Predicting corpus example quality for lexicographic purposes by supervised machine learning

Nikola Ljubešić[1] Mario Peronja[1] Ivo-Pavao Jazbec[2]

[1]http://nlp.ffzg.hr
Department of Information and Communication Sciences
University of Zagreb

[2]Institute of Croatian Language and Linguistics

ENEL WG3 workshop
Vienna, 2015-02-12

Introduction

- good corpus examples are a very important part of every lexical resource
- frequently used approach – heuristics, GDEX, predefined variables that are weighted by a human, requires manual tweaking
- alternative – use supervised machine learning – learn to discriminate between good and bad corpus examples on manually annotated data
- difference – manual weighting vs. manual annotation
- ranking problem – want the good examples to be ranked high, bad examples low
  - the lexicographer examines only the N first candidates
  - we just include the first N candidates

Introduction

- good corpus examples are a very important part of every lexical resource
- frequently used approach – heuristics, GDEX, predefined variables that are weighted by a human, requires manual tweaking
- alternative – use supervised machine learning – learn to discriminate between good and bad corpus examples on manually annotated data
- difference – manual weighting vs. manual annotation
- ranking problem – want the good examples to be ranked high, bad examples low
  - the lexicographer examines only the N first candidates
  - we just include the first N candidates

Introduction

- good corpus examples are a very important part of every lexical resource
- frequently used approach – heuristics, GDEX, predefined variables that are weighted by a human, requires manual tweaking
- alternative – use supervised machine learning – learn to discriminate between good and bad corpus examples on manually annotated data
- difference – manual weighting vs. manual annotation
- ranking problem – want the good examples to be ranked high, bad examples low
  - the lexicographer examines only the N first candidates
  - we just include the first N candidates

Introduction

- good corpus examples are a very important part of every lexical resource
- frequently used approach – heuristics, GDEX, predefined variables that are weighted by a human, requires manual tweaking
- alternative – use supervised machine learning – learn to discriminate between good and bad corpus examples on manually annotated data
- difference – manual weighting vs. manual annotation
- ranking problem – want the good examples to be ranked high, bad examples low
  - the lexicographer examines only the N first candidates
  - we just include the first N candidates

Introduction

- good corpus examples are a very important part of every lexical resource
- frequently used approach – heuristics, GDEX, predefined variables that are weighted by a human, requires manual tweaking
- alternative – use supervised machine learning – learn to discriminate between good and bad corpus examples on manually annotated data
- difference – manual weighting vs. manual annotation
- ranking problem – want the good examples to be ranked high, bad examples low
    - the lexicographer examines only the N first candidates
    - we just include the first N candidates

The dataset

- 4 lexemes, one per each PoS
- 16 collocations, 4 per lexeme
- 1094 example sentences from the hrWaC corpus
- each example annotated by a 4-class schema:
  - 1 – very bad 14%
  - 2 – bad 41.7%
  - 3 – good 33.3%
  - 4 – very good 11.1%
- double annotation of 100 sentences, observed agreement 44%, on two classes 66%

	1	2	3	4
1	9	11	7	2
2	9	25	14	6
3	1	4	9	2
4	0	0	0	1

The dataset

- 4 lexemes, one per each PoS
- 16 collocations, 4 per lexeme
- 1094 example sentences from the hrWaC corpus
- each example annotated by a 4-class schema:
  - 1 – very bad 14%
  - 2 – bad 41.7%
  - 3 – good 33.3%
  - 4 – very good 11.1%
- double annotation of 100 sentences, observed agreement 44%, on two classes 66%

	1	2	3	4
1	9	11	7	2
2	9	25	14	6
3	1	4	9	2
4	0	0	0	1

The dataset

- 4 lexemes, one per each PoS
- 16 collocations, 4 per lexeme
- 1094 example sentences from the hrWaC corpus
- each example annotated by a 4-class schema:
  - 1 – very bad 14%
  - 2 – bad 41.7%
  - 3 – good 33.3%
  - 4 – very good 11.1%
- double annotation of 100 sentences, observed agreement 44%, on two classes 66%

	1	2	3	4
1	9	11	7	2
2	9	25	14	6
3	1	4	9	2
4	0	0	0	1

Experimental setup

- define 23 explanatory variables / features
  - string-based
  - corpus-based
  - linguistic
- inspect the strength of each variable
  - univariate analysis – ANOVA on each variable grouped by the 2-class response
  - feature elimination – remove the variable from the set of all variables and measure the loss
- our response variable (quality of the example) is an ordinal value – use regression for prediction (RandomForestRegressor from sklearn)
- output as a ranking task – sort examples of each collocate by the response variable
- evaluation – precision on first N results (P@N) for each collocate

Experimental setup

- define 23 explanatory variables / features
  - string-based
  - corpus-based
  - linguistic
- inspect the strength of each variable
  - univariate analysis – ANOVA on each variable grouped by the 2-class response
  - feature elimination – remove the variable from the set of all variables and measure the loss
- our response variable (quality of the example) is an ordinal value – use regression for prediction (RandomForestRegressor from sklearn)
- output as a ranking task – sort examples of each collocate by the response variable
- evaluation – precision on first N results (P@N) for each collocate

Experimental setup

- define 23 explanatory variables / features
  - string-based
  - corpus-based
  - linguistic
- inspect the strength of each variable
  - univariate analysis – ANOVA on each variable grouped by the 2-class response
  - feature elimination – remove the variable from the set of all variables and measure the loss
- our response variable (quality of the example) is an ordinal value – use regression for prediction (RandomForestRegressor from sklearn)
- output as a ranking task – sort examples of each collocate by the response variable
- evaluation – precision on first N results (P@N) for each collocate

Experimental setup

- define 23 explanatory variables / features
  - string-based
  - corpus-based
  - linguistic
- inspect the strength of each variable
  - univariate analysis – ANOVA on each variable grouped by the 2-class response
  - feature elimination – remove the variable from the set of all variables and measure the loss
- our response variable (quality of the example) is an ordinal value – use regression for prediction (RandomForestRegressor from sklearn)
- output as a ranking task – sort examples of each collocate by the response variable
- evaluation – precision on first N results (P@N) for each collocate

Features

1. sent_len – length of the sentence
2. avg_len – average token length
3. gte10_perc – percentage of tokens longer or equal to 10 characters
4. lt3_perc – percentage of tokens shorter than 3 characters
5. alphanum_perc – percentage of tokens being alphanumeric
6. alphanumpunc_perc – percentage of tokens being alphanumeric or standard punctuations
7. startswithucase – whether the sentence starts with an uppercase letter
8. endswithpunc – whether the sentence ends with a punctuation
9. diac_perc – percentage of tokens containing diacritics
10. lcase_perc – percentage of lowercased tokens
11. ucase_perc – percentage of uppercased tokens
12. tcase_perc – percentage of titlecased tokens
13. headpos_perc – relative position of the start of collocation

Features

14. mf1k_perc – percentage of tokens in the 1k most frequent corpus tokens
15. mf10k_perc – percentage of tokens in the 10k most frequent corpus tokens
16. mf100k_perc – percentage of tokens in the 100k most frequent corpus tokens

17. pron_perc – percentage of pronoun tokens
18. pn_perc – percentage of proper noun tokens
19. num_perc – percentage of numeral tokens
20. sub_num – number of subordinating conjunctions
21. co_num – number of coordinating conjunctions
22. subco_num – number of conjunctions
23. syntcomplex – syntactic complexity as the average length of the dependency arcs

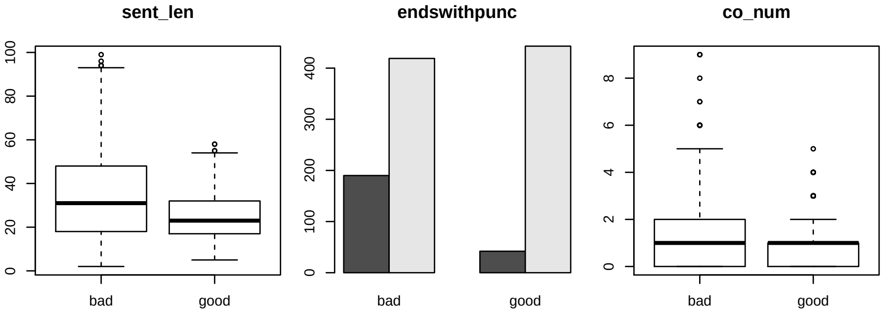
Feature strength

	univariate	elimination
sent_len	7.0e-18	-0.0207
avg_len	5.7e-05	-0.0029
gte10_perc	0.1087	0.0000
lt3_perc	9.9e-05	0.0001
alphanum_perc	4.1e-09	-0.0086
alphanumpunc_perc	5.1e-05	-0.0012
startswithucase	3.5e-04	0.0005
endswithpunc	2.7e-20	-0.0459
diac_perc	0.0064	-0.0002
lcase_perc	0.0063	0.0015
ucase_perc	0.0045	-0.0039
tcase_perc	0.0760	-0.0040
headpos_perc	0.0007	-0.0082

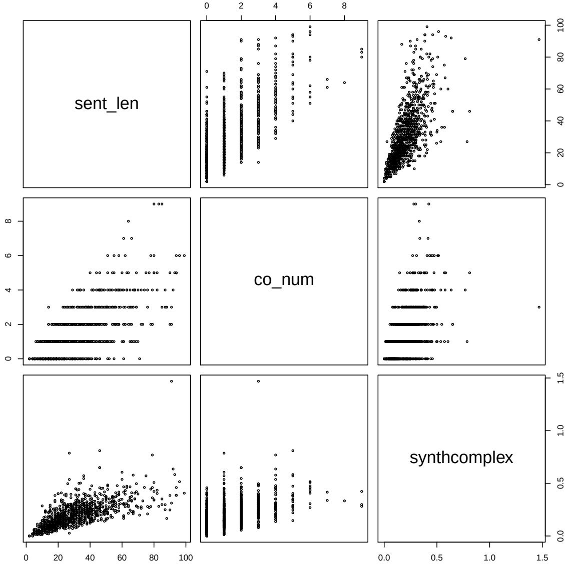
Feature strength

	univariate	elimination
mf1k_perc	0.0687	0.0034
mf10k_perc	0.0008	0.0009
mf100k_perc	1.7e-05	-0.0067
pron_perc	0.4039	-0.0016
pn_perc	0.0018	-0.0017
num_perc	0.0037	0.0019
sub_num	5.7e-08	0.0031
co_num	7.4e-16	-0.0018
subco_num	1.3e-15	-0.0021
syntcomplex	8.2e-12	-0.0045

Feature distribution



sent_len, co_num, syntcomplex?



Results

- evaluate the regression results as a ranking task
- produce ranked results for each collocate – realistic setting
- calculate precision of first N results ($P@N$) with $N = 3, 5, 10$
- baseline – random order of sentences
- ceiling – sentences ordered by human annotation
- regressor_all – all 23 features
- regressor_string – only string features (no outer knowledge)
- regressor_string_langind – only language independent string features (without diac_perc)

	P@10	P@5	P@3
baseline	0.489	0.495	0.496
ceiling	0.988	1.0	1.0
regressor_all	**0.819**	**0.900**	**0.940**
regressor_string	0.794	0.888	0.922
regressor_string_langind	0.783	0.850	0.880

Results

- evaluate the regression results as a ranking task
- produce ranked results for each collocate – realistic setting
- calculate precision of first N results ($P@N$) with $N = 3, 5, 10$
- baseline – random order of sentences
- ceiling – sentences ordered by human annotation
- regressor_all – all 23 features
- regressor_string – only string features (no outer knowledge)
- regressor_string_langind – only language independent string features (without diac_perc)

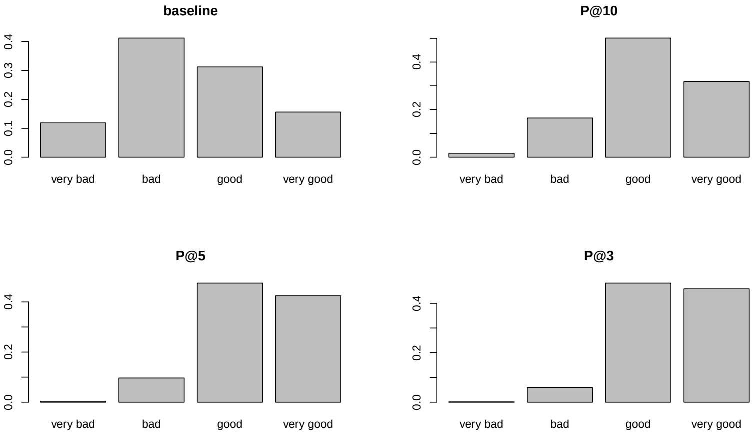
	P@10	P@5	P@3
baseline	0.489	0.495	0.496
ceiling	0.988	1.0	1.0
regressor_all	**0.819**	**0.900**	**0.940**
regressor_string	0.794	0.888	0.922
regressor_string_langind	0.783	0.850	0.880

Results

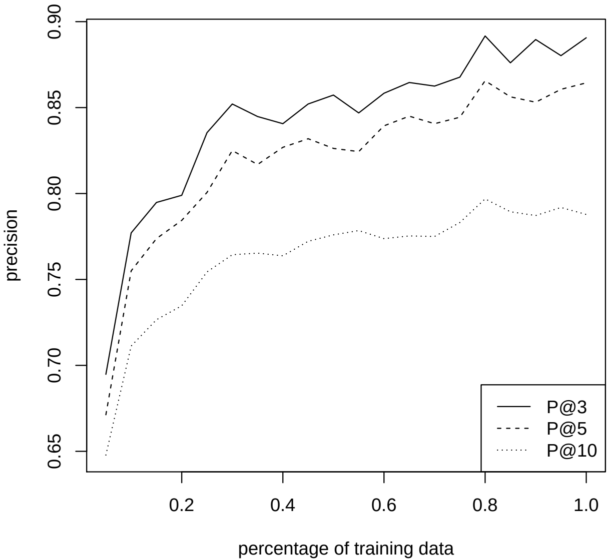
- evaluate the regression results as a ranking task
- produce ranked results for each collocate – realistic setting
- calculate precision of first N results ($P@N$) with $N = 3, 5, 10$
- baseline – random order of sentences
- ceiling – sentences ordered by human annotation
- regressor_all – all 23 features
- regressor_string – only string features (no outer knowledge)
- regressor_string_langind – only language independent string features (without diac_perc)

	P@10	P@5	P@3
baseline	0.489	0.495	0.496
ceiling	0.988	1.0	1.0
regressor_all	**0.819**	**0.900**	**0.940**
regressor_string	0.794	0.888	0.922
regressor_string_langind	0.783	0.850	0.880

Distribution of the results



Learning curve



Conclusion

- supervised learning approach for predicting corpus example quality – train a regression model, use it for ranking
- 23 variables from three different categories
- best prediction (94%) when using all variables
- loss of 2% when using only string variables (no extra knowledge necessary), language independent setting 6% loss
- language independence should be tested on multilingual data
  - train and evaluate on L2 data
  - evaluate the L1 model on L2 data
- what does pay off more? – manual weighting vs. manual annotation
- additional features? such as "example prototypicality"?
  - compare each example to a bag-of-words model of all examples
  - the prototypical usages should be most similar to the model

Conclusion

- supervised learning approach for predicting corpus example quality – train a regression model, use it for ranking
- 23 variables from three different categories
- best prediction (94%) when using all variables
- loss of 2% when using only string variables (no extra knowledge necessary), language independent setting 6% loss
- language independence should be tested on multilingual data
  - train and evaluate on L2 data
  - evaluate the L1 model on L2 data
- what does pay off more? – manual weighting vs. manual annotation
- additional features? such as "example prototypicality"?
  - compare each example to a bag-of-words model of all examples
  - the prototypical usages should be most similar to the model

Conclusion

- supervised learning approach for predicting corpus example quality – train a regression model, use it for ranking
- 23 variables from three different categories
- best prediction (94%) when using all variables
- loss of 2% when using only string variables (no extra knowledge necessary), language independent setting 6% loss
- language independence should be tested on multilingual data
  - train and evaluate on L2 data
  - evaluate the L1 model on L2 data
- what does pay off more? – manual weighting vs. manual annotation
- additional features? such as "example prototypicality"?
  - compare each example to a bag-of-words model of all examples
  - the prototypical usages should be most similar to the model

Conclusion

- supervised learning approach for predicting corpus example quality – train a regression model, use it for ranking
- 23 variables from three different categories
- best prediction (94%) when using all variables
- loss of 2% when using only string variables (no extra knowledge necessary), language independent setting 6% loss
- language independence should be tested on multilingual data
  - train and evaluate on L2 data
  - evaluate the L1 model on L2 data
- what does pay off more? – manual weighting vs. manual annotation
- additional features? such as "example prototypicality"?
  - compare each example to a bag-of-words model of all examples
  - the prototypical usages should be most similar to the model

Conclusion

- supervised learning approach for predicting corpus example quality – train a regression model, use it for ranking
- 23 variables from three different categories
- best prediction (94%) when using all variables
- loss of 2% when using only string variables (no extra knowledge necessary), language independent setting 6% loss
- language independence should be tested on multilingual data
  - train and evaluate on L2 data
  - evaluate the L1 model on L2 data
- what does pay off more? – manual weighting vs. manual annotation
- additional features? such as "example prototypicality"?
  - compare each example to a bag-of-words model of all examples
  - the prototypical usages should be most similar to the model

Conclusion

- supervised learning approach for predicting corpus example quality – train a regression model, use it for ranking
- 23 variables from three different categories
- best prediction (94%) when using all variables
- loss of 2% when using only string variables (no extra knowledge necessary), language independent setting 6% loss
- language independence should be tested on multilingual data
  - train and evaluate on L2 data
  - evaluate the L1 model on L2 data
- what does pay off more? – manual weighting vs. manual annotation
- additional features? such as "example prototypicality"?
  - compare each example to a bag-of-words model of all examples
  - the prototypical usages should be most similar to the model

Predicting corpus example quality for lexicographic purposes by supervised machine learning

Nikola Ljubešić[1] Mario Peronja[1] Ivo-Pavao Jazbec[2]

[1]http://nlp.ffzg.hr
Department of Information and Communication Sciences
University of Zagreb

[2]Institute of Croatian Language and Linguistics

ENEL WG3 workshop
Vienna, 2015-02-12