Master in Artificial Intelligence

Advanced Human Language Technologies

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Machine Learning DDI

Relation Extraction

General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Task 2.1 - DDI using machine learning

Machine Learning DDI

Relation Extraction

General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables

Assignment

The main program parses input XML file and classifies drug-drug interactions between pairs of drugs, using a machine learning classification algorithm.

```
$ python3 ./ml-DDI.py data/devel/ > result.out
DDI-DrugBank.d398.s0|DDI-DrugBank.d398.s0.e0|DDI-DrugBank.d398.s0.e1|effect
DDI-DrugBank.d398.s0|DDI-DrugBank.d398.s0.e0|DDI-DrugBank.d398.s0.e2|effect
DDI-DrugBank.d211.s2|DDI-DrugBank.d211.s2.e0|DDI-DrugBank.d211.s2.e5|mechanism
...
```

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Relation Extraction

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables

- Relation Extraction is a NLP task, frequently required in Information Extraction applications.
- The goal of the task is to extract relations between entities (previously detected), expressed in the text. E.g.: `is_CEO_of(Person,Organization)`:
  - *Steve Jobs was the chairman, the chief executive officer (CEO), and a co-founder of Apple Inc., ...*
  - *During his career at Microsoft, Bill Gates held the positions of chairman, chief executive officer (CEO), president and chief software architect.*
  - *Mark Zuckerberg is known for co-founding Facebook, Inc. and serves as its chairman, chief executive officer, and controlling shareholder.*

Relation Extraction

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables

Other examples:

- Medical domain:
  `caused_by(diagnose,drug)`
  `prescribed_for(drug,diagnose)`
  `drug_interaction(drug,drug)`
- Legal domain:
  `is_suing(Person/Org,Person/Org)`
  `is_representing(Person,Person/Org)`
  `is_sentenced_for(Person/Org,Crime)`
  `is_sentenced_to(Person/Org,Penalty)`
- Business/Economy:
  `is_CEO_of(Person,Organization)`
  `absorbed_by(Organization,Organization)`
- etc.

Relation Extraction

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables

Relation Extraction can be approached as a classical ML classification task, where:

- The objects to be classified are a text fragment (sentence, paragraph...) plus a pair of target entities in it.
- Each object *(text,entity1,entity2)* is encoded as a feature vector.
- The output class is either `None`, or one relation type chosen among a *predefined list*.

Informative enough features are crucial to get good results.

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

General Structure

Machine Learning DDI

Relation Extraction

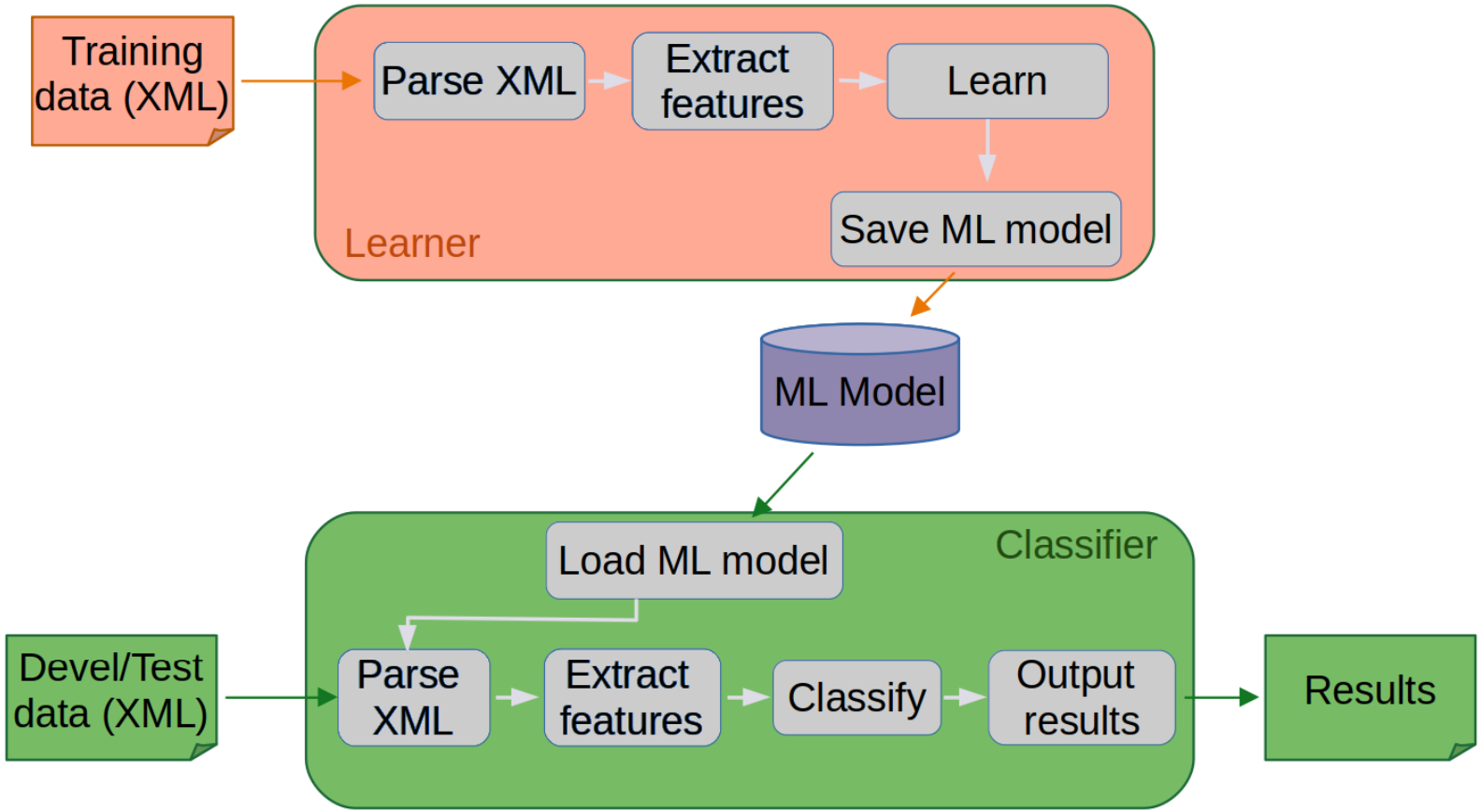
General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables



General Structure

Machine Learning DDI

Relation Extraction

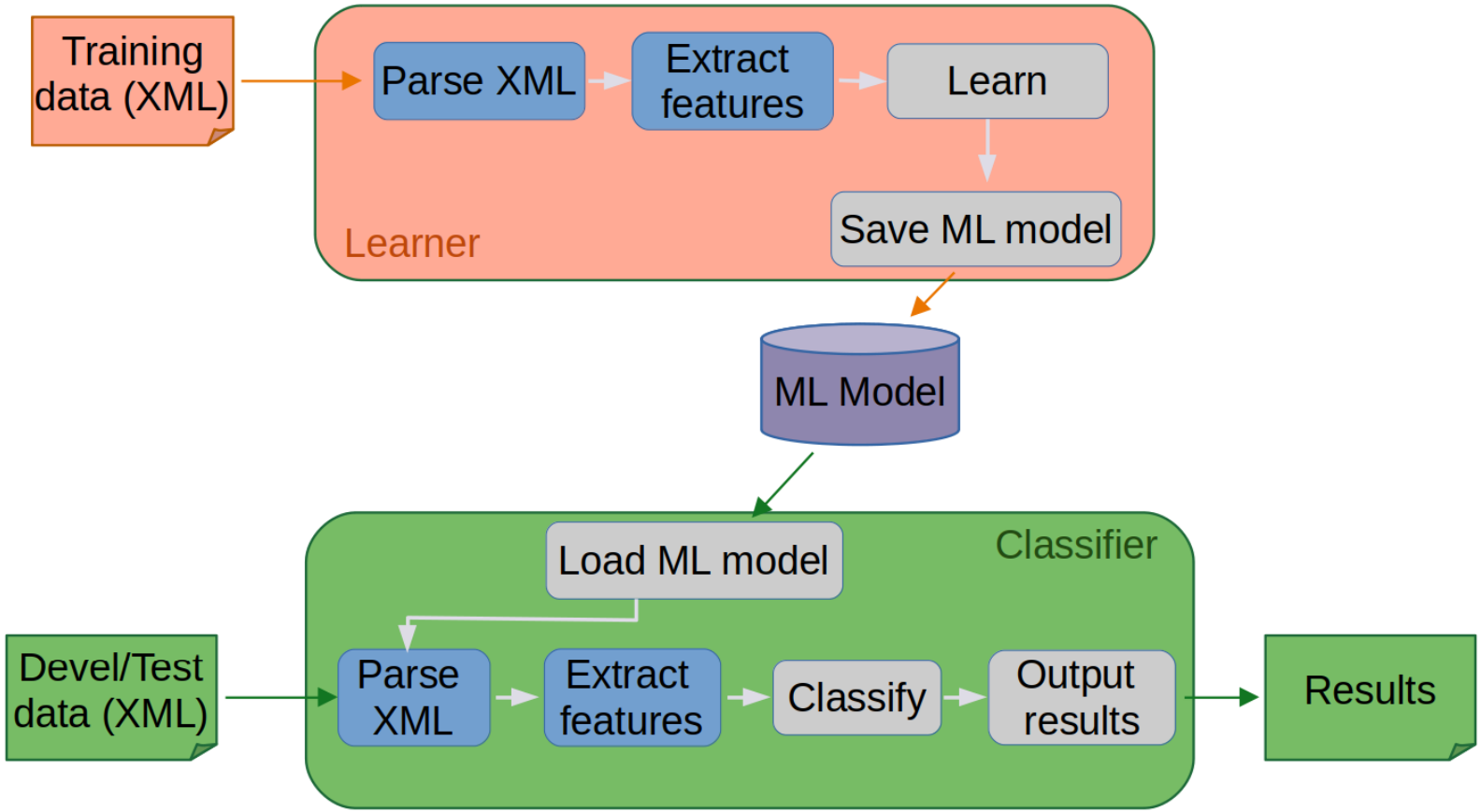
General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables



Extracting features is a costly operation, which we do not want to repeat for every possible experiment or algorithm parametrization.

General Structure

Machine Learning DDI

Relation Extraction

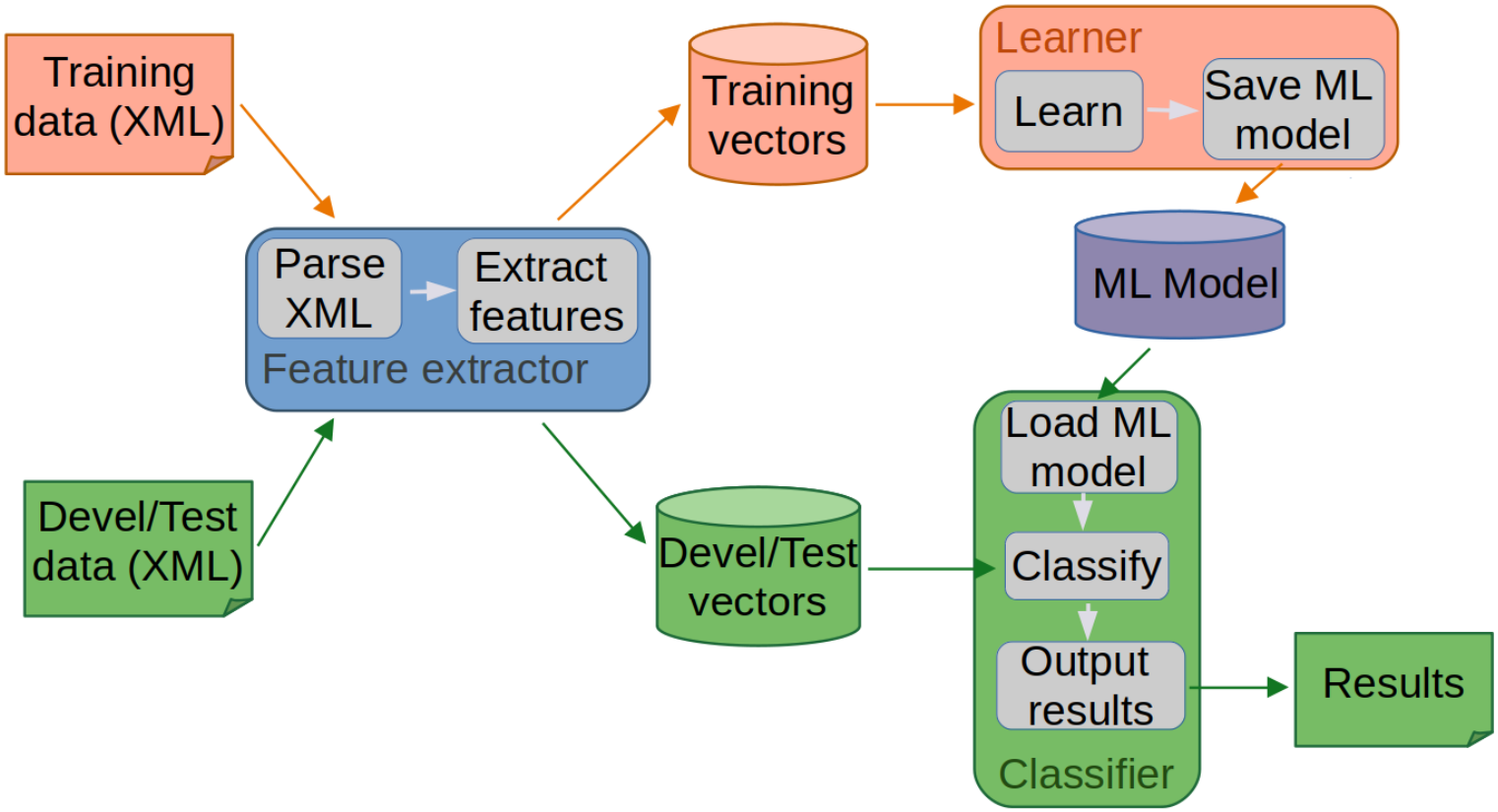
General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables



Feature extraction process is performed once, out of learning or predicting processes.

General Structure

Machine Learning DDI

Relation Extraction

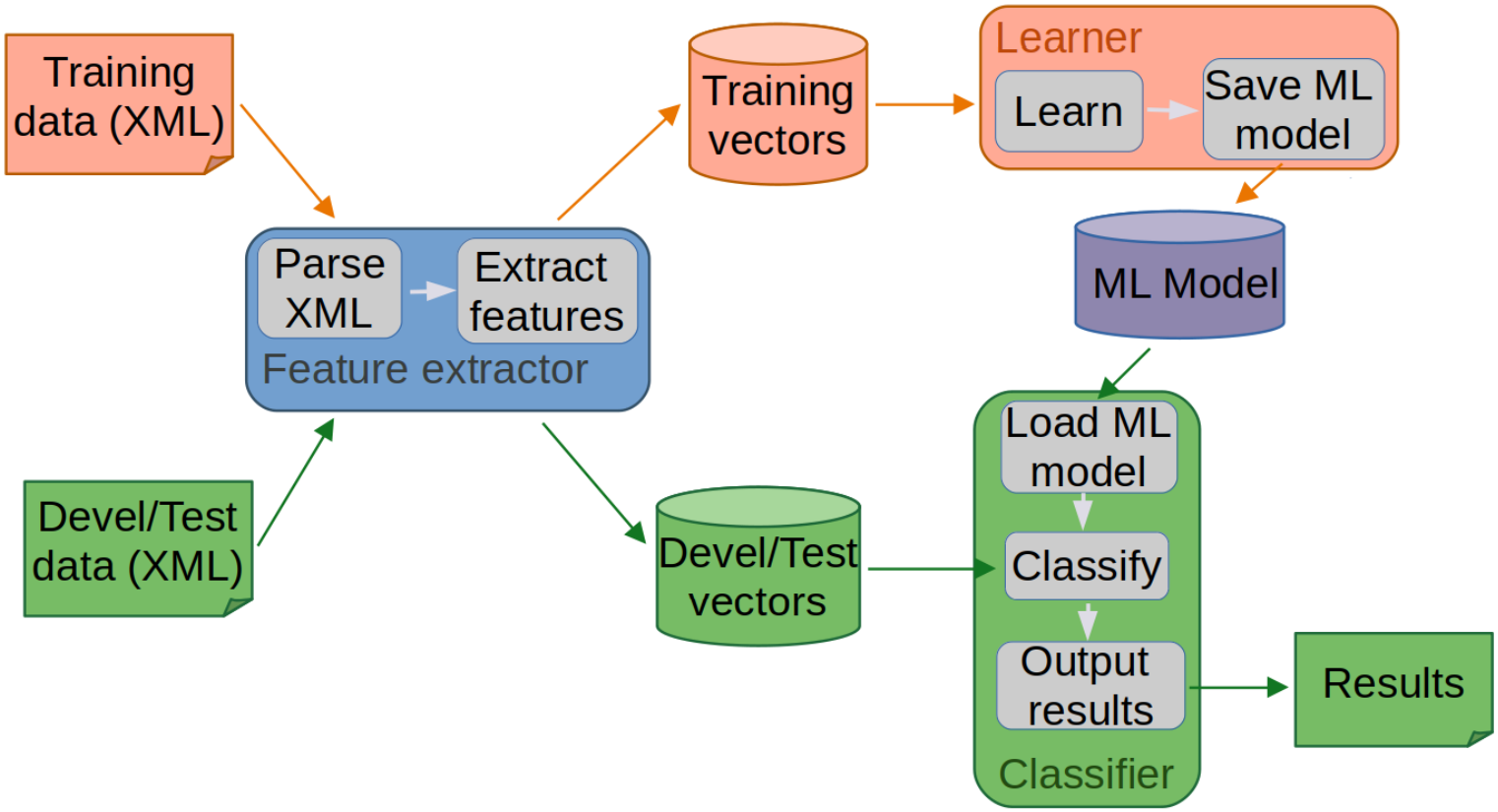
General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables



Feature extraction process is performed once, out of learning or predicting processes.

Thus, we need to write not a single program, but three different components: feature extractor, learner, and classifier.

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Resources

Machine Learning DDI

Relation Extraction

General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables

We will use SpaCy NLP suite, which integrates a tokenizer, a part-of-speech tagger, and a dependency parser.

- Install Spacy
- Output is always a list of tokens.
  Dependency tree information is inside each token.
  Check the API for details.

Resources - Spacy results

Machine Learning DDI

Relation Extraction

General Structure

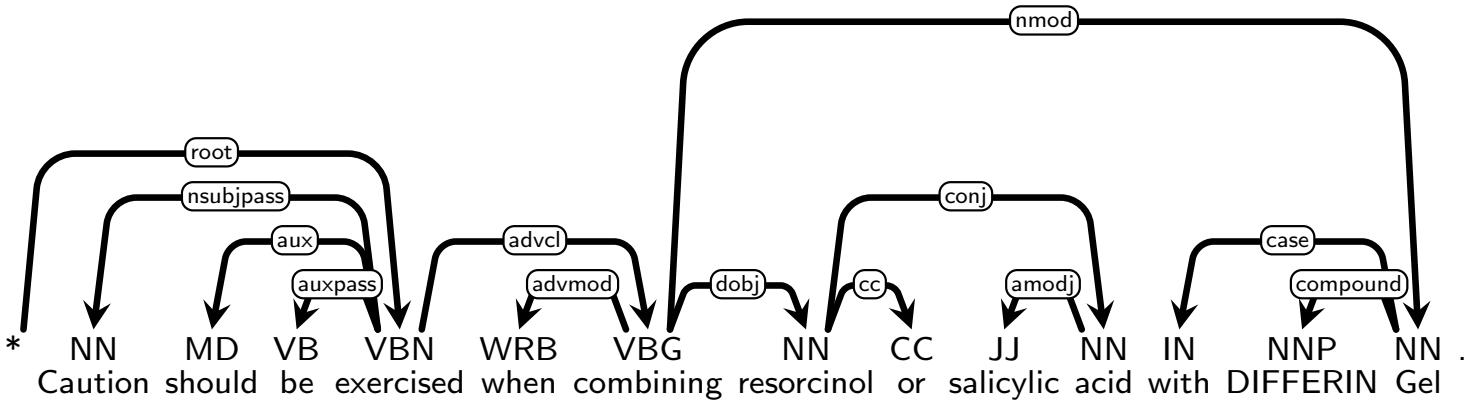
Resources

Detailed Structure

Core task

Goals & Deliverables

```
nlp = spacy.load("en_core_web_trf")
text = 'Caution should be exercised when combining resorcinol or
        salicylic acid with DIFFERIN Gel'
analysis = nlp(text)
for tk in analysis :
   print(tk.text)   # word form
   print(tk.lemma_) # lemma
   print(tk.pos_)   # coarse part-of-speech (NOUN, VERB, ADJ, ...)
   print(tk.tag_)   # coarse part-of-speech (NN, NNS, VB, VBZ, VBD, ..
   print(tk.head_)  # parent token
   print(tk.dep_)   # dependency function wrt parent
   for c in tk.children : ...  # iterator over token children nodes
   for c in tk.ancestors : ...  # iterator over ancestor chain
etc.
```



Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Feature Extractor

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Feature Extractor
Core task
Goals & Deliverables

The feature extractor:

- Independent program, separated from learner and classifier
- Receives as argument the XML file to encode.
- Prints the feature vectors to given output file

```
$ python3 ./extract_features.py data/devel devel.feat
$ more devel.feat
DDI-DrugBank.d339.s0 DDI-DrugBank.d339.s0.e0 DDI-DrugBank.d339.s0.e1 null lib=elevated
wib=Elevated lpib=elevated_JJ la2=level wa2=levels lpa2=level_NNS la2=have wa2=have
lpa2=have_VBP la2=be wa2=been lpa2=be_VBN la2=report (...)  lpa2=concomitantly_RB path1=NNP
path2=NNP\dep\nsubjpass\compound path=NNP\dep\nsubjpass\compound
DDI-DrugBank.d339.s0 DDI-DrugBank.d339.s0.e0 DDI-DrugBank.d339.s0.e2 null lib=elevated
wib=Elevated lpib=elevated_JJ wib=experience lpib=experience_NN la2=be wa2=is lpa2=be_VBZ
la2=administer wa2=administered lpa2=administer_VBN la2=concomitantly wa2=concomitantly
lpa2=concomitantly_RB path1=NNP path2=NNP\dep\advcl\advcl\nsubjpass
path=NNP\dep\advcl\advcl\nsubjpass
DDI-DrugBank.d339.s0 DDI-DrugBank.d339.s0.e1 DDI-DrugBank.d339.s0.e2 mechanism
lb1=carbamazepine wb1=Carbamazepine wb1=Elevated lpb1=elevated_JJ lib=level wib=levels
lpib=level_NNS lib=have wib=have lpib=have_VBP lib=be wib=been lpib=be_VBN lib=report
wib=reported lpib=report_VBN lib=postmarket wib=postmarketing lpib=postmarket_VBG
lib=experience wib=experience lpib=experience_NN la2=be wa2=is lpa2=concomitantly_RB
path1=compound/nsubjpass/VBN path2=VBN\advcl\advcl\nsubjpass
path=compound/nsubjpass/VBN\advcl\advcl\nsubjpass
...
```

Feature extraction for NLP

- In most ML applications, the feature space is finite and known (e.g. credit scoring, medical diagnose prediction, churn prevention, fraud detection, etc).
- Also, most of the used features are numerical or categorial (income, age, sex, colestherol level, number of receipts returned, etc.)
- Thus, in these ML applications, feature vectors are usually *exhaustive* lists of pairs feature-value.

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Feature Extractor
Core task
Goals & Deliverables

Feature extraction for NLP

- In most ML applications, the feature space is finite and known (e.g. credit scoring, medical diagnose prediction, churn prevention, fraud detection, etc).
- Also, most of the used features are numerical or categorial (income, age, sex, colestherol level, number of receipts returned, etc.)
- Thus, in these ML applications, feature vectors are usually *exhaustive* lists of pairs feature-value.

BUT...

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Feature Extractor
Core task
Goals & Deliverables

Feature extraction for NLP

- In most ML applications, the feature space is finite and known (e.g. credit scoring, medical diagnose prediction, churn prevention, fraud detection, etc).
- Also, most of the used features are numerical or categorial (income, age, sex, colestherol level, number of receipts returned, etc.)
- Thus, in these ML applications, feature vectors are usually *exhaustive* lists of pairs feature-value.

BUT...

- In most NLP applications, features are related to appearing words, suffixes, prefixes, lemmas, etc.
- Thus, the feature space is huge.

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Feature Extractor
Core task
Goals & Deliverables

Feature extraction for NLP

- Example: We want to encode in a feature the word form for a token.
- We could use a feature (i.e. a single column) `word_form` with thousands of possible values.
- But no ML algorithm is able to deal with categorical features with that amount of possible values, and not enough data will ever be available to learn the behaviour of that feature.
- Thus, features in NLP are typically *binary-valued* (e.g. `word-form-is-THE`, `word-form-is-SOME`, `word-form-is-DOG`, etc.).
  That is, a column for each possible value, with boolean value.
- This creates feature spaces with tens or hundreds of thousands of features, but each feature has only two possible values. This is manageable by many ML algoritmhs.

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Feature Extractor
Core task
Goals & Deliverables

Feature extraction for NLP

- Thus, in NLP applications, feature vectors are usually *intensive* lists of strings (i.e. listing the names for features with value `true`, and ommiting those with value `false`), and are stored as *sparse vectors*.
- But, feature spaces of hundreds of thousands of dimensions can not be stored in a table or a matrix.
- However, feature vectors are usually very sparse: Each example has only a reduced number of non-zero columns (e.g. even there are thousands of columns like `word-form-is-XXX`, only one of them will be `true` for a given example).

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Feature Extractor
Core task
Goals & Deliverables

Sparse vector representation: CSR matrix

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Feature Extractor
Core task
Goals & Deliverables

- CSR matrix representation consists of storing the coordinates and the value for non-zero elements in the matrix.
- The matrix elements are indexed by their (row,column) coordinates.

$$\begin{pmatrix} 0 & 0 & 3 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 6 \end{pmatrix} \Longrightarrow \begin{pmatrix} row & col & val \\ 0 & 2 & 3 \\ 1 & 0 & 2 \\ 3 & 2 & 1 \\ 3 & 4 & 6 \end{pmatrix}$$

- Each row (of the full matrix) corresponds to a training example, and each column to a feature.
- The CSR matrix is a compressed representation of the same information.

- In the case of DDI, all features are binary (either 0 or 1) so the third column in CSR matrix may be ommited, saving more space.
- A feature index is needed to keep the correspondence between feature names and their column number.

Executing the feature extractor

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Feature Extractor
Core task
Goals & Deliverables

```
$ python3 run.py extract
```

Will process `train.xml` and `devel.xml` and produce `train.feat` and `devel.feat` in directory `predicted`.

If the function `extract_pair_features` is modified (to add or remove features), the datasets must be processed again.

Time to have a look to the `extract_pair_features` and to the `train.feat` file.

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Learner

- DDI task is a mere classification, not a sequence prediction.
- For a given sentence and pair of entities in it, the output is just **one** label, not a label per token as in NERC
- Sequence labeling algorithms such as CRFs are overdimensioned, and not straightforward to apply.
- Any classification ML algorithm may be used.
  - Maximum Entropy Classifiers (MEM)
  - Support Vector Machines (SVM)
  - ...

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Learner
Core task
Goals & Deliverables

Learner: MEM (a.k.a. Logistic Regression)

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Learner
Core task
Goals & Deliverables

- Install and import `scikit-learn`
  `$ pip install scikit-learn`
- Use provided `train.py` to learn a MEM model.
  `$ python3 run.py train MEM`

  You may modifiy learner parameters (loss function, thresholds, learning rates, etc).
- Check performance on development data.
  `$ python3 run.py predict MEM`
- `scikit-learn` implementation of `LogisticRegression` (MEM), admits sparse matrix representations such as CSR (Compressed Sparse Row) matrix.

Learner: SVM

- Install and import `scikit-learn`
  `$ pip install scikit-learn`
- Use provided `train.py` to learn a SVM model.
  `$ python3 run.py train SVM`

  You may modifiy learner parameters (loss function, thresholds, learning rates, etc).
- Check performance on development data.
  `$ python3 run.py predict SVM`
- `scikit-learn` implementation of `SVC` (SVM), admits sparse matrix representations such as CSR (Compressed Sparse Row) matrix.

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Learner
Core task
Goals & Deliverables

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Classifier

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Classifier
Core task
Goals & Deliverables

You can apply the learned models to new data and evaluate the results (stored in directory `results`):

```
$ python3 run.py predict MEM devel
$ python3 run.py predict SVM devel
$ python3 run.py predict CRF devel
```

Once a best configuration is selected experimenting with different features, algorithms, and hyperparameters, it can be applied to `test` data to evaluate generalization ability of the model.

```
$ python3 run.py extract test
$ python3 run.py predict MEM test
$ python3 run.py predict SVM test
$ python3 run.py predict CRF test
```

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Improve provided ML system

Machine Learning DDI

Relation Extraction

General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables

Starting point:

- Lemmas-in-between baseline reaches about 25% macro F1.
- Provided ML system reaches about 65%, depending on the algorithm

Task:

- Improve the performance of the provided system by:
  - Adding new features
  - Experimenting with different hyperparameters for the ML algorithms
  - Get insights from training partition. Use devel partition only to evaluate system behaviour.

The goal is not to reach the highest F1 score but to learn through experimenting with different strategies.

Choosing useful features

- Presence of certain *clue verbs* may be indicative of the interaction type.
- Clue verb position (before/inbetween/after) with respect to the target entities.
- Presence of other entities in between.
- Words, lemmas, PoS (or combinations of them) appearing before/inbetween/after the target pair.

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables

Choosing useful features

- Presence of certain *clue verbs* may be indicative of the interaction type.
- Clue verb position (before/inbetween/after) with respect to the target entities.
- Presence of other entities in between.
- Words, lemmas, PoS (or combinations of them) appearing before/inbetween/after the target pair.
- Features encoding information from the syntactic tree.

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables

Choosing useful features

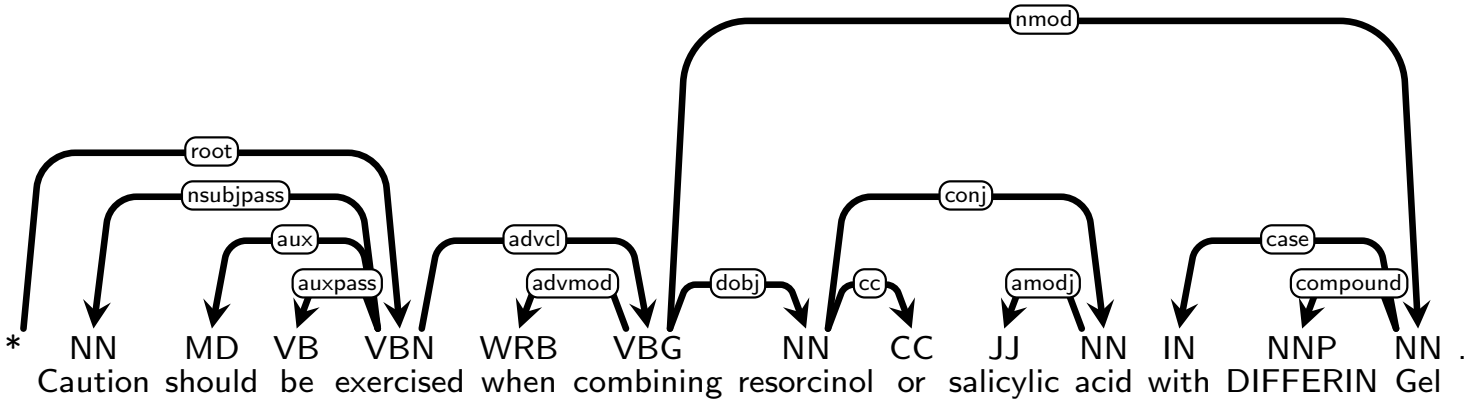
- Presence of certain *clue verbs* may be indicative of the interaction type.
- Clue verb position (before/inbetween/after) with respect to the target entities.
- Presence of other entities in between.
- Words, lemmas, PoS (or combinations of them) appearing before/inbetween/after the target pair.
- Features encoding information from the syntactic tree.

Remember: All features are *binary features* (they are either present or not present in a given example) and examples are encoded as *sparse vectors* (lists feature names present in the example)

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables

Choosing useful features

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables



Entities:

e0: `resorcinol` e1: `salicylic acid` e2: `DIFFERIN Gel`

Example path features:

PAIR (e0,e1)

Tree fragment: e0 $\overset{conj}{\rightarrow}$ e1

(e1 is direct child of e0. The arc is labeled *conj*)

Feature name: `path=conj>`

Choosing useful features

Machine Learning DDI

Relation Extraction

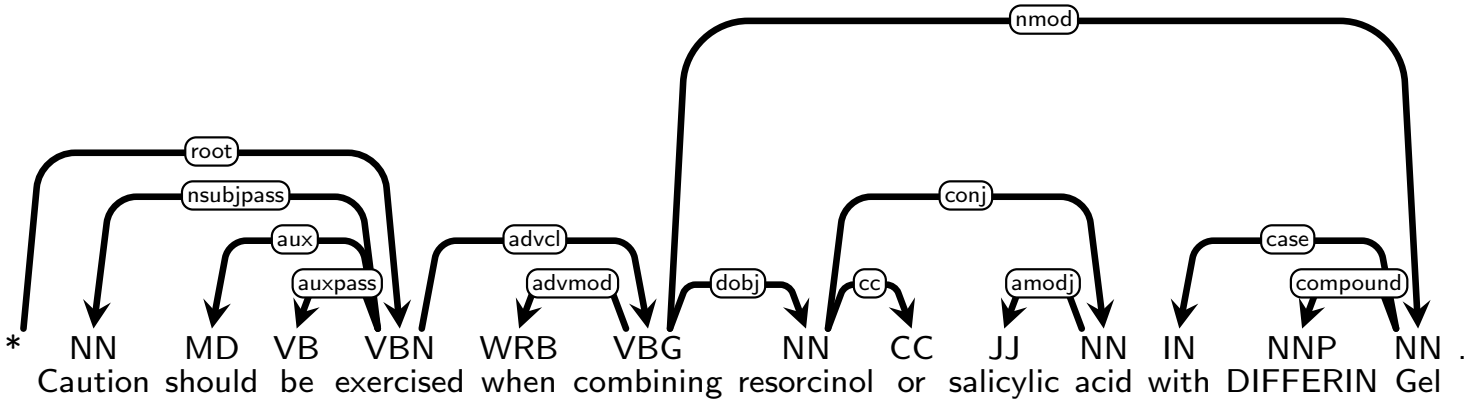
General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables



Entities:

e0: `resorcinol` e1: `salicylic acid` e2: `DIFFERIN Gel`

Example path features:

PAIR (e0,e2)

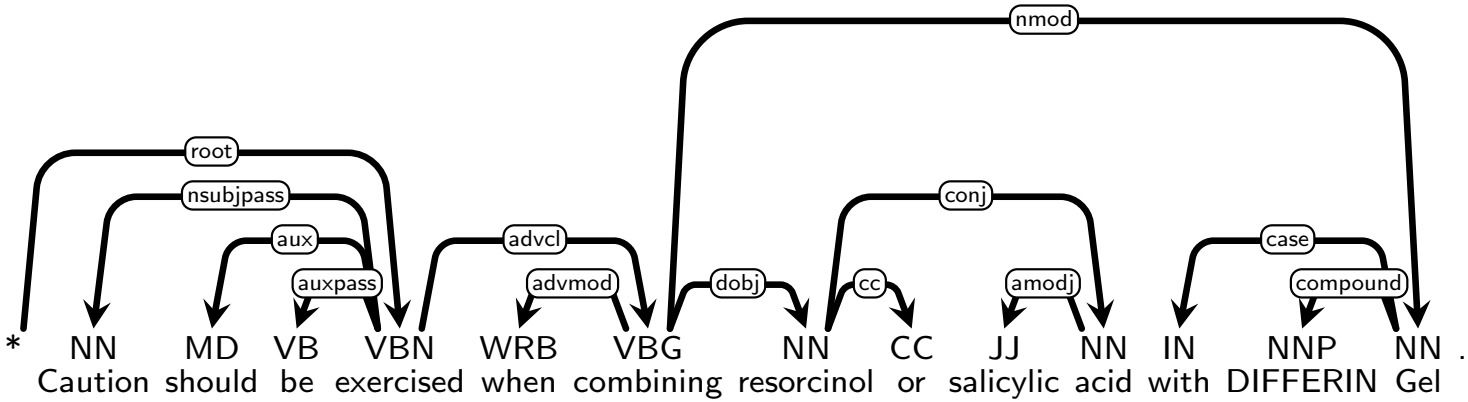
Tree fragment: e0 $\overset{dobj}{\leftarrow}$ combine $\overset{nmod}{\rightarrow}$ e2

(e0 is direct child of verb "combine" with label *dobj*, and e2 is direct child of the same verb, with label *nmod*)

Feature name: `path=dobj<combine>nmod`

Choosing useful features

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables



Entities:

e0: `resorcinol` e1: `salicylic acid` e2: `DIFFERIN Gel`

Example path features:

PAIR (e1,e2)

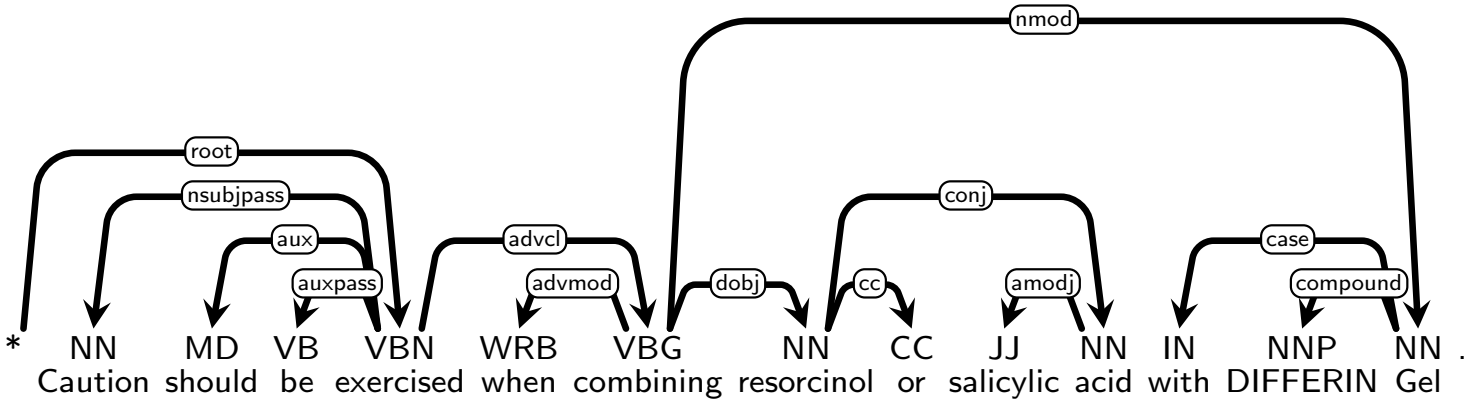
Tree fragment: e1 $\overset{conj}{\leftarrow}$ resorcinol $\overset{dobj}{\leftarrow}$ combine $\overset{nmod}{\rightarrow}$ e2

(e1 is *conj* child of "resorcinol", which is under verb "combine" with label *dobj*, and e2 is direct child of the same verb, with label *nmod*)

Feature name: `path=conj<dobj<combine>nmod`

Choosing useful features

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables



Entities:

e0: `resorcinol` e1: `salicylic acid` e2: `DIFFERIN Gel`

Example path features:

PAIR (e1,e2)

Tree fragment: e1 $\overset{conj}{\leftarrow}$ resorcinol $\overset{dobj}{\leftarrow}$ combine $\overset{nmod}{\rightarrow}$ e2
(e1 is *conj* child of "resorcinol", which is under verb "combine" with label *dobj*, and e2 is direct child of the same verb, with label *nmod*)

Also possible: `path=conj<ENTITY/dobj<combine>nmod`

Choosing useful features

Machine Learning DDI

Relation Extraction

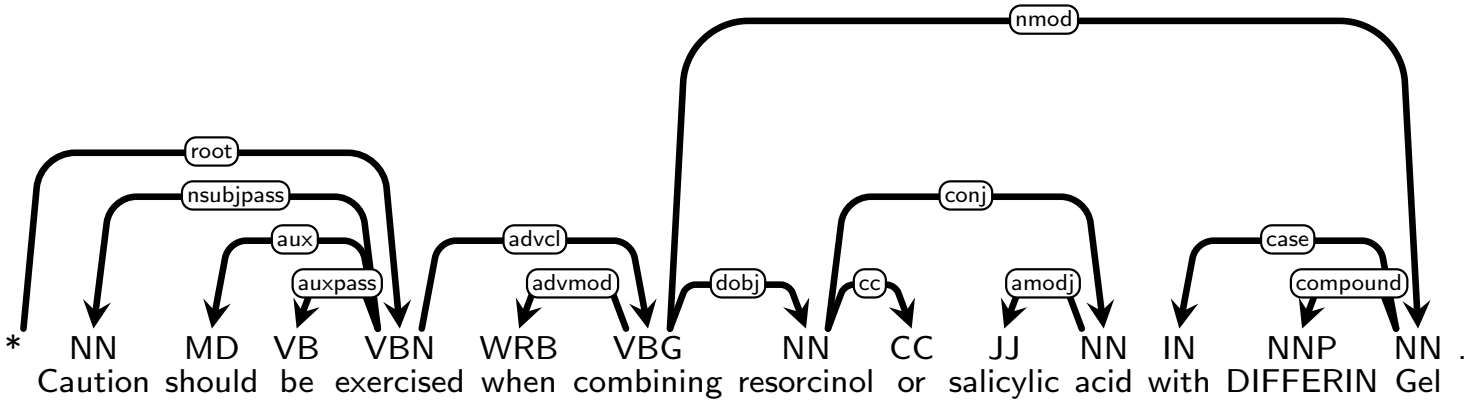
General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables



Entities:

e0: `resorcinol` e1: `salicylic acid` e2: `DIFFERIN Gel`

Example path features:

PAIR (e1,e2)

Tree fragment: e1 $\overset{conj}{\leftarrow}$ resorcinol $\overset{dobj}{\leftarrow}$ combine $\overset{nmod}{\rightarrow}$ e2

(e1 is *conj* child of "resorcinol", which is under verb "combine" with label *dobj*, and e2 is direct child of the same verb, with label *nmod*)

Also possible: `path=dobj*<combine>nmod`

Choosing useful features

Path features may be build in different ways, encoding different information about the tree

- Node words
- Node lemmas
- Node PoS
- Edge labels
- Edge direction
- Direct/indirect dependencies
- ... or any combination of these ...

Machine Learning DDI

Relation Extraction

General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables

Choosing useful features

Machine Learning DDI

Relation Extraction

General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables

Path features may be build in different ways, encoding different information about the tree

- Node words
- Node lemmas
- Node PoS
- Edge labels
- Edge direction
- Direct/indirect dependencies
- ... or any combination of these ...

Remember: All features are *binary features* (they are either present or not present in a given example) and examples are encoded as *sparse vectors* (lists of present feature names)

Choosing useful features

Some feature possibilities to explore:

- Features detailing the position of words in the sentece (e.g. before E1, between E1 and E2, after E2)
- More general (or more specific) path features (with/without all elements in the path, lemma/tag of each, etc)
- Features encoding specific tree patterns (and maybe their combination with certain verbs)
- Information about the LCS (PoS, lemma, ...)
- Features considering lists of relevant verbs.
- Type of entities in the pair
- Presence of a third entity (in the sentence, in between the target pair, in the path connecting the pair in the tree, ...)
- ...

Machine Learning DDI
Relation Extraction
General Structure
Resources
Detailed Structure
Core task
Goals & Deliverables

Outline

1 Machine Learning DDI

2 Relation Extraction

3 General Structure

4 Resources

5 Detailed Structure
- Feature Extractor
- Learner
- Classifier

6 Core task

7 Goals & Deliverables

Exercise Goals

Machine Learning DDI

Relation Extraction

General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables

What you should do:

- Extend your feature extractor. Pay special attention to features encoding syntactic information.
- Experiment with different algorithms and parameterizations.
- Keep track of tried features and parameter combinations, and results produced by each.

What you should **NOT** do:

- Use neural network learners. We'll do that later on the course.
- Get insights about errors from `devel` dataset. You have `train` for that. `devel` is only used to evaluate the performance of a given configuration.
- Select features or hyperparameters based on system performance on `test` dataset. You have `devel` for that. `test` is only used to evaluate model generalization ability once the best configuration has been chosen.

Deliverables

Machine Learning DDI

Relation Extraction

General Structure

Resources

Detailed Structure

Core task

Goals & Deliverables

At the end of the DDI task, you will need to deliver a single report on the work carried out on DDI-ML, DDI-NN, and DDI-LLM systems

So, during the development and experimentation on DDI-ML:

- keep track of tried/discarded features
- keep track of used algorithms and parameterizations.
- Record obtained results in the different experiments, and compile the information you'll later need to elaborate the report.