Fluency ranking for Dutch

Daniël de Kok <d.j.a.de.kok@rug.nl>

Gertjan van Noord <g.j.m.van.noord@rug.nl>

Fluency ranking

- How do we select the most fluent sentence from a set of sentences?
  - *pas in 1842 werd er een geregelde kabinetsvergadering in het leven geroepen*
  - *er werd pas in 1842 een geregelde kabinetsvergadering in het leven geroepen*
  - […]
  - *'n geregelde kabinet-vergadering werd in het leven er pas in 1842 geroepen*
- Useful for various applications:
  - Sentence generation
  - Sentence compression
  - Sentence fusion

Overview

- Feature templates
- Maximum entropy modeling
- Feature selection methods
- Evaluation methodology
- Results

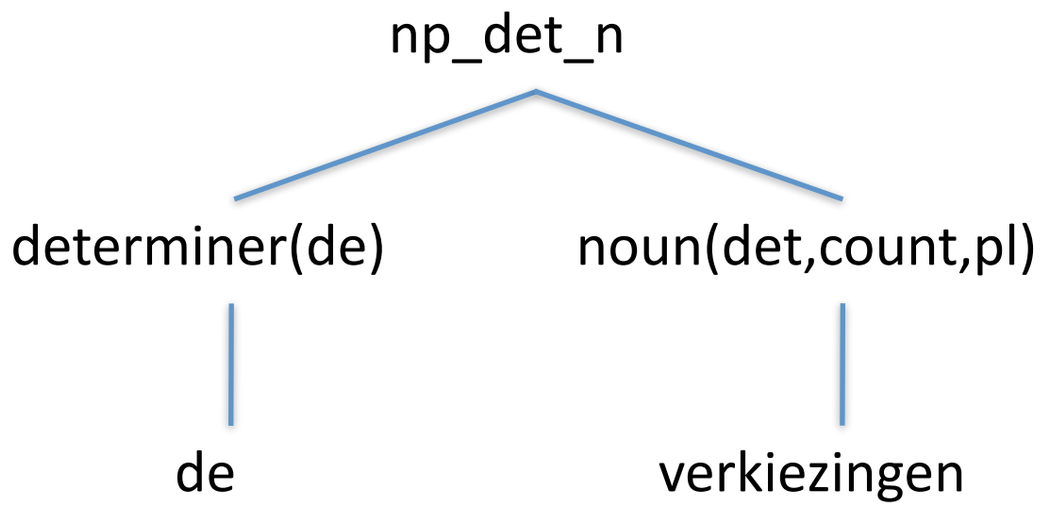
Feature engineering

- (Non-)fluency is characterized using features, which are automatically extracted using feature templates.
- Features can have arbitrary values, such as frequencies, scores, or (prior) probabilities.
- We can distinguish two classes of features in this task:
  - Output features (generated sentence/surface)
  - Construction features (derivation tree)
- Comparable to parse disambiguation:
  - Output features (dependency triples, semantics)
  - Construction features (derivation tree)

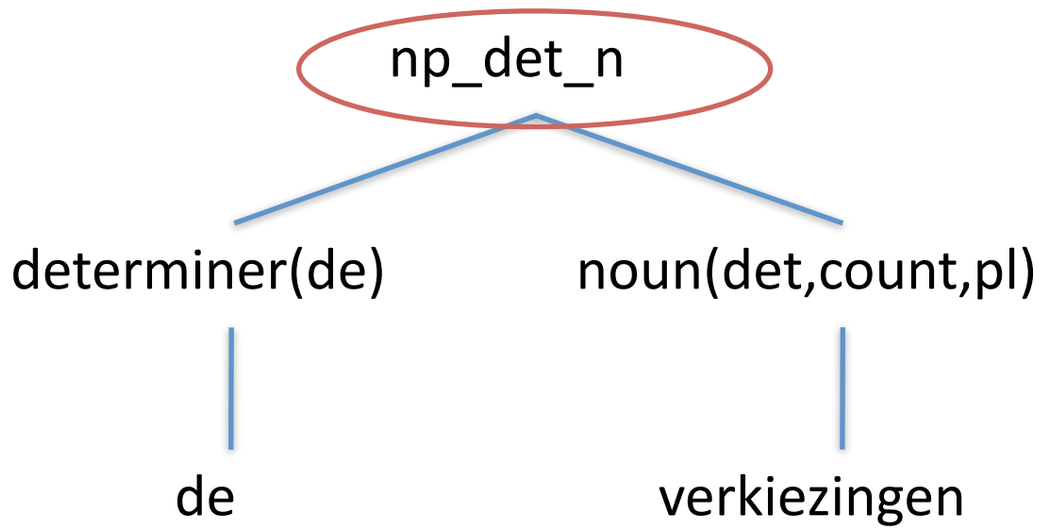
Output features

- *Barack Obama won de verkiezingen*
  - ngram_lm → 64.3 (- log $P(w_0..w_n)$)
- *proper_name verb determiner noun*
  - ngram_tag → 11.9 (- log $P(t_0..t_n)$)

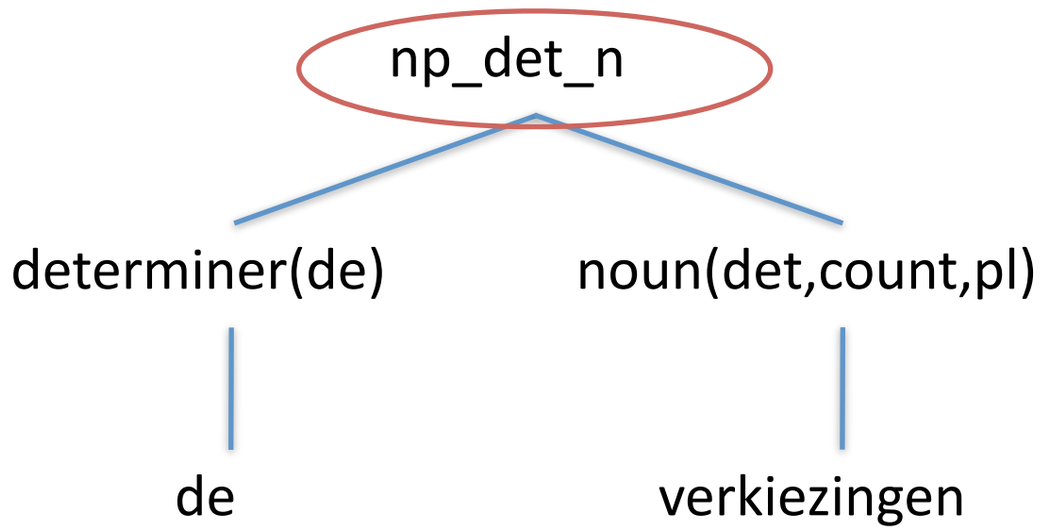
Construction features



Construction features

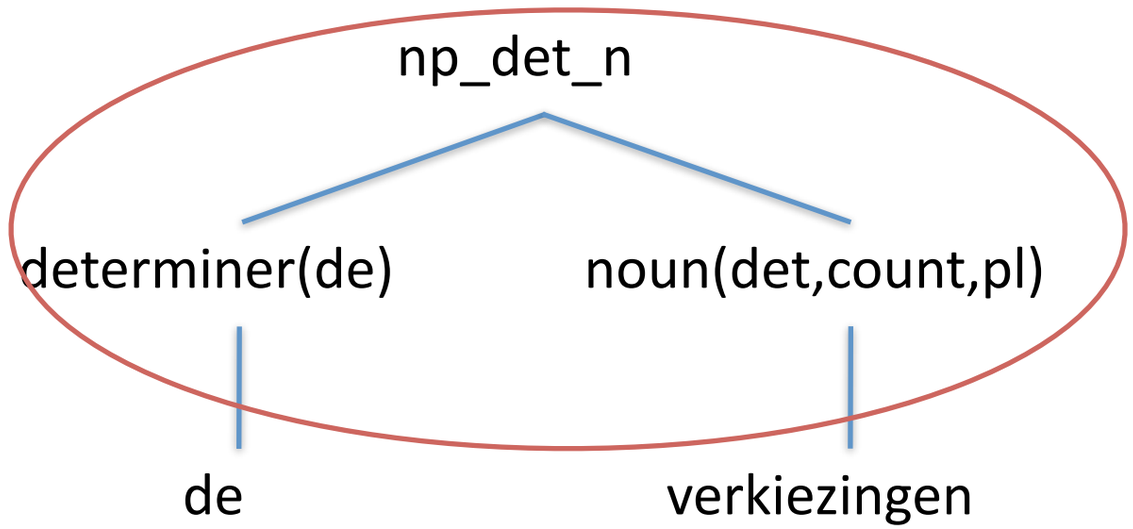


Construction features



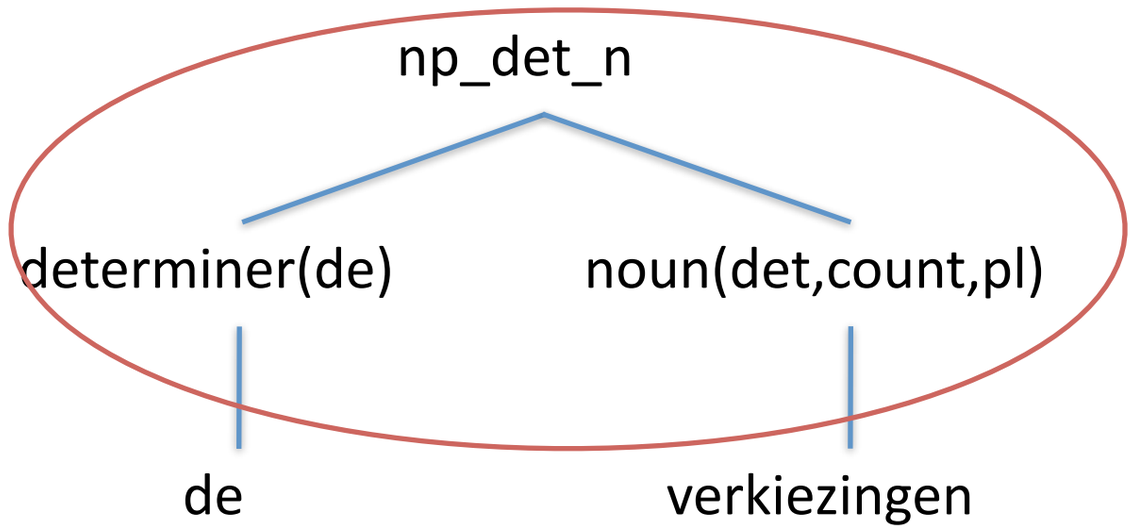
- r1(np_det_n) → 1

Construction features



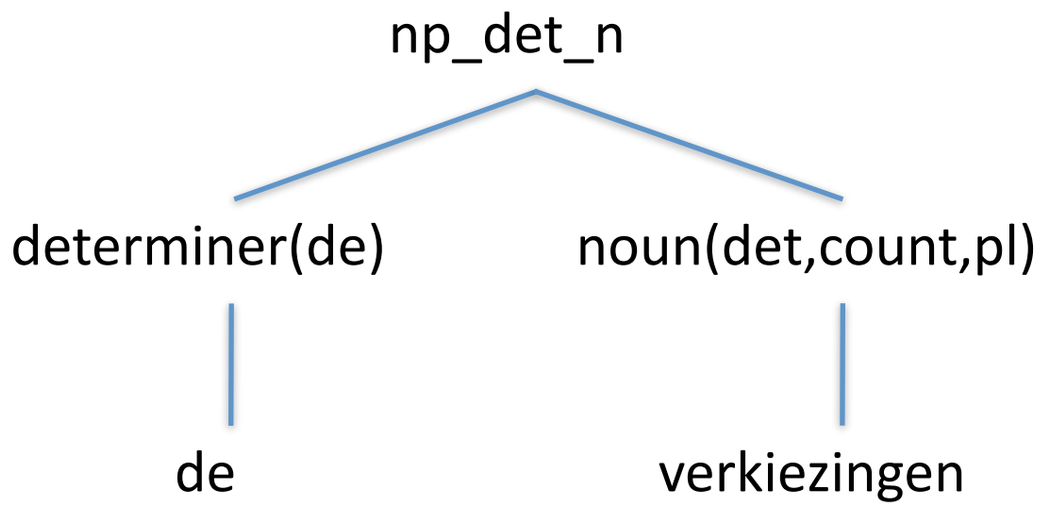
- r1(np_det_n) → 1

Construction features



- r1(np_det_n) → 1
- lds(np_det_n,[determiner(de),noun(det,count,pl),[]) → 1

Construction features



- r1(np_det_n) → 1
- lds(np_det_n,[determiner(de),noun(det,count,pl),[]) → 1
- r1(n_adj_n) → 0

Construction feature templates

- Parse disambiguation:
  - Topicalized/non-topicalized subjects and NPs
  - The use of long-distance/local dependencies
  - Orderings in the middle field
  - Identifiers of grammar rules used to build the derivation tree, and parent-daughter combinations
- Dependency orderings
- Local derivation subtrees (Velldal)
- Domination over words (Velldal):
  - Binned frequency of words nodes dominate over
  - Binned standard deviation of words nodes dominate over

Maximum entropy modeling

- We want to pick the model that has as few assumptions as possible, other than the feature expectation constraints (maximum entropy principle).
- We train a model that assigns a weight λ_i to each feature, so that the model maximizes entropy and the (empiric) expected feature values are equal to those of the model.
- We can now calculate the score of a realization (y) given a logical form (x).

$$score(y|x) = \frac{1}{Z_\lambda(x)} exp(\sum_i \lambda_i f_i(x, y))$$

Maximum entropy modeling

- We want to pick the model that has as few assumptions as possible, other than the feature expectation constraints (maximum entropy principle).
- We train a model that assigns a weight λ_i to each feature, so that the model maximizes entropy and the (empiric) expected feature values are equal to those of the model.
- We can now calculate the score of a realization (y) given a logical form (x).

$$score(y|x) = \frac{1}{Z(x)} exp(\sum_i \lambda_i f_i(x, y))$$

Maximum entropy modeling

- We want to pick the model that has as few assumptions as possible, other than the feature expectation constraints (maximum entropy principle).
- We train a model that assigns a weight λ_i to each feature, so that the model maximizes entropy and the (empiric) expected feature values are equal to those of the model.
- We can now calculate the score of a realization (y) given a logical form (x).

$$score(y|x) = \frac{1}{Z(x)} exp(\sum_i \lambda_i f_i(x, y))$$

Maximum entropy modeling

- We want to pick the model that has as few assumptions as possible, other than the feature expectation constraints (maximum entropy principle).
- We train a model that assigns a weight λ_i to each feature, so that the model maximizes entropy and the (empiric) expected feature values are equal to those of the model.
- We can now calculate the score of a realization (y) given a logical form (x).

$$score(y|x) = \sum_i \lambda_i f_i(x, y)$$

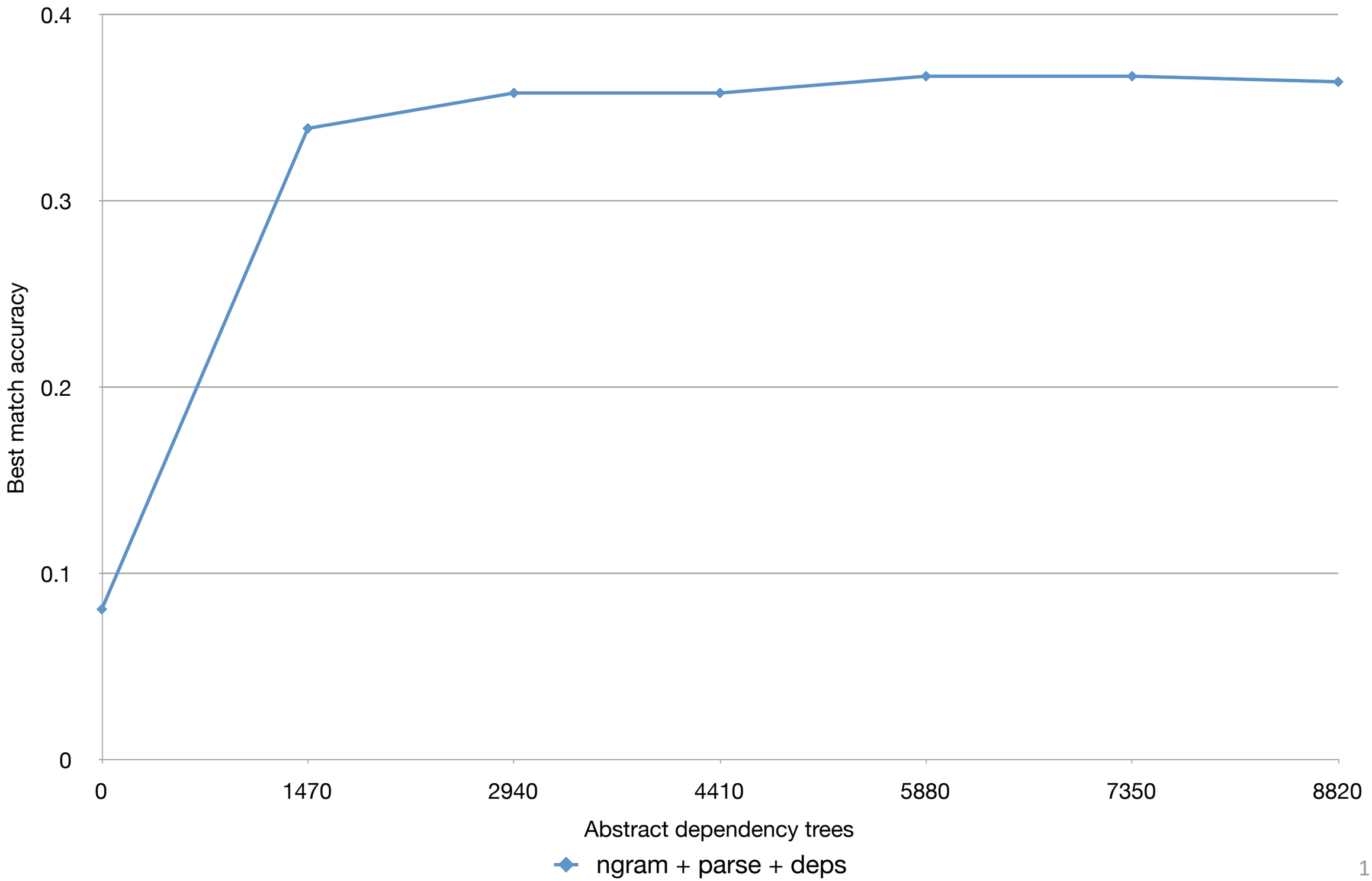
Feature selection

- Feature creation based on templates creates huge models (hundreds of thousands of features).
- Can we select only necessary features, making the model more compact?
- Experiments with two selection methods:
  - **Cut-off selection:** count how often a feature changes within a context. Features that change more often are more useful (Van Noord and Malouf, 2005).
  - **Maximum entropy selection:** start with a uniform model, and add features one by one, selecting the features that provide the highest gain (Berger, et al. 1996, modified for non-classification tasks).

Evaluation methodology

- Extract 5884 sentences from Wikipedia of 5 to 25 tokens.
- Parse this corpus, and create abstract dependency trees based on the highest-ranked parse.
- Use the Alpino chart generator to generate from the abstract dependency tree.
- Assign a score to each realization for a given input by comparing it to the sentence as it occurred in Wikipedia, using General Text Matching (Melamed, et al., 2003).
- Extract features and train models.
- Perform ten-fold cross validation. For each ranker, we calculate the best-match score (% of times the ranker preferred the best realization), for inputs with >= 5 realizations.

Learning curve

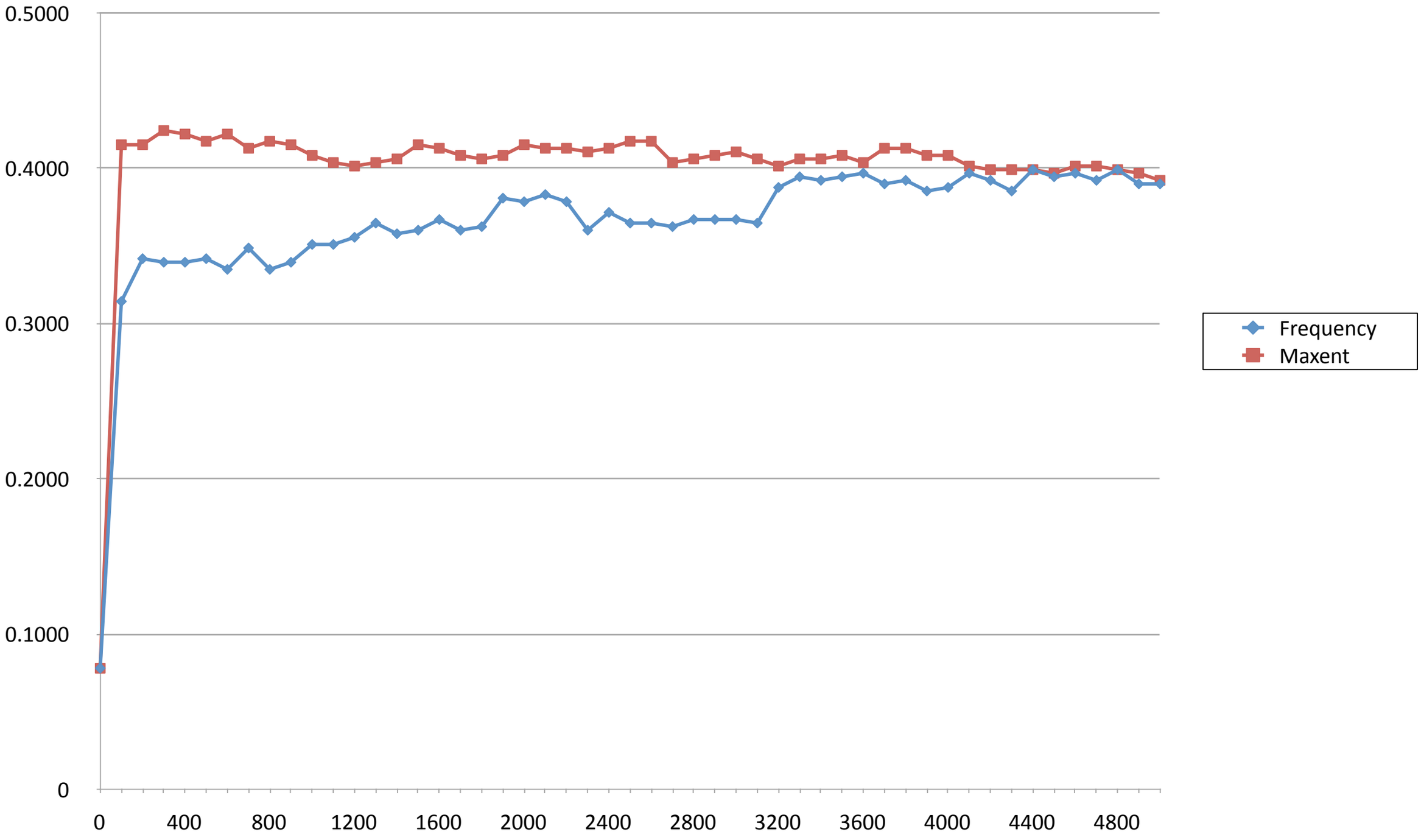


Fluency ranking results

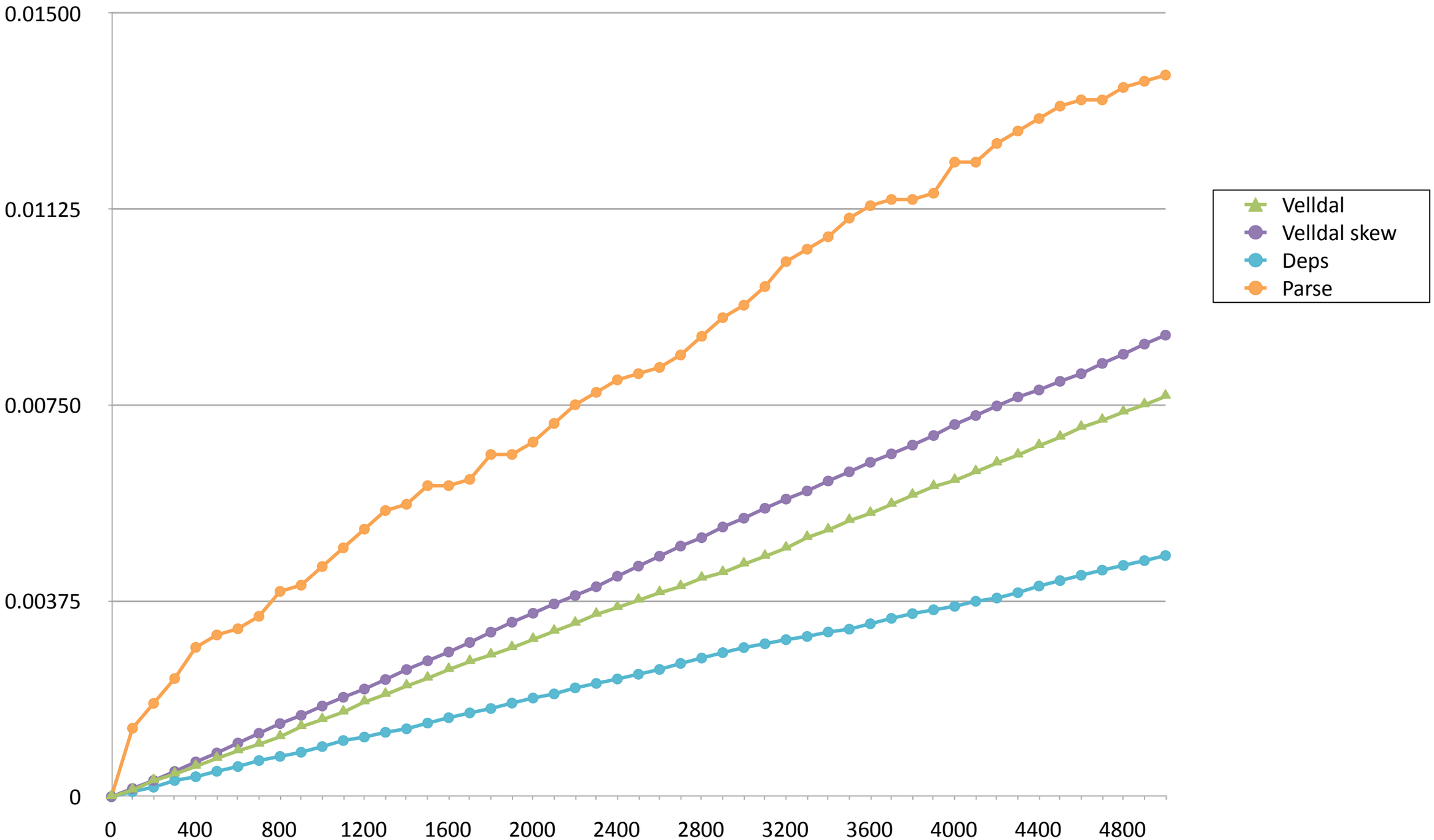
Features	Best match accuracy
random	0.0632
ngram	0.2901
ngram + parse	0.3655
ngram + parse + deps	0.3619
ngram + velldal	0.4064
ngram + velldal + deps	0.4051
ngram + velldal + parse	**0.4094**
all	0.4051

Manual annotation for 50 inputs (ngram + velldal + parse): 70%

Feature selection (1 fold)



Ratio of features selected



Typical problems

- Multiple fluent sentences:
  - Correct: *Luiz Inácio Lula da Silva vormt op dit moment een centrumlinkse regering*
  - Candidate: *op dit moment vormt Luiz Inácio Lula da Silva een centrum-linkse regering*
- Lexical variation caused by the productive lexicon:
  - Correct: *vatbaar voor het sharka-virus*
  - Candidate: *vatbaar voor het sharkavirus*
- Variation in punctuation:
  - Correct: *Serge Gainsbourg 1928-1991 , zanger , Frans*
  - Candidate: *Serge Gainsbourg 1928-1991 : zanger , Frans*

Typical problems (2)

- Incorrect parses
- Coordinations with identifiable preferred order:
  - Correct: *Britannicus had een proever en deze had van die hete drank geproefd*
  - Candidate: *deze had geproefd van die hete drank en Britannicus had een proever*
- Conjunctions without identifiable preferred order:
  - Correct: *meestal zijn het Lilo en Stitch die de experimenten vinden*
  - Candidate: *het zijn meestal Stitch en Lilo die de experimenten vinden*

Conclusions

- We can already do fairly good fluency ranking with very general features, such as n-gram probabilities and features that capture local derivation trees and word dominance.
- There is a fair amount of overlap between parse disambiguation and fluency ranking with respect to construction features.
- Some choices will be hard to make without semantics or context.
- Feature selection methods can reduce model sizes enormously.
- In our initial experiments, maximum entropy-based selection is more effective than frequency-based selection.

Thanks!

- Surf to http://www.let.rug.nl/vannoord/alp/Alpino/ to get the latest stable version of the Alpino chart generator and sentence ranker.