Definition Extraction with Balanced Random Forests

Łukasz Kobyliński[1] Adam Przepiórkowski[2,3]

[1]Institute of Computer Science, Warsaw University of Technology,
L.Kobylinski@elka.pw.edu.pl

[2]Institute of Computer Science, Polish Academy of Sciences,
adamp@ipipan.waw.pl

[3]Institute of Informatics, University of Warsaw

GoTAL 2008 — 25 August 2008

Introduction

Context: *Language Technology for eLearning* (LT4eL):

- an FP6 STReP European Project ended 31 May 2008,
- `http://www.lt4el.eu/`,
- **project aim**: develop multilingual language technology tools for improving the retrieval of learning material.

Task:

- given an instructive text,
- find passages in this text which seem to define technical terms;
- such passages are presented to text creator or maintainer,
- who may:
  - reject them,
  - include them in the glossary (after minor editing).

Introduction

Context: *Language Technology for eLearning* (LT4eL):

- an FP6 STReP European Project ended 31 May 2008,
- `http://www.lt4el.eu/`,
- **project aim**: develop multilingual language technology tools for improving the retrieval of learning material.

Task:

- given an instructive text,
- find passages in this text which seem to define technical terms;
- such passages are presented to text creator or maintainer,
- who may:
  - reject them,
  - include them in the glossary (after minor editing).

Introduction

Context: *Language Technology for eLearning* (LT4eL):

- an FP6 STReP European Project ended 31 May 2008,
- `http://www.lt4el.eu/`,
- **project aim**: develop multilingual language technology tools for improving the retrieval of learning material.

Task:

- given an instructive text,
- find passages in this text which seem to define technical terms;
- such passages are presented to text creator or maintainer,
- who may:
  - reject them,
  - include them in the glossary (after minor editing).

Introduction (contd.)

Empirical background: a collection of various e-learning materials in Polish:

	#
tokens	300 636
sentences	10 830
definitional sentences	546

Evaluation:

- approximate definitions by definitional sentences,
- precision and recall at sentence level,
- recall more important than precision, so summarised by F_2 (formula for F_2 given later),
- 10-fold cross-validation (in case of ML methods).

Introduction (contd.)

Empirical background: a collection of various e-learning materials in Polish:

	#
tokens	300 636
sentences	10 830
definitional sentences	546

Evaluation:

- approximate definitions by definitional sentences,
- precision and recall at sentence level,
- recall more important than precision, so summarised by F_2 (formula for F_2 given later),
- 10-fold cross-validation (in case of ML methods).

Introduction (contd.)

Difficult:

- rather small empirical basis: $< 11K$ sentences, incl. < 550 definitional,
- very ill-defined task: Cohen's $\kappa = 0.31$ (but $\kappa_{\max} = 0.425$; cf. Przepiórkowski *et al.* 2007),
- very imbalanced: the ratio of definitions to non-definitions $\approx 1 : 20$.

Looks like a task perhaps best approached symbolically (rather than statistically)...

Introduction (contd.)

Difficult:

- rather small empirical basis: $< 11K$ sentences, incl. < 550 definitional,
- very ill-defined task: Cohen's $\kappa = 0.31$ (but $\kappa_{\max} = 0.425$; cf. Przepiórkowski *et al.* 2007),
- very imbalanced: the ratio of definitions to non-definitions $\approx 1 : 20$.

Looks like a task perhaps best approached symbolically (rather than statistically)...

Definition extraction grammars

Przepiórkowski *et al.* 2007:

- a cascade of regular grammars
- based on the recognition of copula expressions and other indicators of definitions,
- included subgrammars for NPs, PPs, etc.,
- implemented using `lxtransduce` (Tobin, 2005), a component of LTXML2 (University of Edinburgh),
- around 2 weeks of intensive work:
  - developed on the basis of a development subcorpus (5 218 sentences),
  - tuning on the basis of a held-out subcorpus (2 263),
- evaluation on the basis of unseen testing data (3 349).

Results:

	P	R	$F_{\alpha=1}$	$F_{\alpha=2}$	$F_{\alpha=5}$
GR′	18.7%	59.3%	28.4	**34.4**	43.6

where $F_{\alpha} = \dfrac{(1+\alpha)\cdot(P\cdot R)}{(\alpha\cdot P + R)}$

Definition extraction grammars

Przepiórkowski *et al.* 2007:

- a cascade of regular grammars
- based on the recognition of copula expressions and other indicators of definitions,
- included subgrammars for NPs, PPs, etc.,
- implemented using `lxtransduce` (Tobin, 2005), a component of LTXML2 (University of Edinburgh),
- around 2 weeks of intensive work:
  - developed on the basis of a development subcorpus (5 218 sentences),
  - tuning on the basis of a held-out subcorpus (2 263),
- evaluation on the basis of unseen testing data (3 349).

Results:

	P	R	$F_{\alpha=1}$	$F_{\alpha=2}$	$F_{\alpha=5}$
GR′	18.7%	59.3%	28.4	**34.4**	43.6

where $F_\alpha = \dfrac{(1+\alpha)\cdot(P\cdot R)}{(\alpha\cdot P + R)}$

Definition extraction grammars

Przepiórkowski *et al.* 2007:

- a cascade of regular grammars
- based on the recognition of copula expressions and other indicators of definitions,
- included subgrammars for NPs, PPs, etc.,
- implemented using `lxtransduce` (Tobin, 2005), a component of LTXML2 (University of Edinburgh),
- around 2 weeks of intensive work:
  - developed on the basis of a development subcorpus (5 218 sentences),
  - tuning on the basis of a held-out subcorpus (2 263),
- evaluation on the basis of unseen testing data (3 349).

Results:

	P	R	$F_{\alpha=1}$	$F_{\alpha=2}$	$F_{\alpha=5}$
GR′	18.7%	59.3%	28.4	**34.4**	43.6

where $F_\alpha = \dfrac{(1+\alpha)\cdot(P\cdot R)}{(\alpha\cdot P + R)}$

Definition extraction grammars

Przepiórkowski *et al.* 2007:

- a cascade of regular grammars
- based on the recognition of copula expressions and other indicators of definitions,
- included subgrammars for NPs, PPs, etc.,
- implemented using `lxtransduce` (Tobin, 2005), a component of LTXML2 (University of Edinburgh),
- around 2 weeks of intensive work:
  - developed on the basis of a development subcorpus (5 218 sentences),
  - tuning on the basis of a held-out subcorpus (2 263),
- evaluation on the basis of unseen testing data (3 349).

Results:

	P	R	$F_{\alpha=1}$	$F_{\alpha=2}$	$F_{\alpha=5}$
GR′	18.7%	59.3%	28.4	**34.4**	43.6

where $F_\alpha = \dfrac{(1+\alpha)\cdot(P\cdot R)}{(\alpha\cdot P + R)}$

Definition extraction grammars

Przepiórkowski *et al.* 2007:

- a cascade of regular grammars
- based on the recognition of copula expressions and other indicators of definitions,
- included subgrammars for NPs, PPs, etc.,
- implemented using `lxtransduce` (Tobin, 2005), a component of LTXML2 (University of Edinburgh),
- around 2 weeks of intensive work:
  - developed on the basis of a development subcorpus (5 218 sentences),
  - tuning on the basis of a held-out subcorpus (2 263),
- evaluation on the basis of unseen testing data (3 349).

Results:

	P	R	$F_{\alpha=1}$	$F_{\alpha=2}$	$F_{\alpha=5}$
GR′	18.7%	59.3%	28.4	**34.4**	43.6

where $F_\alpha = \dfrac{(1+\alpha)\cdot(\mathrm{P}\cdot\mathrm{R})}{(\alpha\cdot\mathrm{P}+\mathrm{R})}$

Machine Learning?

Fact: $F_{\alpha=2} = \mathbf{34.4}$ is pathetic.

Maybe **Machine Learning** (ML) approaches more suitable after all?

Degórski *et al.* 2008b:

- use a simple, linguistically lean grammar to select definition candidates (small precision, very high recall; $F_{\alpha=2} = 25.5$),
- apply ML methods to the result:
  - homogeneous ensembles of classifiers of the same type, for various types of ML methods tested (Decision Trees, Naïve Bayes, SVM, AdaBoost, lazy learning),
  - best results for ID3 (better than for C4.5): $F_{\alpha=2} = \mathbf{37.95}$,
- the use of simple grammar crucial.

Machine Learning?

Fact: $F_{\alpha=2} = \mathbf{34.4}$ is pathetic.

Maybe **Machine Learning** (ML) approaches more suitable after all?

Degórski *et al.* 2008b:

- use a simple, linguistically lean grammar to select definition candidates (small precision, very high recall; $F_{\alpha=2} = 25.5$),
- apply ML methods to the result:
  - homogeneous ensembles of classifiers of the same type, for various types of ML methods tested (Decision Trees, Naïve Bayes, SVM, AdaBoost, lazy learning),
  - best results for ID3 (better than for C4.5): $F_{\alpha=2} = \mathbf{37.95}$,
- the use of simple grammar crucial.

Machine Learning?

Fact: $F_{\alpha=2} = \mathbf{34.4}$ is pathetic.

Maybe **Machine Learning** (ML) approaches more suitable after all?

Degórski *et al.* 2008b:

- use a simple, linguistically lean grammar to select definition candidates (small precision, very high recall; $F_{\alpha=2} = 25.5$),
- apply ML methods to the result:
  - homogeneous ensembles of classifiers of the same type, for various types of ML methods tested (Decision Trees, Naïve Bayes, SVM, AdaBoost, lazy learning),
  - best results for ID3 (better than for C4.5): $F_{\alpha=2} = \mathbf{37.95}$,
- the use of simple grammar crucial.

Machine Learning approaches (contd.)

Przepiórkowski *et al.* 2008:

- use the same simple grammar, the full grammar, and various homogeneous ensembles of classifiers,
- combine them linearly into a single heterogeneous classifier;
- the best result: $F_{\alpha=2} = \mathbf{38.9}$ (compare to the previous $F_{\alpha=2} = \mathbf{34.4}$ and $F_{\alpha=2} = \mathbf{37.95}$);
- again, the use of the grammars crucial.

Here:

- use a novel ML technique (balanced random forests; BRFs),
- no grammars at all!,
- the best result: $F_{\alpha=2} = \mathbf{39.6}$,
- to some extent improvement due to a more careful attribute selection procedure.

Machine Learning approaches (contd.)

Przepiórkowski *et al.* 2008:

- use the same simple grammar, the full grammar, and various homogeneous ensembles of classifiers,
- combine them linearly into a single heterogeneous classifier;
- the best result: $F_{\alpha=2} = \mathbf{38.9}$ (compare to the previous $F_{\alpha=2} = \mathbf{34.4}$ and $F_{\alpha=2} = \mathbf{37.95}$);
- again, the use of the grammars crucial.

Here:

- use a novel ML technique (balanced random forests; BRFs),
- no grammars at all!,
- the best result: $F_{\alpha=2} = \mathbf{39.6}$,
- to some extent improvement due to a more careful attribute selection procedure.

Machine Learning approaches (contd.)

Przepiórkowski *et al.* 2008:

- use the same simple grammar, the full grammar, and various homogeneous ensembles of classifiers,
- combine them linearly into a single heterogeneous classifier;
- the best result: $F_{\alpha=2} = \mathbf{38.9}$ (compare to the previous $F_{\alpha=2} = \mathbf{34.4}$ and $F_{\alpha=2} = \mathbf{37.95}$);
- again, the use of the grammars crucial.

Here:

- use a novel ML technique (balanced random forests; BRFs),
- no grammars at all!,
- the best result: $F_{\alpha=2} = \mathbf{39.6}$,
- to some extent improvement due to a more careful attribute selection procedure.

Random Forests

Random Forest (Breiman, 2001):

- an ensemble of decision trees (ensemble, i.e., final decisions reached by voting),
- unpruned;
- random (1):
  - at each node of a tree
  - a subset of attributes is randomly selected
  - from which the best attribute to further grow the tree is calculated;
- random (2; bagging, i.e., bootstrap aggregating):
  - for each tree,
  - bootstrap (randomly select with replacing) a multiset (bag) of training examples,
  - of the size of the original training set.

Random Forests

Random Forest (Breiman, 2001):

- an ensemble of decision trees (ensemble, i.e., final decisions reached by voting),
- unpruned;
- random (1):
  - at each node of a tree
  - a subset of attributes is randomly selected
  - from which the best attribute to further grow the tree is calculated;
- random (2; bagging, i.e., bootstrap aggregating):
  - for each tree,
  - bootstrap (randomly select with replacing) a multiset (bag) of training examples,
  - of the size of the original training set.

Random Forests

Random Forest (Breiman, 2001):

- an ensemble of decision trees (ensemble, i.e., final decisions reached by voting),
- unpruned;
- random (1):
  - at each node of a tree
  - a subset of attributes is randomly selected
  - from which the best attribute to further grow the tree is calculated;
- random (2; bagging, i.e., bootstrap aggregating):
  - for each tree,
  - bootstrap (randomly select with replacing) a multiset (bag) of training examples,
  - of the size of the original training set.

Random Forests

Random Forest (Breiman, 2001):

- an ensemble of decision trees (ensemble, i.e., final decisions reached by voting),
- unpruned;
- random (1):
  - at each node of a tree
  - a subset of attributes is randomly selected
  - from which the best attribute to further grow the tree is calculated;
- random (2; bagging, i.e., bootstrap aggregating):
  - for each tree,
  - bootstrap (randomly select with replacing) a multiset (bag) of training examples,
  - of the size of the original training set.

Random Forests

Random Forest (Breiman, 2001):

- an ensemble of decision trees (ensemble, i.e., final decisions reached by voting),
- unpruned;
- random (1):
  - at each node of a tree
  - a subset of attributes is randomly selected
  - from which the best attribute to further grow the tree is calculated;
- random (2; bagging, i.e., bootstrap aggregating):
  - for each tree,
  - bootstrap (randomly select with replacing) a multiset (bag) of training examples,
  - of the size of the original training set.

Random Forests

Random Forest (Breiman, 2001):

- an ensemble of decision trees (ensemble, i.e., final decisions reached by voting),
- unpruned;
- random (1):
  - at each node of a tree
  - a subset of attributes is randomly selected
  - from which the best attribute to further grow the tree is calculated;
- random (2; bagging, i.e., bootstrap aggregating):
  - for each tree,
  - bootstrap (randomly select with replacing) a multiset (bag) of training examples,
  - of the size of the original training set.

Balanced Random Forests

Balanced Random Forest (BRF; Chen *et al.* 2004):

- for each tree, instead of bootstrapping a bag of examples from the whole training set:
- separate the training set into positive and negative examples,
- bootstrap two multisets of the same size (the size of the smaller set of training examples),
- combine them into a training multiset for the tree.

Balanced Random Forests

Balanced Random Forest (BRF; Chen *et al.* 2004):

- for each tree, instead of bootstrapping a bag of examples from the whole training set:
- separate the training set into positive and negative examples,
- bootstrap two multisets of the same size (the size of the smaller set of training examples),
- combine them into a training multiset for the tree.

Balanced Random Forests

Balanced Random Forest (BRF; Chen *et al.* 2004):

- for each tree, instead of bootstrapping a bag of examples from the whole training set:
- separate the training set into positive and negative examples,
- bootstrap two multisets of the same size (the size of the smaller set of training examples),
- combine them into a training multiset for the tree.

Balanced Random Forests

Balanced Random Forest (BRF; Chen *et al.* 2004):

- for each tree, instead of bootstrapping a bag of examples from the whole training set:
- separate the training set into positive and negative examples,
- bootstrap two multisets of the same size (the size of the smaller set of training examples),
- combine them into a training multiset for the tree.

Balanced Random Forests

Balanced Random Forest (BRF; Chen *et al.* 2004):

- for each tree, instead of bootstrapping a bag of examples from the whole training set:
- separate the training set into positive and negative examples,
- bootstrap two multisets of the same size (the size of the smaller set of training examples),
- combine them into a training multiset for the tree.

Parameters of BRFs

- Which decision tree construction algorithm? CART (Classification and Regression Trees), as usual in Random Forests.
- If M is the total number of attributes, how many attributes to select randomly at each node? Here $m = \sqrt{M}$ (but this does not matter much; cf. Breiman 2001).
- How many trees in an ensemble? Best results for about 700–800.
- What attributes?

Parameters of BRFs

- Which decision tree construction algorithm? CART (Classification and Regression Trees), as usual in Random Forests.
- If M is the total number of attributes, how many attributes to select randomly at each node? Here $m = \sqrt{M}$ (but this does not matter much; cf. Breiman 2001).
- How many trees in an ensemble? Best results for about 700–800.
- What attributes?

Parameters of BRFs

- Which decision tree construction algorithm? CART (Classification and Regression Trees), as usual in Random Forests.
- If M is the total number of attributes, how many attributes to select randomly at each node? Here $m = \sqrt{M}$ (but this does not matter much; cf. Breiman 2001).
- How many trees in an ensemble? Best results for about 700–800.
- What attributes?

Parameters of BRFs

- Which decision tree construction algorithm? CART (Classification and Regression Trees), as usual in Random Forests.
- If M is the total number of attributes, how many attributes to select randomly at each node? Here $m = \sqrt{M}$ (but this does not matter much; cf. Breiman 2001).
- How many trees in an ensemble? Best results for about 700–800.
- What attributes?

Parameters of BRFs

- Which decision tree construction algorithm? CART (Classification and Regression Trees), as usual in Random Forests.
- If M is the total number of attributes, how many attributes to select randomly at each node? Here $m = \sqrt{M}$ (but this does not matter much; cf. Breiman 2001).
- How many trees in an ensemble? Best results for about 700–800.
- What attributes?

Parameters of BRFs

- Which decision tree construction algorithm? CART (Classification and Regression Trees), as usual in Random Forests.
- If M is the total number of attributes, how many attributes to select randomly at each node? Here $m = \sqrt{M}$ (but this does not matter much; cf. Breiman 2001).
- How many trees in an ensemble? Best results for about 700–800.

- What attributes?

Parameters of BRFs

- Which decision tree construction algorithm? CART (Classification and Regression Trees), as usual in Random Forests.
- If M is the total number of attributes, how many attributes to select randomly at each node? Here $m = \sqrt{M}$ (but this does not matter much; cf. Breiman 2001).
- How many trees in an ensemble? Best results for about 700–800.
- What attributes?

Attributes for BRFs

As usual in Machine Learning, each instance (here: sentence) represented by a vector of attributes and their values.

Attributes chosen for definition extraction:

- each attribute corresponds to an *n*-gram,
- and its binary value indicates the presence or absence of that *n*-gram in the sentence. (No improvement for frequencies.)

n-grams of what?

- base forms (lemmata),
- parts of speech (POSs, here called *ctags*),
- grammatical cases (this is Polish!).
- $1 \leq n \leq 3$

Examples of *types* of *n*-grams:

- <*base*, *base*, *case*>
- <*ctag*, *case*>

Attributes for BRFs

As usual in Machine Learning, each instance (here: sentence) represented by a vector of attributes and their values.

Attributes chosen for definition extraction:

- each attribute corresponds to an *n*-gram,
- and its binary value indicates the presence or absence of that *n*-gram in the sentence. (No improvement for frequencies.)

n-grams of what?

- base forms (lemmata),
- parts of speech (POSs, here called *ctags*),
- grammatical cases (this is Polish!).
- $1 \leq n \leq 3$

Examples of *types* of *n*-grams:

- <*base*, *base*, *case*>
- <*ctag*, *case*>

Attributes for BRFs

As usual in Machine Learning, each instance (here: sentence) represented by a vector of attributes and their values.

Attributes chosen for definition extraction:

- each attribute corresponds to an *n*-gram,
- and its binary value indicates the presence or absence of that *n*-gram in the sentence. (No improvement for frequencies.)

n-grams of what?

- base forms (lemmata),
- parts of speech (POSs, here called *ctags*),
- grammatical cases (this is Polish!).
- $1 \leq n \leq 3$

Examples of *types* of *n*-grams:

- <*base*, *base*, *case*>
- <*ctag*, *case*>

Attributes for BRFs

As usual in Machine Learning, each instance (here: sentence) represented by a vector of attributes and their values.

Attributes chosen for definition extraction:

- each attribute corresponds to an *n*-gram,
- and its binary value indicates the presence or absence of that *n*-gram in the sentence. (No improvement for frequencies.)

n-grams of what?

- base forms (lemmata),
- parts of speech (POSs, here called *ctags*),
- grammatical cases (this is Polish!).
- $1 \leq n \leq 3$

Examples of *types* of *n*-grams:

- <*base*, *base*, *case*>
- <*ctag*, *case*>

Attributes for BRFs

As usual in Machine Learning, each instance (here: sentence) represented by a vector of attributes and their values.

Attributes chosen for definition extraction:

- each attribute corresponds to an *n*-gram,
- and its binary value indicates the presence or absence of that *n*-gram in the sentence. (No improvement for frequencies.)

n-grams of what?

- base forms (lemmata),
- parts of speech (POSs, here called *ctags*),
- grammatical cases (this is Polish!).
- $1 \leq n \leq 3$

Examples of *types* of *n*-grams:

- <*base*, *base*, *case*>
- <*ctag*, *case*>

Attributes for BRFs

As usual in Machine Learning, each instance (here: sentence) represented by a vector of attributes and their values.

Attributes chosen for definition extraction:

- each attribute corresponds to an n-gram,
- and its binary value indicates the presence or absence of that n-gram in the sentence. (No improvement for frequencies.)

n-grams of what?

- base forms (lemmata),
- parts of speech (POSs, here called *ctags*),
- grammatical cases (this is Polish!).
- $1 \leq n \leq 3$

Examples of *types* of n-grams:

- <*base*, *base*, *case*>
- <*ctag*, *case*>

Attributes for BRFs

As usual in Machine Learning, each instance (here: sentence) represented by a vector of attributes and their values.

Attributes chosen for definition extraction:

- each attribute corresponds to an *n*-gram,
- and its binary value indicates the presence or absence of that *n*-gram in the sentence. (No improvement for frequencies.)

n-grams of what?

- base forms (lemmata),
- parts of speech (POSs, here called *ctags*),
- grammatical cases (this is Polish!).
- $1 \leq n \leq 3$

Examples of *types* of *n*-grams:

- <*base*, *base*, *case*>
- <*ctag*, *case*>

Attributes for BRFs (contd.)

There are $3^1 + 3^2 + 3^3 = 39$ possible types of *n*-grams.

10 selected on the basis of:

- their informativeness (measured by the average χ^2 statistic for 100 most common *n*-grams of each type) w.r.t. the definition/non-definition distinction,
- rejection of longer *n*-gram types statistically dependent on shorter *n*-gram types.

n-gram types selected:

no.	*n*-gram type	no.	*n*-gram type
1	<*base*>	6	<*base*, *base*>
2	<*ctag*, *ctag*, *case*>	7	<*ctag*, *ctag*>
3	<*ctag*, *base*>	8	<*ctag*, *case*>
4	<*base*, *case*>	9	<*base*, *base*, *base*>
5	<*base*, *ctag*>	10	<*ctag*>

Attributes for BRFs (contd.)

There are $3^1 + 3^2 + 3^3 = 39$ possible types of *n*-grams.

10 selected on the basis of:

- their informativeness (measured by the average χ^2 statistic for 100 most common *n*-grams of each type) w.r.t. the definition/non-definition distinction,
- rejection of longer *n*-gram types statistically dependent on shorter *n*-gram types.

n-gram types selected:

no.	*n*-gram type	no.	*n*-gram type
1	*<base>*	6	*<base, base>*
2	*<ctag, ctag, case>*	7	*<ctag, ctag>*
3	*<ctag, base>*	8	*<ctag, case>*
4	*<base, case>*	9	*<base, base, base>*
5	*<base, ctag>*	10	*<ctag>*

Attributes for BRFs (contd.)

There are $3^1 + 3^2 + 3^3 = 39$ possible types of *n*-grams.

10 selected on the basis of:

- their informativeness (measured by the average χ^2 statistic for 100 most common *n*-grams of each type) w.r.t. the definition/non-definition distinction,
- rejection of longer *n*-gram types statistically dependent on shorter *n*-gram types.

n-gram types selected:

no.	*n*-gram type	no.	*n*-gram type
1	$<$*base*$>$	6	$<$*base*, *base*$>$
2	$<$*ctag*, *ctag*, *case*$>$	7	$<$*ctag*, *ctag*$>$
3	$<$*ctag*, *base*$>$	8	$<$*ctag*, *case*$>$
4	$<$*base*, *case*$>$	9	$<$*base*, *base*, *base*$>$
5	$<$*base*, *ctag*$>$	10	$<$*ctag*$>$

Attributes for BRFs (contd.)

Which *n*-grams of each type?

- For each *n*-gram type,
- separately for definitions and non-definitions,
- find the frequencies of various *n*-grams of that type,
- merge the two lists ordered by relative frequency,
- take the first 100 different *n*-grams from that list;
- altogether 929 different attributes (10 *n*-gram types $\times$ 100 *n*-grams, but there are fewer than 100 <*ctag*> unigrams),
- for 10 830 instances (sentences).

Attributes for BRFs (contd.)

Which *n*-grams of each type?

- For each *n*-gram type,
- separately for definitions and non-definitions,
- find the frequencies of various *n*-grams of that type,
- merge the two lists ordered by relative frequency,
- take the first 100 different *n*-grams from that list;
- altogether 929 different attributes (10 *n*-gram types × 100 *n*-grams, but there are fewer than 100 <*ctag*> unigrams),
- for 10 830 instances (sentences).

Attributes for BRFs (contd.)

Which *n*-grams of each type?

- For each *n*-gram type,
- separately for definitions and non-definitions,
- find the frequencies of various *n*-grams of that type,
- merge the two lists ordered by relative frequency,
- take the first 100 different *n*-grams from that list;
- altogether 929 different attributes (10 *n*-gram types × 100 *n*-grams, but there are fewer than 100 <*ctag*> unigrams),
- for 10 830 instances (sentences).

Attributes for BRFs (contd.)

Which *n*-grams of each type?

- For each *n*-gram type,
- separately for definitions and non-definitions,
- find the frequencies of various *n*-grams of that type,
- merge the two lists ordered by relative frequency,
- take the first 100 different *n*-grams from that list;
- altogether 929 different attributes (10 *n*-gram types $\times$ 100 *n*-grams, but there are fewer than 100 <*ctag*> unigrams),
- for 10 830 instances (sentences).

Attributes for BRFs (contd.)

Which *n*-grams of each type?

- For each *n*-gram type,
- separately for definitions and non-definitions,
- find the frequencies of various *n*-grams of that type,
- merge the two lists ordered by relative frequency,
- take the first 100 different *n*-grams from that list;
- altogether 929 different attributes (10 *n*-gram types $\times$ 100 *n*-grams, but there are fewer than 100 $<$*ctag*$>$ unigrams),
- for 10 830 instances (sentences).

Attributes for BRFs (contd.)

Which *n*-grams of each type?

- For each *n*-gram type,
- separately for definitions and non-definitions,
- find the frequencies of various *n*-grams of that type,
- merge the two lists ordered by relative frequency,
- take the first 100 different *n*-grams from that list;
- altogether 929 different attributes (10 *n*-gram types $\times$ 100 *n*-grams, but there are fewer than 100 <*ctag*> unigrams),
- for 10 830 instances (sentences).

Attributes for BRFs (contd.)

Which *n*-grams of each type?

- For each *n*-gram type,
- separately for definitions and non-definitions,
- find the frequencies of various *n*-grams of that type,
- merge the two lists ordered by relative frequency,
- take the first 100 different *n*-grams from that list;
- altogether 929 different attributes (10 *n*-gram types $\times$ 100 *n*-grams, but there are fewer than 100 $<$*ctag*$>$ unigrams),
- for 10 830 instances (sentences).

Attributes for BRFs (contd.)

Which *n*-grams of each type?

- For each *n*-gram type,
- separately for definitions and non-definitions,
- find the frequencies of various *n*-grams of that type,
- merge the two lists ordered by relative frequency,
- take the first 100 different *n*-grams from that list;

- altogether 929 different attributes (10 *n*-gram types × 100 *n*-grams, but there are fewer than 100 <*ctag*> unigrams),
- for 10 830 instances (sentences).

Results

Best results:

	P	R	$F_{\alpha=2}$
new attributes:			
BRFs (700 trees)	21.4%	69.0%	**39.6**
old attributes:			
previous best: hybrid (Przepiórkowski *et al.*, 2008)	25.2%	53.5%	**38.9**
previous best: linguistic (Przepiórkowski *et al.*, 2007)	18.7%	59.3%	**34.4**
BRFs (800 trees)	17.0%	64.1%	**33.4**
previous best: pure ML (SVM; Degórski *et al.* 2008b)	20.4%	38.5%	**29.7**

BRFs and grammars

Linguistic insights are useful, after all?

Here: $F_{\alpha=2} = \mathbf{39.60}$.

Degórski *et al.* 2008a:

- additional filtering by the naïve grammar: $F_{\alpha=2} = \mathbf{40.95}$ (relative gain of 3.4%);
- additional fine-tuning of BRFs: $F_{\alpha=2} = \mathbf{42.47}$,
- additional fine-tuning and filtering: $F_{\alpha=2} = \mathbf{43.09}$ (relative gain of 1.5%).

BRFs and grammars

Linguistic insights are useful, after all?

Here: $F_{\alpha=2} = \mathbf{39.60}$.

Degórski *et al.* 2008a:

- additional filtering by the naïve grammar: $F_{\alpha=2} = \mathbf{40.95}$ (relative gain of 3.4%);
- additional fine-tuning of BRFs: $F_{\alpha=2} = \mathbf{42.47}$,
- additional fine-tuning and filtering: $F_{\alpha=2} = \mathbf{43.09}$ (relative gain of 1.5%).

BRFs and grammars

Linguistic insights are useful, after all?

Here: $F_{\alpha=2} = \mathbf{39.60}$.

Degórski *et al.* 2008a:

- additional filtering by the naïve grammar: $F_{\alpha=2} = \mathbf{40.95}$ (relative gain of 3.4%);
- additional fine-tuning of BRFs: $F_{\alpha=2} = \mathbf{42.47}$,
- additional fine-tuning and filtering: $F_{\alpha=2} = \mathbf{43.09}$ (relative gain of 1.5%).

Conclusions

For the definition extraction task at hand:

- careful procedure of n-gram selection significantly improves the results,
- independently, Balanced Random Forest is a significantly better classifier than classifiers commonly used in NLP, including SVM and AdaBoost,
- filtering by an additional simplistic grammar of little help.

This task is typical of many NLP tasks (excluding POS tagging!):

- small data size,
- noisy,
- heavily imbalanced.

So perhaps BRFs will turn out to be the new hot ML method in NLP?

Conclusions

For the definition extraction task at hand:

- careful procedure of *n*-gram selection significantly improves the results,
- independently, Balanced Random Forest is a significantly better classifier than classifiers commonly used in NLP, including SVM and AdaBoost,
- filtering by an additional simplistic grammar of little help.

This task is typical of many NLP tasks (excluding POS tagging!):

- small data size,
- noisy,
- heavily imbalanced.

So perhaps BRFs will turn out to be the new hot ML method in NLP?

Conclusions

For the definition extraction task at hand:

- careful procedure of *n*-gram selection significantly improves the results,
- independently, Balanced Random Forest is a significantly better classifier than classifiers commonly used in NLP, including SVM and AdaBoost,
- filtering by an additional simplistic grammar of little help.

This task is typical of many NLP tasks (excluding POS tagging!):

- small data size,
- noisy,
- heavily imbalanced.

So perhaps BRFs will turn out to be the new hot ML method in NLP?

Conclusions

For the definition extraction task at hand:

- careful procedure of *n*-gram selection significantly improves the results,
- independently, Balanced Random Forest is a significantly better classifier than classifiers commonly used in NLP, including SVM and AdaBoost,
- filtering by an additional simplistic grammar of little help.

This task is typical of many NLP tasks (excluding POS tagging!):

- small data size,
- noisy,
- heavily imbalanced.

So perhaps BRFs will turn out to be the new hot ML method in NLP?

Conclusions

For the definition extraction task at hand:

- careful procedure of *n*-gram selection significantly improves the results,
- independently, Balanced Random Forest is a significantly better classifier than classifiers commonly used in NLP, including SVM and AdaBoost,
- filtering by an additional simplistic grammar of little help.

This task is typical of many NLP tasks (excluding POS tagging!):

- small data size,
- noisy,
- heavily imbalanced.

So perhaps BRFs will turn out to be the new hot ML method in NLP?

Conclusions

For the definition extraction task at hand:

- careful procedure of *n*-gram selection significantly improves the results,
- independently, Balanced Random Forest is a significantly better classifier than classifiers commonly used in NLP, including SVM and AdaBoost,
- filtering by an additional simplistic grammar of little help.

This task is typical of many NLP tasks (excluding POS tagging!):

- small data size,
- noisy,
- heavily imbalanced.

So perhaps BRFs will turn out to be the new hot ML method in NLP?

Thank you for your attention!

Breiman, L. (2001). Random forests. *Machine Learning*, **45**, 5–32.

Chen, C., Liaw, A., and Breiman, L. (2004). Using random forest to learn imbalanced data. Technical Report 666, University of California, Berkeley. http://www.stat.berkeley.edu/tech-reports/666.pdf.

Degórski, Ł., Kobyliński, Ł., and Przepiórkowski, A. (2008a). Definition extraction: Improving balanced random forests. In *Proceedings of the International Multiconference on Computer Science and Information Technology (IMCSIT 2008): Computational Linguistics – Applications (CLA'08)*, Wisła, Poland. PTI.

Degórski, Ł., Marcińczuk, M., and Przepiórkowski, A. (2008b). Definition extraction using a sequential combination of baseline grammars and machine learning classifiers. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation, LREC 2008*, Marrakech. ELRA.

Przepiórkowski, A., Degórski, Ł., and Wójtowicz, B. (2007). On the evaluation of Polish definition extraction grammars. In Z. Vetulani, editor, *Proceedings of the 3rd Language & Technology Conference*, pages 473–477, Poznań, Poland.

Przepiórkowski, A., Marcińczuk, M., and Degórski, Ł. (2008). Dealing with small, noisy and imbalanced data: Machine learning or manual grammars? In P. Sojka, I. Kopeček, and K. Pala, editors, *Text, Speech and Dialogue: 9th International Conference, TSD 2008, Brno, Czech Republic, September 2008*, Lecture Notes in Artificial Intelligence, Berlin. Springer-Verlag. Forthcoming.

Tobin, R. (2005). *Lxtransduce, a replacement for fsgmatch*. University of Edinburgh. `http://www.cogsci.ed.ac.uk/~richard/ltxml2/lxtransduce-manual.html`.