Drug-Drug Interaction Extraction

Matthias Hertel & Martin Guy

Advanced Natural Language Processing

Spring 2017

Outline

I. **Introduction**

  A. Explanation of tasks

  B. Performance measures

II. **Recognition and classification of drug names**

  A. Baseline approach

  B. Naive Bayes

  C. Results

III. **DDI classification**

  A. First approach: SVM

  B. Second approach: Naive Bayes

  C. Results

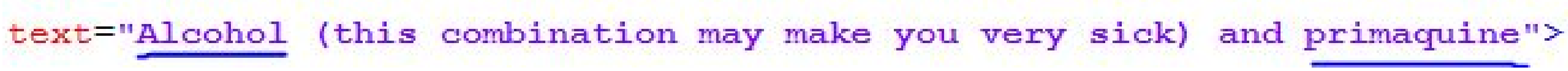
IV. **Conclusions**

Introduction

Explanation of the tasks

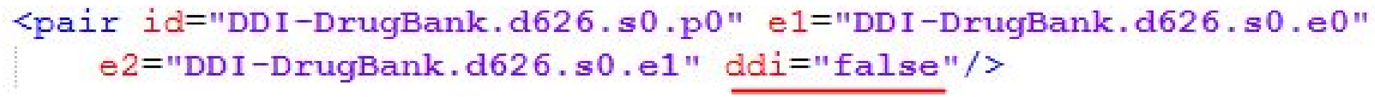
Extraction of Drug-Drug Interactions from Biomedical texts

- **Task 1 :** Recognition and classification of drug names



```
text="Alcohol (this combination may make you very sick) and primaquine">
```

- **Task 2 :** Extraction and classification of their interactions



```
<pair id="DDI-DrugBank.d626.s0.p0" e1="DDI-DrugBank.d626.s0.e0"
    e2="DDI-DrugBank.d626.s0.e1" ddi="false"/>
```

Task 1 : Classes of drugs

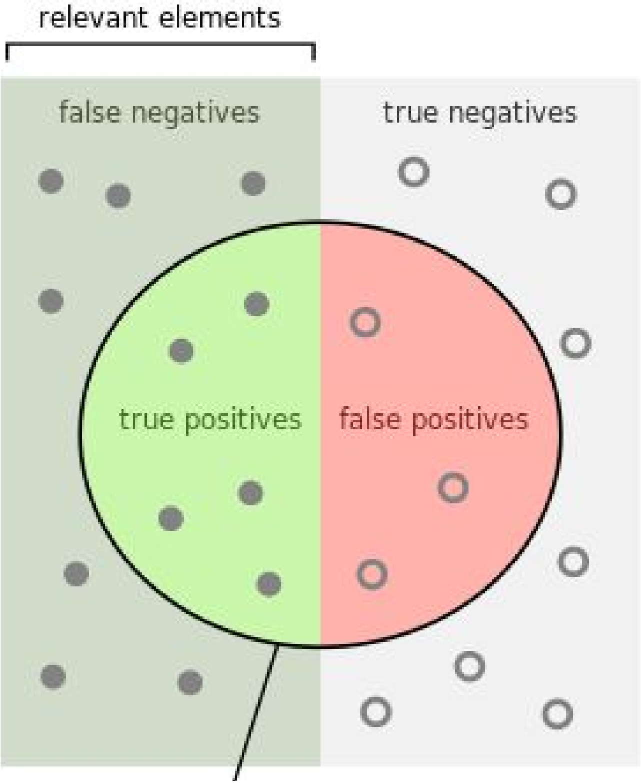
Class	Description
brand	branded drug names
drug	generic drug names
drug_n	active substances not approved for human use
group	drug group names

Task 2 : Classes of interaction

Class	Description
null	no interaction between pairs
effect	effect of DDI is described
mechanism	when the process by which drugs are absorbed, metabolised... are affected is described
advice	a recommendation or advice is given about using the two drugs in the pair.
int	interaction but no information about the type

Performance measures

	truth: class c	truth: other or no class
prediction: class c	true positive	false positive
prediction: other or no class	false negative	true negative



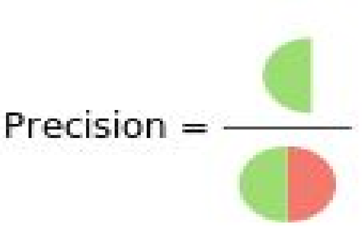
Performance measures

Class-wise measures

- Precision (positive predictive value):

$$\text{Precision} = \frac{tp}{tp + fp}$$

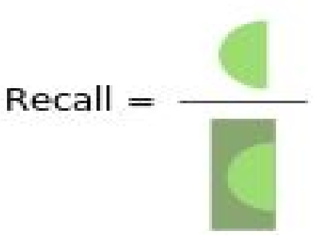
How many selected items are relevant?



- Recall:

$$\text{Recall} = \frac{tp}{tp + fn}$$

How many relevant items are selected?



Performance measures

Class-wise measures

- F1-measure:

$$F_1 = 2 \cdot \frac{1}{\frac{1}{\mathrm{recall}} + \frac{1}{\mathrm{precision}}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

Performance measures

Overall performance measures

- Accuracy:

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

- Micro-averaged F1: compute precision, recall and F1 using the sums of *tp*, *fp* and *fn* for all classes.

- Macro-averaged F1: average precision and recall over classes, then compute F1 using the average precision and average recall.

Task 1 : Recognition and classification of drug names

Baseline approach

Training: store (drug_name, class) pairs in a dictionary

Prediction: search for the stored drug names, return according classes

Problem 1: false positives for words “drug” and “drugs”

Solution: don’t predict them

Problem 2: some drug names are substring of others

Solution:

- don’t predict if overlap with another prediction
- predict beginning with the longest drug names

Baseline results

- good precision
- bad recall
- micro-averaged f-measure = 56,31 %

Naive Bayes Classifier

$$p(\text{class=}''\text{effect}''|\text{feature\_vector})$$

$$= \frac{p(\text{feature\_vector}|\text{class=}''\text{effect}'') * p(\text{class=}''\text{effect}'')}{p(\text{feature\_vector})}$$

$$= \frac{p(\text{class} = ''\text{effect}'')}{p(\text{feature\_vector})} \prod_{featuref_i} p(f_i = \text{value}_i|\text{class} = ''\text{effect}'')$$

Classification:

given a vector of features,

predict class c that maximizes $p(\text{class} = \text{c}|\text{feature\_vector})$

Naive Bayes Classifier

Training:

estimate parameters from training samples

$$p(\text{feature} = \text{value}|\text{class} = \text{c}) = \frac{count(\text{feature} = \text{value}|\text{class=c})}{count(\text{feature} = \text{any\_value}|\text{class=c})}$$

$$p(\text{class} = \text{c}) = \frac{count(\text{class=c})}{n}$$

Smoothing techniques:

- Laplace smoothing
- epsilon smoothing

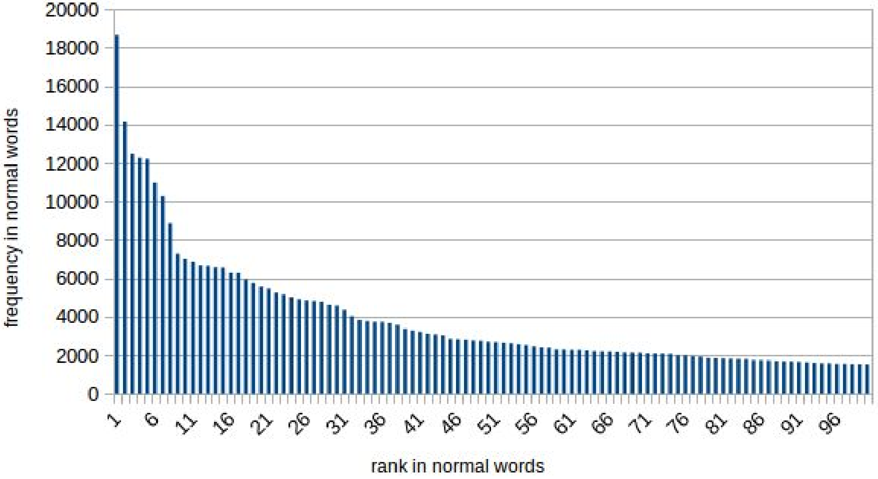
Features

Extract words using a tokenizer

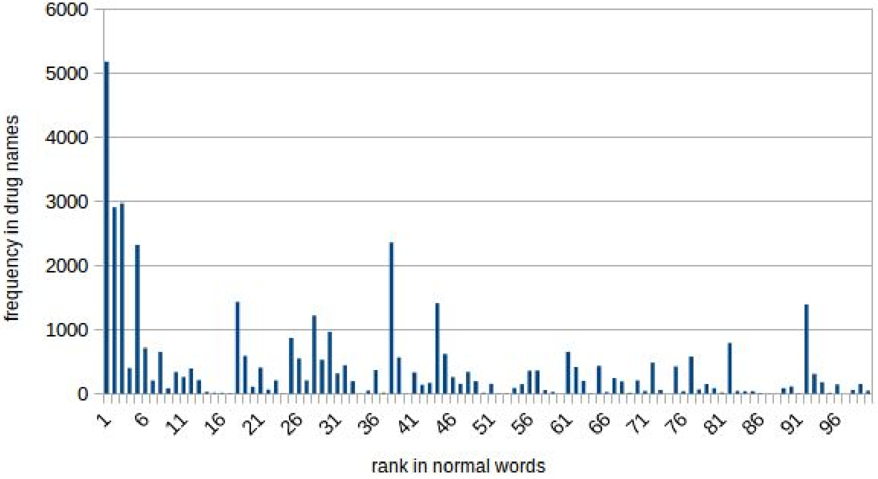
For each word, generate a feature vector, containing

- n-grams of the word
- word features
    - word length
    - first letter uppercase
    - last letter ‘s’ (plural-s)
- context lemmas
    - lemmas of a 5-word-window
- POS-tags
    - POS-tags of a 5-word-window

3-gram frequencies



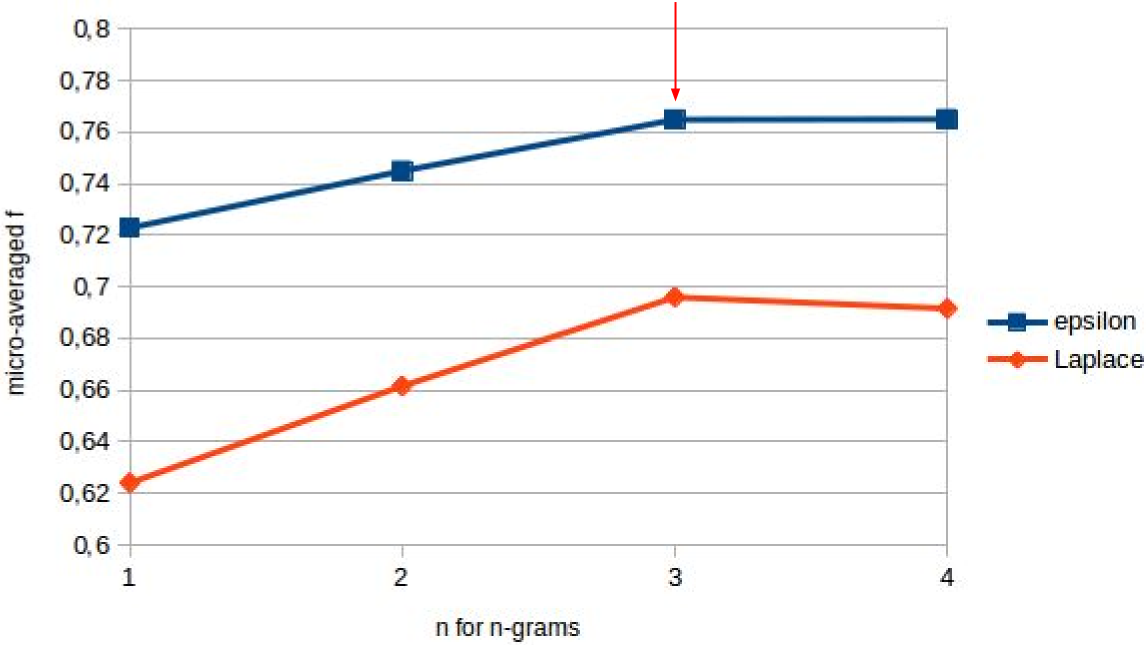
English words



drug names

Parameter selection

Parameters were selected with a 10-fold cross-validation on the training data



Naive Bayes Results

Class prediction performance

class	**precision**	**recall**	**f-measure**
brand	50.00 %	76.27 %	60.40 %
drug	60,63 %	81,19 %	69,42 %
drug_n	15,62 %	8,26 %	10,81 %
group	46,82 %	52,25 %	49,39 %

Drug extraction (regardless of class) performance

class	**precision**	**recall**	**f-measure**
any	60,10 %	69,82 %	64,59 %

Naive Bayes Results

Overall performance

- macro-averaged F1 = 48,23 %
- **micro-averaged F1= 56,77 %**
  - better than the baseline (but close)
  - we would have placed 12th out of 17 at the conference
  - best result there was 71,50 %

Difficulties

Our system fails when a drug name is longer than one word

Possible solutions:

- concatenate consecutive predictions of the same class
- train another Naive Bayes on two word sequences
- or train a classifier to predict when a word has to be added

Drugs can be substrings of words

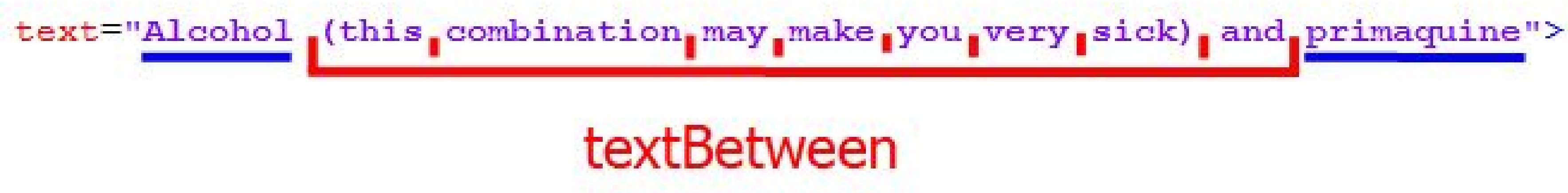
e.g. in “alcohol-addicted”

- need to classify substrings too

Task 2 : Drug-Drug Interaction Classification

II. DDI classification

A. First approach : SVM with a) MostFrequentBetween Strategy

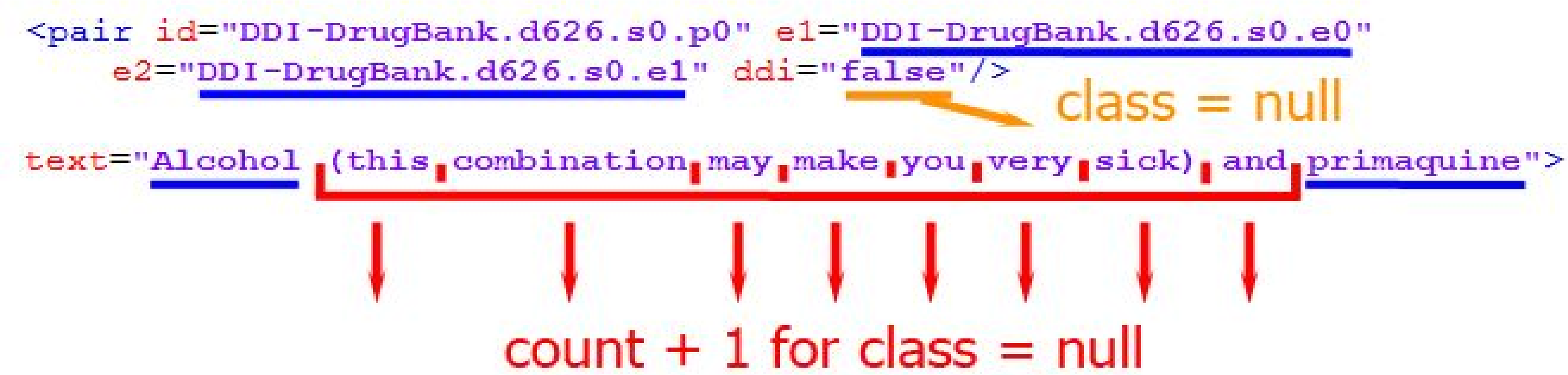


```
text="Alcohol (this combination may make you very sick) and primaquine">
```

- For each pair, count the number of occurrence of each words between entities in the pair.
- Take the first *n* **most occurring words** as features.

II. DDI classification

A. First approach : SVM with b) Entropy Strategy



```
<pair id="DDI-DrugBank.d626.s0.p0" e1="DDI-DrugBank.d626.s0.e0"
    e2="DDI-DrugBank.d626.s0.e1" ddi="false"/>
text="Alcohol (this combination may make you very sick) and primaquine">
```

- For each pair, count the number of occurrence of each words between entities in the pair, categorized in each class.
- Compute entropy of each word.
- Take the first *n* words with **lowest entropy** as features.

II. DDI classification

A. First approach : SVM with b) Entropy Strategy

Problem:

- Predicted always the **null** class → unbalanced dataset
- Class weighting helped to break that
- But not enough : problem with features
- Tuning of the C parameter did not change anything
- Too slow to train

II. DDI classification

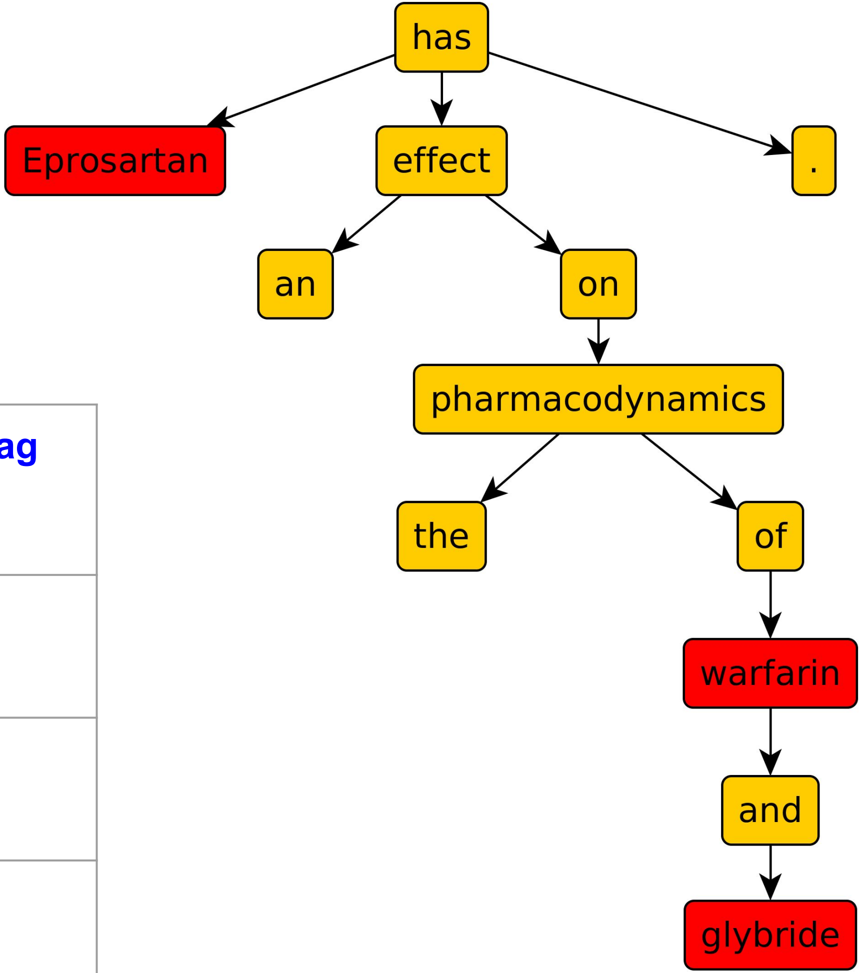
B. Second approach : Naive Bayes with more features

- Syntactic features using the dependency parse tree
- Words between the entities
- Negations
- Features from the entities
- Handcrafted key-words

Syntactic features

“**Eprosartan** has an effect on the pharmacodynamics of **warfarin** and **glybride**.”

drug 1	**drug 2**	**class**	**lowest common ancestor**	**POS tag**
Eprosartan	warfarin	effect	has	VBZ
Eprosartan	glybride	effect	has	VBZ
glybride	warfarin	none	warfarin	NN



II. DDI classification

B. Second approach : selected features

- syntactic
    - word and POS-tag of the lowest common ancestor in parse tree
- word features
    - words between the entities
    - number of words in between
- semantic
    - negation (“no”, “not”, “none”, “never”) in between
    - negation somewhere in the sentence
- key-words (binary features)
    - “increase”, “increases”, “decrease”, “decreases”
    - “no”, “not”, “none”, “never
    - “no” and “effect” in the sentence
- are the entities the same? (based on the drug names)

II. DDI classification

B. Second approach : rejected features

Features are not used when they do not improve the *micro-averaged f-measure* in a **10-fold cross-validation** on the training data

- type of the entities
- the entities’ texts
- all words in the sentence

II. DDI classification

B. Second approach : Naive Bayes with more features

Possible improvements:

- Consider text “outside” a pair
- Try other classifiers that could suit more to this task
- Add more semantic and syntactic features using NLP
- First learn classifier to recognize interaction/no interaction, then recognize the type of interaction

II. DDI classification

C. Results

Class results

Class	Precision	Recall	F1 Measure
null	92,15 %	84,27 %	88,03 %
advise	36,97 %	52,03 %	43,23 %
effect	32,56 %	55,00 %	40,90 %
int	72,22 %	13,54 %	22,80 %
mechanism	37,25 %	55,62 %	44,62 %

II. DDI classification

C. Results

Overall results

- interaction (regardless of type)
    - precision = 45,82 %
    - recall = 64,96 %
    - F1 = 53,73 %
- macro-averaged F1 (without class “null”) = 44,39 %
- **micro-averaged F1 (without class “null”) = 41,74 %**
    - we would have scored 19th out of 23 at the SemEval competition
    - best result there was 65,10 %

Conclusions

- Hard task to extract useful features of texts “by hand”
- Use of NLP tools helps a lot
- Some features improve results, others don’t
- Unbalanced dataset
  - difficult to predict low-frequent classes (“drug_n” and “int”)
  - SVM always predicted the same
- Naive Bayes more efficient than SVM on this task

Thank you for listening