czech POS Tagger Model

- reimplementation of Morče
  - same algorithm (averaged perceptron)
    - improved to be able to train on features not present in golden data
  - slightly improved feature set
    - better handling of lemmatization
- trained on PDT 2.5
- also available under CC BY-NC-SA licence
- several additional variants
  - pos\_only: only two first tag letters
  - no\_dia: no diacritical marks on input text
Czech Morphology Performance

- Czech Morfflex contains 120M form-tag, 1M unique lemmas, 3992 tags; total size 6.7GB
- binary form of the dictionary uses 2MB (3000 smaller)
- analysis: \( \approx 600k \) analyzed forms per sec
- generation \( \approx 1M \) generated forms per sec

Czech POS Tagger Performance

<table>
<thead>
<tr>
<th>Tagger</th>
<th>Task</th>
<th>Accuracy</th>
<th>Words/s</th>
<th>Model size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morče</td>
<td>tag</td>
<td>95.67%</td>
<td>1K</td>
<td>178MB</td>
</tr>
<tr>
<td>Featurama</td>
<td>tag</td>
<td>95.66%</td>
<td>2K</td>
<td>210MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag</td>
<td>95.75%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma</td>
<td>97.80%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma+tag</td>
<td>95.03%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag-first two pos.</td>
<td>99.18%</td>
<td>200K</td>
<td>2MB</td>
</tr>
</tbody>
</table>
Czech Model

Czech Morphology Performance

- Czech Morfflex contains 120M form-tag, 1M unique lemmas, 3992 tags; total size 6.7GB
- binary form of the dictionary uses 2MB (3000 smaller)
- analysis: ≈ 600k analyzed forms per sec
- generation ≈ 1M generated forms per sec

Czech POS Tagger Performance

<table>
<thead>
<tr>
<th>Tagger</th>
<th>Task</th>
<th>Accuracy</th>
<th>Words/s</th>
<th>Model size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morče</td>
<td>tag</td>
<td>95.67%</td>
<td>1K</td>
<td>178MB</td>
</tr>
<tr>
<td>Featurama</td>
<td>tag</td>
<td>95.66%</td>
<td>2K</td>
<td>210MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag</td>
<td>95.75%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma</td>
<td>97.80%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma+tag</td>
<td>95.03%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag-first two pos.</td>
<td>99.18%</td>
<td>200K</td>
<td>2MB</td>
</tr>
</tbody>
</table>

Milan Straka
MorphoDiTa and NameTag
Czech Model

Czech Morphology Performance

- Czech Morfflex contains 120M form-tag, 1M unique lemmas, 3992 tags; total size 6.7GB
- binary form of the dictionary uses 2MB (3000 smaller)
- analysis: $\approx 600k$ analyzed forms per sec
- generation $\approx 1M$ generated forms per sec

Czech POS Tagger Performance

<table>
<thead>
<tr>
<th>Tagger</th>
<th>Task</th>
<th>Accuracy</th>
<th>Words/s</th>
<th>Model size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morče</td>
<td>tag</td>
<td>95.67%</td>
<td>1K</td>
<td>178MB</td>
</tr>
<tr>
<td>Featurama</td>
<td>tag</td>
<td>95.66%</td>
<td>2K</td>
<td>210MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag</td>
<td>95.75%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma</td>
<td>97.80%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma+tag</td>
<td>95.03%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag-first two pos.</td>
<td>99.18%</td>
<td>200K</td>
<td>2MB</td>
</tr>
</tbody>
</table>
Czech Model

Czech Morphology Performance
- Czech Morfflex contains 120M form-tag, 1M unique lemmas, 3992 tags; total size 6.7GB
- binary form of the dictionary uses 2MB (3000 smaller)
- analysis: ≈ 600k analyzed forms per sec
- generation ≈ 1M generated forms per sec

Czech POS Tagger Performance

<table>
<thead>
<tr>
<th>Tagger</th>
<th>Task</th>
<th>Accuracy</th>
<th>Words/s</th>
<th>Model size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morče</td>
<td>tag</td>
<td>95.67%</td>
<td>1K</td>
<td>178MB</td>
</tr>
<tr>
<td>Featurama</td>
<td>tag</td>
<td>95.66%</td>
<td>2K</td>
<td>210MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag</td>
<td>95.75%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma</td>
<td>97.80%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma+tag</td>
<td>95.03%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag-first two pos.</td>
<td>99.18%</td>
<td>200K</td>
<td>2MB</td>
</tr>
</tbody>
</table>
Czech Morphology Performance

- Czech Morfflex contains 120M form-tag, 1M unique lemmas, 3992 tags; total size 6.7GB
- binary form of the dictionary uses 2MB (3000 smaller)
- analysis: ≈ 600k analyzed forms per sec
- generation ≈ 1M generated forms per sec

Czech POS Tagger Performance

<table>
<thead>
<tr>
<th>Tagger</th>
<th>Task</th>
<th>Accuracy</th>
<th>Words/s</th>
<th>Model size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morče</td>
<td>tag</td>
<td>95.67%</td>
<td>1K</td>
<td>178MB</td>
</tr>
<tr>
<td>Featurama</td>
<td>tag</td>
<td>95.66%</td>
<td>2K</td>
<td>210MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag</td>
<td>95.75%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma</td>
<td>97.80%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>lemma+tag</td>
<td>95.03%</td>
<td>10K</td>
<td>16MB</td>
</tr>
<tr>
<td>MorphoDiTa</td>
<td>tag-first two pos.</td>
<td>99.18%</td>
<td>200K</td>
<td>2MB</td>
</tr>
</tbody>
</table>
English Model

English Morphological Model

- morphological analyzer is a reimplementation of
  - POS tag analyzer Morphium by Johanka
    - lemmatizer developed by Martin Popel, based on morpha analyzer
  - morphological generation is performed by running the morphological analyzer on an English word list and using the result as a morphological dictionary
    - (SCOWL – Spell Checker Oriented Word Lists)

English POS Tagger Model

- trained on standard parts of WSJ
- released under CC BY-NC-SA licence
  - quite surprising decision caused by the fact that several European layers agreed with each other on this matter
English Morphological Model

- morphological analyzer is a reimplementation of
  - POS tag analyzer Morphium by Johanka
  - lemmatizer developed by Martin Popel, based on morpha analyzer

- morphological generation is performed by running the morphological analyzer on an English word list and using the result as a morphological dictionary
  - (SCOWL – Spell Checker Oriented Word Lists)

English POS Tagger Model

- trained on standard parts of WSJ
- released under CC BY-NC-SA licence
  - quite surprising decision caused by the fact that several European layers agreed with each other on this matter
English Model

English Morphological Model

- morphological analyzer is a reimplementation of
  - POS tag analyzer Morphium by Johanka lemmerizer developed by Martin Popel, based on morpha analyzer
- morphological generation is performed by running the morphological analyzer on an English word list and using the result as a morphological dictionary
  - (SCOWL – Spell Checker Oriented Word Lists)

English POS Tagger Model

- trained on standard parts of WSJ
- released under CC BY-NC-SA licence
  - quite surprising decision caused by the fact that several European layers agreed with each other on this matter
English Model

English Morphological Model

- morphological analyzer is a reimplementation of
  - POS tag analyzer Morphium by Johanka
  - lemmatizer developed by Martin Popel, based on morpha analyzer

- morphological generation is performed by running the morphological analyzer on an English word list and using the result as a morphological dictionary
  - (SCOWL – Spell Checker Oriented Word Lists)

English POS Tagger Model

- trained on standard parts of WSJ
- released under CC BY-NC-SA licence
  - quite surprising decision caused by the fact that several European layers agreed with each other on this matter
English Model

English Morphological Model

- morphological analyzer is a reimplementation of
  - POS tag analyzer Morphium by Johanka
  - lemmatizer developed by Martin Popel, based on morpha analyzer

- morphological generation is performed by running the morphological analyzer on an English word list and using the result as a morphological dictionary
  - (SCOWL – Spell Checker Oriented Word Lists)

English POS Tagger Model

- trained on standard parts of WSJ
- released under CC BY-NC-SA licence

  quite surprising decision caused by the fact that several European layers agreed with each other on this matter
English Model

**English Morphological Model**
- morphological analyzer is a reimplementation of
  - POS tag analyzer Morphium by Johanka
  - lemmatizer developed by Martin Popel, based on *morpha* analyzer
- morphological generation is performed by running the morphological analyzer on an English word list and using the result as a morphological dictionary
  - *(SCOWL – Spell Checker Oriented Word Lists)*

**English POS Tagger Model**
- trained on standard parts of WSJ
- released under CC BY-NC-SA licence
  - quite surprising decision caused by the fact that several European layers agreed with each other on this matter
Slovak
- morphological dictionary by prof. Hajič and others
- tag set is a “translated” version of PDT
  - mapping to the tag set of SNC may be created if requested
- tagger trained on PDT 2.5 translated by Česílko
- because of licensing issues, currently we do not utilize Slovak National Corpus

Swedish
- we can use (part of) SUC corpus
- no morphological dictionary, guesser only

Arabic
- ElixirFM by O. Smrž as morphological dictionary
- Prague Arabic Dependency Treebank for POS tagger
**Slovak**
- morphological dictionary by prof. Hajič and others
  - tag set is a “translated” version of PDT
    - mapping to the tag set of SNC may be created if requested
  - tagger trained on PDT 2.5 translated by Česílko
  - because of licensing issues, currently we do not utilize Slovak National Corpus

**Swedish**
- we can use (part of) SUC corpus
- no morphological dictionary, guesser only

**Arabic**
- ElixirFM by O. Smrž as morphological dictionary
- Prague Arabic Dependency Treebank for POS tagger
More Languages

Slovak
- morphological dictionary by prof. Hajič and others
- tag set is a “translated” version of PDT
  - mapping to the tag set of SNC may be created if requested
- tagger trained on PDT 2.5 translated by Česílko
- because of licensing issues, currently we do not utilize Slovak National Corpus

Swedish
- we can use (part of) SUC corpus
- no morphological dictionary, guesser only

Arabic
- ElixirFM by O. Smrž as morphological dictionary
- Prague Arabic Dependency Treebank for POS tagger
Slovak
- morphological dictionary by prof. Hajič and others
- tag set is a “translated” version of PDT
  - mapping to the tag set of SNC may be created if requested
- tagger trained on PDT 2.5 translated by Česílko
  - because of licensing issues, currently we do not utilize Slovak National Corpus

Swedish
- we can use (part of) SUC corpus
- no morphological dictionary, guesser only

Arabic
- ElixirFM by O. Smrž as morphological dicionary
- Prague Arabic Dependency Treebank for POS tagger
More Languages

Slovak
- morphological dictionary by prof. Hajič and others
- tag set is a “translated” version of PDT
  - mapping to the tag set of SNC may be created if requested
- tagger trained on PDT 2.5 translated by Česílko
- because of licensing issues, currently we do not utilize Slovak National Corpus

Swedish
- we can use (part of) SUC corpus
- no morphological dictionary, guesser only

Arabic
- ElixirFM by O. Smrž as morphological dictionary
- Prague Arabic Dependency Treebank for POS tagger
More Languages

Slovak
- morphological dictionary by prof. Hajič and others
- tag set is a “translated” version of PDT
  - mapping to the tag set of SNC may be created if requested
- tagger trained on PDT 2.5 translated by Česílko
- because of licensing issues, currently we do not utilize Slovak National Corpus

Swedish
- we can use (part of) SUC corpus
- no morphological dictionary, guesser only

Arabic
- ElixirFM by O. Smrž as morphological dictionary
- Prague Arabic Dependency Treebank for POS tagger
Slovak
- morphological dictionary by prof. Hajič and others
- tag set is a “translated” version of PDT
  - mapping to the tag set of SNC may be created if requested
- tagger trained on PDT 2.5 translated by Česílko
- because of licensing issues, currently we do not utilize Slovak National Corpus

Swedish
- we can use (part of) SUC corpus
  - no morphological dictionary, guesser only

Arabic
- ElixirFM by O. Smrž as morphological dictionary
- Prague Arabic Dependency Treebank for POS tagger
Slovak
- morphological dictionary by prof. Hajič and others
- tag set is a “translated” version of PDT
  - mapping to the tag set of SNC may be created if requested
- tagger trained on PDT 2.5 translated by Česílko
- because of licensing issues, currently we do not utilize Slovak National Corpus

Swedish
- we can use (part of) SUC corpus
- no morphological dictionary, guesser only

Arabic
- ElixirFM by O. Smrž as morphological dicionary
- Prague Arabic Dependency Treebank for POS tagger
More Languages

- Slovak
  - morphological dictionary by prof. Hajič and others
  - tag set is a “translated” version of PDT
    - mapping to the tag set of SNC may be created if requested
  - tagger trained on PDT 2.5 translated by Česílko
  - because of licensing issues, currently we do not utilize Slovak National Corpus

- Swedish
  - we can use (part of) SUC corpus
  - no morphological dictionary, guesser only

- Arabic
  - ElixirFM by O. Smrž as morphological dictionary
  - Prague Arabic Dependency Treebank for POS tagger
More Languages

- **Slovak**
  - morphological dictionary by prof. Hajič and others
  - tag set is a “translated” version of PDT
    - mapping to the tag set of SNC may be created if requested
  - tagger trained on PDT 2.5 translated by Česílko
  - because of licensing issues, currently we do not utilize Slovak National Corpus

- **Swedish**
  - we can use (part of) SUC corpus
  - no morphological dictionary, guesser only

- **Arabic**
  - ElixirFM by O. Smrž as morphological dictionary
  - Prague Arabic Dependency Treebank for POS tagger
More Languages

**Slovak**
- morphological dictionary by prof. Hajič and others
- tag set is a “translated” version of PDT
  - mapping to the tag set of SNC may be created if requested
- tagger trained on PDT 2.5 translated by Česílko
- because of licensing issues, currently we do not utilize Slovak National Corpus

**Swedish**
- we can use (part of) SUC corpus
- no morphological dictionary, guesser only

**Arabic**
- ElixirFM by O. Smrž as morphological dictionary
- Prague Arabic Dependency Treebank for POS tagger
Guesser

Morphological Analysis Guesser

- **currently we only use guesser in Czech morphology**
- we will implement (hopefully improved) training of the morphological analysis guesser in MorphoDiTa
- that would allow to create POS taggers using disambiguated data only, even for languages with rich morphology
  - create the dictionary from disambiguated data only and train the guesser to analyse words with similar prefixes/suffixes patterns
- nevertheless, such guesser cannot be used when performing morphological generation
Morphological Analysis Guesser

- currently we only use guesser in Czech morphology
- we will implement (hopefully improved) training of the morphological analysis guesser in MorphoDiTa
- that would allow to create POS taggers using disambiguated data only, even for languages with rich morphology
  - create the dictionary from disambiguated data only and train the guesser to analyse words with similar prefixes/suffixes patterns
- nevertheless, such guesser cannot be used when performing morphological generation
Morphological Analysis Guesser

- currently we only use guesser in Czech morphology
- we will implement (hopefully improved) training of the morphological analysis guesser in MorphoDiTa
- that would allow to create POS taggers using disambiguated data only, even for languages with rich morphology
  - create the dictionary from disambiguated data only and train the guesser to analyse words with similar prefixes/suffixes patterns
- nevertheless, such guesser cannot be used when performing morphological generation
**Morphological Analysis Guesser**

- currently we only use guesser in Czech morphology
- we will implement (hopefully improved) training of the morphological analysis guesser in MorphoDiTa
- that would allow to create POS taggers using disambiguated data only, even for languages with rich morphology
  - create the dictionary from disambiguated data only and train the guesser to analyse words with similar prefixes/suffixes patterns
- nevertheless, such guesser cannot be used when performing morphological generation
Morphological Analysis Guesser

- currently we only use guesser in Czech morphology
- we will implement (hopefully improved) training of the morphological analysis guesser in MorphoDiTa
- that would allow to create POS taggers using disambiguated data only, even for languages with rich morphology
  - create the dictionary from disambiguated data only and train the guesser to analyse words with similar prefixes/suffixes patterns
- nevertheless, such guesser cannot be used when performing morphological generation
Creating Morphological Dictionaries

- an extension to morphological analysis guesser
- create flat form - lemma - tag morphological dictionary using disambiguated data only
- can be used for both morphological analysis and morphological generation
- still a research area (ideas welcome)

More Semi-supervised Training

- make use of available large corpora
Further Improvements

Creating Morphological Dictionaries
- an extension to morphological analysis guesser
- create flat form - lemma - tag morphological dictionary using disambiguated data only
  - can be used for both morphological analysis and morphological generation
  - still a research area (ideas welcome)

More Semi-supervised Training
- make use of available large corpora
Further Improvements

Creating Morphological Dictionaries
- an extension to morphological analysis guesser
- create flat form - lemma - tag morphological dictionary using disambiguated data only
- can be used for both morphological analysis and morphological generation
- still a research area (ideas welcome)

More Semi-supervised Training
- make use of available large corpora
Further Improvements

Creating Morphological Dictionaries

- an extension to morphological analysis guesser
- create flat form - lemma - tag morphological dictionary using disambiguated data only
- can be used for both morphological analysis and morphological generation
- still a research area (ideas welcome)

More Semi-supervised Training

- make use of available large corpora
Further Improvements

Creating Morphological Dictionaries

- an extension to morphological analysis guesser
- create flat form - lemma - tag morphological dictionary using disambiguated data only
- can be used for both morphological analysis and morphological generation
- still a research area (ideas welcome)

More Semi-supervised Training

- make use of available large corpora
Any Ideas?

Is there anything you would like in MorphoDiTa?
**NameTag**

- **Named entity tagger**
- named entity recognizer build upon MorphoDiTa
**NameTag**

- **Named entity tagger**
- named entity recognizer build upon MorphoDiTa
NameTag Features

- identifies and classified named entities
- both machine learning model and manual rules can be used at any point in the pipeline
- supervised machine learning model
  - neural network classifier produces BILOU class+named entity type for every word
  - Viterbi decoding of all possible BILOU assignments
  - based on (Straková et al., 2013)
- uses MorphoDiTa to obtain POS tags
NameTag Features

- identifies and classified named entities
- both machine learning model and manual rules can be used at any point in the pipeline
- supervised machine learning model
  - neural network classifier produces BILOU class+named entity type for every word
  - Viterbi decoding of all possible BILOU assignments
  - based on (Straková et al., 2013)
- uses MorphoDiTa to obtain POS tags
NameTag Features

- identifies and classified named entities
- both machine learning model and manual rules can be used at any point in the pipeline
- supervised machine learning model
  - neural network classifier produces BILOU class+named entity type for every word
  - Viterbi decoding of all possible BILOU assignments
  - based on (Straková et al., 2013)
- uses MorphoDiTa to obtain POS tags
NameTag Features

- identifies and classified named entities
- both machine learning model and manual rules can be used at any point in the pipeline
- supervised machine learning model
  - neural network classifier produces BILOU class+named entity type for every word
  - Viterbi decoding of all possible BILOU assignments
  - based on (Straková et al., 2013)
- uses MorphoDiTa to obtain POS tags
NameTag Features

- identifies and classified named entities
- both machine learning model and manual rules can be used at any point in the pipeline
- supervised machine learning model
  - neural network classifier produces BILOU class+named entity type for every word
  - Viterbi decoding of all possible BILOU assignments
    - based on (Straková et al., 2013)
- uses MorphoDiTa to obtain POS tags
NameTag Features

- identifies and classified named entities
- both machine learning model and manual rules can be used at any point in the pipeline
- supervised machine learning model
  - neural network classifier produces BILOU class+named entity type for every word
  - Viterbi decoding of all possible BILOU assignments
  - based on (Straková et al., 2013)
- uses MorphoDiTa to obtain POS tags
NameTag Features

- identifies and classified named entities
- both machine learning model and manual rules can be used at any point in the pipeline
- supervised machine learning model
  - neural network classifier produces BILOU class+named entity type for every word
  - Viterbi decoding of all possible BILOU assignments
  - based on (Straková et al., 2013)
- uses MorphoDiTa to obtain POS tags
NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
NameTag Features

NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
NameTag Features

NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
**NameTag Features**

**NameTag Implementation**

- Similarly as with MorphoDiTa
- Implemented in C++11
- Available under LGPL licence
- Library for using the models, binaries for creating them
  - Precompiled binaries+library for Linux/Windows/OS X
  - Library language bindings for
    - Java (precompiled in the package)
    - Perl (available on CPAN as a standalone package)
    - Python (available on PyPI as a standalone package)
  - Web service running on LINDAT/CLARIN
    - Weblicht integration coming soon
NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
  - library language bindings for
    - Java (precompiled in the package)
    - Perl (available on CPAN as a standalone package)
    - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
  - Weblicht integration coming soon
NameTag Features

NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
NameTag Features

NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
    - Perl (available on CPAN as a standalone package)
    - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
NameTag Features

NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
    - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
NameTag Features

NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
**NameTag Features**

### NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
  - Weblicht integration coming soon
**NameTag Features**

### NameTag Implementation

- similarly as with MorphoDiTa
- implemented in C++11
- available under LGPL licence
- library for using the models, binaries for creating them
- precompiled binaries+library for Linux/Windows/OS X
- library language bindings for
  - Java (precompiled in the package)
  - Perl (available on CPAN as a standalone package)
  - Python (available on PyPI as a standalone package)
- web service running on LINDAT/CLARIN
- Weblicht integration coming soon
Czech NER Model

- trained on Czech Named Entity Corpus
  - embedded name entities
  - two-level name entities hierarchy
    - 7 coarse classes, 46 fine-grained classes
  - \( \approx 35k \) named entities
- released under CC BY-SA-NC
- state-of-the-art results
- reasonable performance
  - 45k words per second
  - 6MB model
Czech NER Model

- trained on Czech Named Entity Corpus
  - embedded name entities
  - two-level name entities hierarchy
    - 7 coarse classes, 46 fine-grained classes
    - $\approx 35k$ named entities
  - released under CC BY-SA-NC
  - state-of-the-art results
  - reasonable performance
    - 45k words per second
    - 6MB model
Czech Model

Czech NER Model

- trained on Czech Named Entity Corpus
  - embedded name entities
  - two-level name entities hierarchy
    - 7 coarse classes, 46 fine-grained classes
    - \( \approx 35k \) named entities
- released under CC BY-SA-NC
- state-of-the-art results
- reasonable performance
  - 45k words per second
  - 6MB model
Czech Model

Czech NER Model
- trained on Czech Named Entity Corpus
  - embedded name entities
  - two-level name entities hierarchy
    - 7 coarse classes, 46 fine-grained classes
  - \( \approx 35k \) named entities
- released under CC BY-SA-NC
- state-of-the-art results
- reasonable performance
  - 45k words per second
  - 6MB model
Czech Model

Czech NER Model

- trained on Czech Named Entity Corpus
  - embedded name entities
  - two-level name entities hierarchy
    - 7 coarse classes, 46 fine-grained classes
  - \(\approx 35k\) named entities

- released under CC BY-SA-NC
  - state-of-the-art results
  - reasonable performance
    - 45k words per second
    - 6MB model
Czech Model

Czech NER Model

- trained on Czech Named Entity Corpus
  - embedded name entities
  - two-level name entities hierarchy
    - 7 coarse classes, 46 fine-grained classes
    - \( \approx 35k \) named entities
- released under CC BY-SA-NC
- state-of-the-art results
  - reasonable performance
    - 45k words per second
    - 6MB model
Czech Model

Czech NER Model
- trained on Czech Named Entity Corpus
  - embedded name entities
  - two-level name entities hierarchy
    - 7 coarse classes, 46 fine-grained classes
    - $\approx 35k$ named entities
- released under CC BY-SA-NC
- state-of-the-art results
- reasonable performance
  - 45k words per second
- 6MB model

MorphoDiTa and NameTag

Milan Straka

MorphoDiTa and NameTag
Czech NER Model

- trained on Czech Named Entity Corpus
  - embedded name entities
  - two-level name entities hierarchy
    - 7 coarse classes, 46 fine-grained classes
  - \( \approx 35k \) named entities
- released under CC BY-SA-NC
- state-of-the-art results
- reasonable performance
  - 45k words per second
  - 6MB model
English NER Model

- currently trained only on CoNLL-2003 data
  - four named entity classes only
- licence being discussed
  - hopefully CC BY-SA-NC
English Model

English NER Model
- currently trained only on CoNLL-2003 data
  - four named entity classes only
- licence being discussed
  - hopefully CC BY-SA-NC
More Languages

More English Models

- train recognizer on other datasets and hierarchies
  - MUC-6 and MUC-7
    - 7 named entity classes
  - ACE datasets
  - 3-classes hierarchy intersection of CoNLL-2003 and MUC

Additional Languages

- Arabic
  - ACE dataset
- German
  - CoNLL-2003 dataset

MorphoDiTa
Models
Plans
NameTag
Models
Plans
Milan Straka
MorphoDiTa and NameTag
More Languages

More English Models

- train recognizer on other datasets and hierarchies
  - MUC-6 and MUC-7
    - 7 named entity classes
  - ACE datasets
  - 3-classes hierarchy intersection of CoNLL-2003 and MUC

Additional Languages

- Arabic
  - ACE dataset
- German
  - CoNLL-2003 dataset
More Languages

More English Models

- train recognizer on other datasets and hierarchies
  - MUC-6 and MUC-7
    - 7 named entity classes
  - ACE datasets
    - 3-classes hierarchy intersection of CoNLL-2003 and MUC

Additional Languages

- Arabic
  - ACE dataset
- German
  - CoNLL-2003 dataset
More Languages

More English Models

- Train recognizer on other datasets and hierarchies
  - MUC-6 and MUC-7
    - 7 named entity classes
  - ACE datasets
  - 3-classes hierarchy intersection of CoNLL-2003 and MUC

Additional Languages

- Arabic
  - ACE dataset
- German
  - CoNLL-2003 dataset
More Languages

More English Models
- train recognizer on other datasets and hierarchies
  - MUC-6 and MUC-7
    - 7 named entity classes
  - ACE datasets
  - 3-classes hierarchy intersection of CoNLL-2003 and MUC

Additional Languages
- Arabic
  - ACE dataset
- German
  - CoNLL-2003 dataset
More Languages

More English Models

- train recognizer on other datasets and hierarchies
  - MUC-6 and MUC-7
    - 7 named entity classes
  - ACE datasets
  - 3-classes hierarchy intersection of CoNLL-2003 and MUC

Additional Languages

- Arabic
  - ACE dataset
- German
  - CoNLL-2003 dataset
Further Improvements

Recognizing Embedded Named Entities

- **recognize embedded named entities**
  - CNEC does contain embedded named entities, but NameTag tries to predict the outer ones (apart from the so-called containers)

More Semi-supervised Training

- **make use of available large corpora**
  - currently, only Brown clusters benefit from them
Further Improvements

Recognizing Embedded Named Entities

- recognize embedded named entities
  - CNEC does contain embedded named entities, but NameTag tries to predict the outer ones (apart from the so-called *containers*).

More Semi-supervised Training

- make use of available large corpora
  - currently, only Brown clusters benefit from them.
Further Improvements

Recognizing Embedded Named Entities
- recognize embedded named entities
  - CNEC does contain embedded named entities, but NameTag tries to predict the outer ones (apart from the so-called *containers*)

More Semi-supervised Training
- make use of available large corpora
  - currently, only Brown clusters benefit from them
Further Improvements

Recognizing Embedded Named Entities

- recognize embedded named entities
  - CNEC does contain embedded named entities, but NameTag tries to predict the outer ones (apart from the so-called containers)

More Semi-supervised Training

- make use of available large corpora
  - currently, only Brown clusters benefit from them
Questions?