

UDeasy: a tool for extracting patterns from treebanks

ROLLING Seminars, Universitat Rovira i Virgili

Luca Brigada Villa

University of Pavia

May 22 2023



UNIVERSITÀ
DI PAVIA

1 Treebanks

What is a treebank

Why treebank are useful

2 Universal Dependencies

What is UD

Format

Annotation following UD guidelines

3 UDeasy

What is UDeasy

How to use UDeasy

4 Tutorial

is English an SVO language?

lemma *nice*

relative clauses

Catalan pro-drop?

The background features a stylized architectural design with two sets of columns supporting arches. The columns are light pink, and the arches are a slightly darker shade of pink. The word "Treebanks" is centered within the largest arch.

Treebanks

What is a treebank

A treebank is a **syntactically annotated resource**.

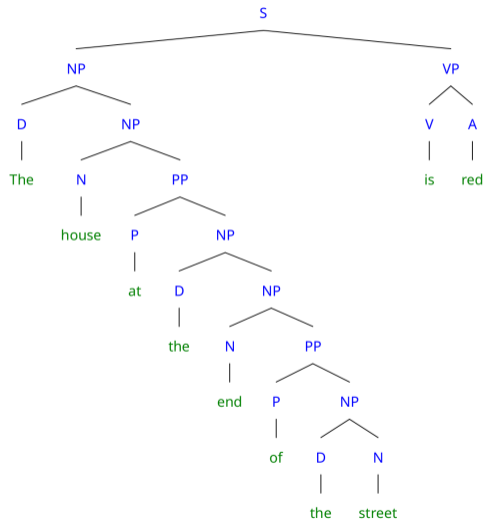
tree-bank → it contains (syntactic) trees

These trees can differ in terms of their structure according to the rules followed for their implementation. We can identify two main formalisms:

- constituency trees
- dependency trees

Depending on the formalism, the trees will have a different structure.

Example - constituency tree



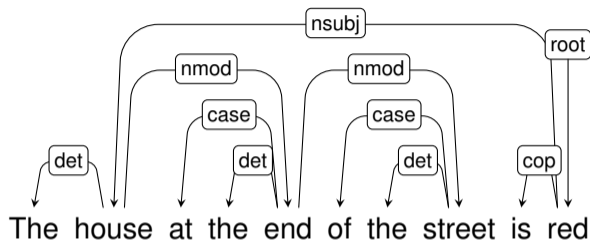
SENTENCE

The house at the end of the street is red

Things to be noticed:

- the elements of the sentence are grouped when they are contiguous
- a lot of empty nodes

Example - dependency tree



SENTENCE: The house at the end of the street is red

Things to be noticed:

- all the tokens (except for one) are connected to each other
- there are no empty nodes

Why treebanks are useful

Treebanks are used for many purposes:

- linguistic analysis:
 - synchronic
 - diachronic
- implementation of NLP models:
 - parsers
 - morphological analyzers
 - language models
 - (in the past) machine translation
- documenting languages

The background features a stylized architectural design with two sets of columns supporting arches. The central arch is the largest and contains the title text. The overall color palette is light pink and white.

Universal Dependencies

What is Universal Dependencies

Univesal Dependencies is a project started in 2014 that has many goals:

- to develop a set of consistent rules to annotate treebanks
- facilitate multilingual parser development
- facilitate linguistic research (typology in particular)

To do so, the UD community have developed an annotation scheme based on:

- Stanford dependencies (syntax)
- the Google universal part-of-speech tagset
- the Interset interlingua tagset (morphological annotation)

Format of UD treebanks

Treebanks are text files that can be opened using a text editor:



Treebanks are formatted in **CoNLL-U**, a format in which:

- tokens and their annotation are stored in a single line
- sentences are separated by a blank line

A token in the CoNLL-U format has ten fields (columns), separated by a “tab” character:

- 1 `id`: the ID of the word, it must be unique within the sentence. Each word must have an ID (starting ID: 1).
- 2 `form`: the form in which the word appears in the sentence.
- 3 `lemma`: the “dictionary entry” of the word.
- 4 `upos`: the universal part-of-speech of the word.
- 5 `xpos`: the specific part-of-speech of the word.
- 6 `feats`: the morphological features of the word (gender, number, case, mood, tense...).
- 7 `head`: the id of the token from which it syntactically depends.
- 8 `deprel`: the type of syntactic dependency between the word and its head.
- 9 `deps`: the complete syntactic dependencies of the word (format `head:deprel`).
- 10 `misc`: any additional information.

Some clarifications on the format of certain fields

- *multiword tokens*: the basic units of annotation are *syntactic words*. Such words are systematically splitted (articled prepositions, verbs with clitics...). To annotate them properly, we follow these rules:
 - the `id` of the multiword token appears in this form: `PRIMO-ULTIMO`
 - all fields except for the `id` and the `form` remain empty in the multiword token (we can't keep a field completely empty, so we use an *underscore* (`_`))
 - in the following lines we annotate the tokens in the multiword token specifying their `ids`
- the fields `feats` and `misc`:
 - are annotated using a set of *key-value* pairs separated by a *pipe* character (`|`)
 - both the *key* and the *value* should be formatted using `CamelCase` (not `dromedaryCase`, neither `UPPERCASE`, nor `lowercase`, or `snake_case` or `kebab-case`)

Example - sentence formatted in CoNLL-U

```
# text = The house at the end of the street is red
# sent_id = 0
1  The  the  DET  DT  Definite=Def|PronType=Art  2  det  _  _
2  house house NOUN NN  Number=Sing  10  nsubj  _  _
3  at   at   ADP  IN  _  5  case  _  _
4  the  the  DET  DT  Definite=Def|PronType=Art  5  det  _  _
5  end  end  NOUN NN  Number=Sing  2  nmod  _  _
6  of   of   ADP  IN  _  8  case  _  _
7  the  the  DET  DT  Definite=Def|PronType=Art  8  det  _  _
8  street street NOUN NN  Number=Sing  5  nmod  _  _
9  is   be   AUX  VBZ  Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=Fin  10  cop  _  _
10 red  red  ADJ  JJ  Degree=Pos  0  root  _  _
```

Annotate following UD guidelines: `upos`

The third field of a token properly formatted stores the annotation of universal parts-of-speech.

The complete POS tagset can be found **here**. It includes a limited set of tags divided in three categories:

- *open class words*: ADJ, ADV, INTJ, NOUN, PROPN, VERB
- *closed class words*: ADP, AUX, CCONJ, DET, NUM, PART, PRON, SCONJ
- *other*: PUNCT, SYM, X

Annotate following UD guidelines: feats

The sixth field of a token properly formatted stores the annotation of the morphological features.

The list of keys with their values can be found **here**. It includes a limited set of features divided in two main groups:

- *lexical features*: PronType, NumType, Poss, Reflex, Foreign, Abbr, Typo
- *inflectional features*:
 - *nominal*: Gender, Animacy, NounClass, Number, Case, Definite, Degree
 - *verbal*: VerbForm, Mood, Tense, Aspect, Voice, Evident, Polarity, Person, Polite, Clusivity

Annotate following UD guidelines: structure

To annotate the **syntactic structure** of the sentences following UD guidelines, we have to keep in mind some basic principles:

- *content words* are the backbone of the syntactic structure
- *function words* in most cases depend from *content words*
- modifiers depend from modified words

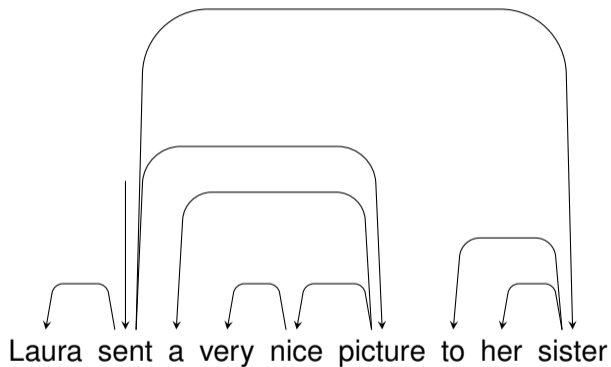
Let's try to annotate the sentence "Laura sent a very nice picture to her sister".

Annotate following UD guidelines: structure

To annotate a sentence like “Laura sent a very nice picture to her sister”:

- 1 we have to identify content words: “Laura”, “sent”, “picture”, “nice”, “sister”
- 2 per ognuna di esse, identifichiamo la head:
 - “Laura” ← “sent”
 - “picture” ← “sent”
 - “nice” ← “picture”
 - “sister” ← “sent”
- 3 for each of the remaining words (*function words*), we identify the head:
 - “a” ← “picture”
 - “very” ← “nice”
 - “to” ← “sister”
 - “her” ← “sister”

Annotate following UD guidelines: structure



Annotate following UD guidelines: syntactic labels

Once we have structured the syntax of the sentence, we can move to the next phase which consists in labeling the syntactic links. To do so, we can choose among the labels in the **UD set**.

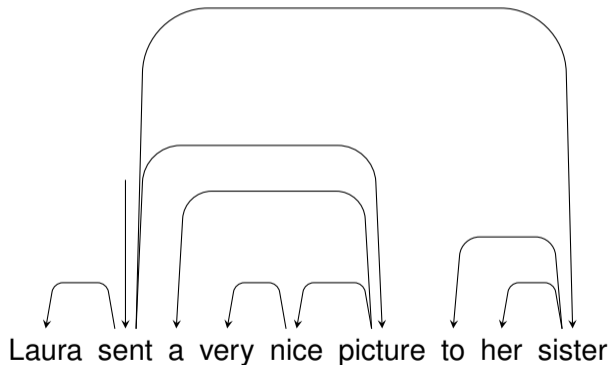
For our sentence, we will choose these ones:

- `root`: this label is assigned to the root of the sentence; the token whose `deprel` is `root` does not have a `head`
- `nsubj`: stands for “nominal subject”; it is assigned to the nominal syntactic subject of the sentence
- `obj`: stands for “object”; it is assigned to the second most important argument after the subject
- `iobj`: stands for “indirect object”; it is assigned to some verb arguments (*always core arguments*). This label is often used to tag the *recipient*
- `det`: stands for “determiner”; it is used to tag the link between a nominal head and its determiner
- `case`: it is used to tag the relation between a head and a *function word* which marks the case (prepositions, postpositions...)
- `amod`: stands for “adjectival modifier”; it is assigned to the adjective or to the adjectival phrase which modifies a noun or a pronoun
- `advmod`: stands for “adverbial modifier”; it is assigned to the adverbs or adverbial phrases that modify a predicate or a modifier
- `nmod`: stands for “nominal modifier”; this relation is used for nominal dependents of another noun or noun phrase. It takes the sublabel `poss` for possessives

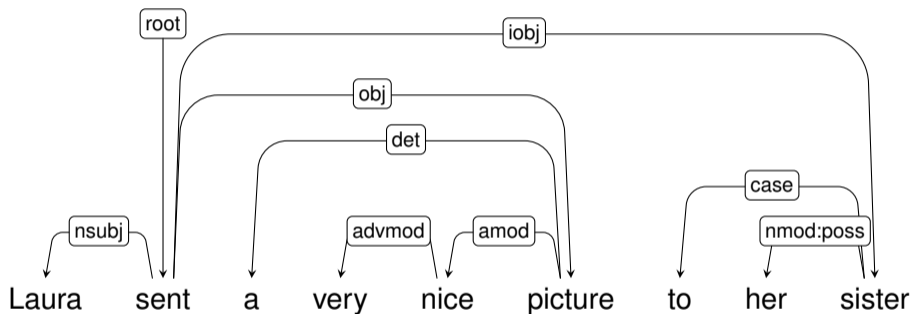
Annotate following UD guidelines: syntactic labels

Proviamo ad assegnare queste etichette alle relazioni sintattiche della frase di prima:

root nsubj obj iobj nmod
det case amod advmod



Annotate following UD guidelines: syntactic labels



The background features a stylized architectural design with two sets of columns supporting arches. The columns are light red, and the arches are a slightly darker shade of red. The text 'UDeasy' is centered within the largest arch.

UDeasy

What is UDeasy

UDeasy is a tool that allows to query and extract occurrences from the treebanks. It has a graphical interface which makes this operation the easiest as possible.

UDeasy is available for these OS:

- Ubuntu
- MacOS
- Windows

Download

Before starting using UDeasy, users should download the latest version of the software compatible their own OS. The download page is:

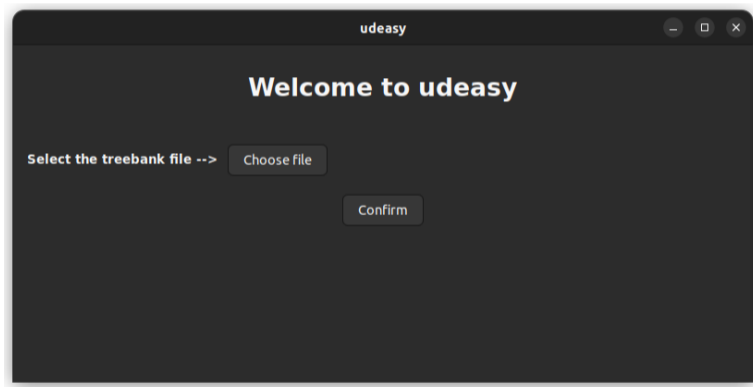
<https://unipv-larl.github.io/udeasy/download.html>.

Install

Then we can install UDeasy. Once installed, we can start UDeasy.

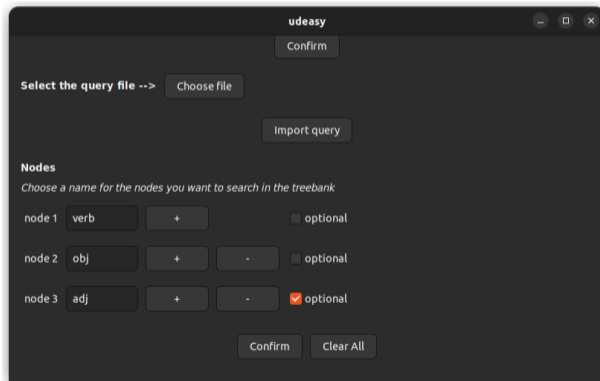
Using UDeasy

When opening UDeasy, a window will pop up. The users will be asked to select a conllu file from which extract the patterns.



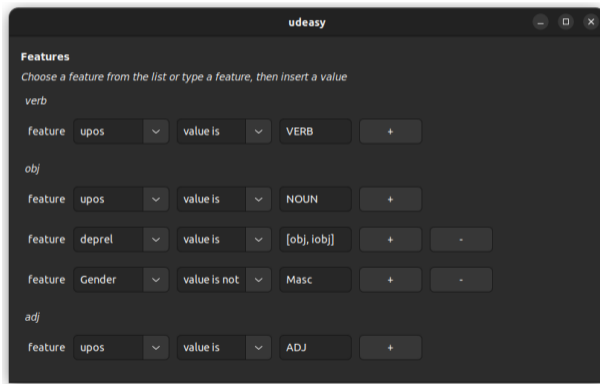
Node names

Once selected the conllu file, two sections will appear in the main window: the first is to import a query, the second to start designing a new one from scratch.



Features

Then we can select the features that the target tokens must (or must not) match in order to be included in the results.



Syntactic relations and ordering within the sentence

At the bottom of the window, users can specify the syntactic relations that the target nodes should have and the relative positions they should occupy in the sentence.

The screenshot shows a window titled "udeasy" with a dark background. It contains two main sections: "Relations" and "Positions".

Relations

- relation 1: verb (dropdown), is parent of (dropdown), obj (dropdown), + (button)
- relation 2: obj (dropdown), is parent of (dropdown), adj (dropdown), + (button), - (button)

Positions

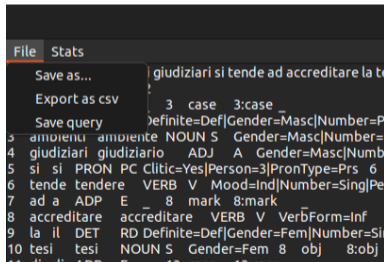
- verb (dropdown), precedes (dropdown), obj (dropdown), (empty dropdown), (empty dropdown), positions, + (button)

Visualize: conllu sentences conllu matched nodes trees

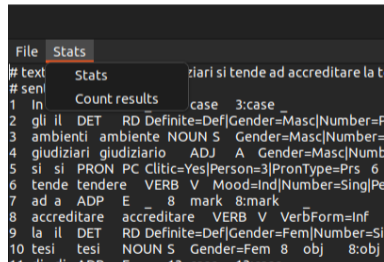
Submit query Clear All

Working with results

After confirming the query, the results will appear in a new window. Results can be saved either as simple text files or as csv file (that can be imported in softwares like Excel) selecting the fields that we want to be included. It is also possible to save the query as YAML file and extract some statistics of the results.



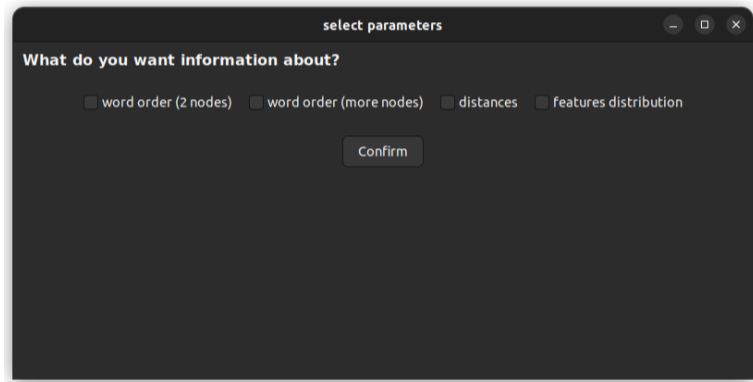
```
File  Stats
Save as...
Export as csv
Save query
5 ambienti ambiente NOUN S Gender=Masc|Number=
4 giudiziari giudiziario ADJ A Gender=Masc|Numb
5 si si PRON PC Clitic=Yes|Person=3|PronType=Prs 6
6 tende tendere VERB V Mood=Ind|Number=Sing|Pe
7 ad a ADP E _ 8 mark 8:mark
8 accreditare accreditare VERB V VerbForm=Inf
9 la il DET RD Definite=Def|Gender=Fem|Number=Si
10 tesi tesi NOUN S Gender=Fem 8 obj 8:obj
```



```
File  Stats
# text
# sen
1 In
2 gli il DET RD Definite=Def|Gender=Masc|Number=P
3 ambienti ambiente NOUN S Gender=Masc|Number=
4 giudiziari giudiziario ADJ A Gender=Masc|Numb
5 si si PRON PC Clitic=Yes|Person=3|PronType=Prs 6
6 tende tendere VERB V Mood=Ind|Number=Sing|Pe
7 ad a ADP E _ 8 mark 8:mark
8 accreditare accreditare VERB V VerbForm=Inf
9 la il DET RD Definite=Def|Gender=Fem|Number=Si
10 tesi tesi NOUN S Gender=Fem 8 obj 8:obj
```

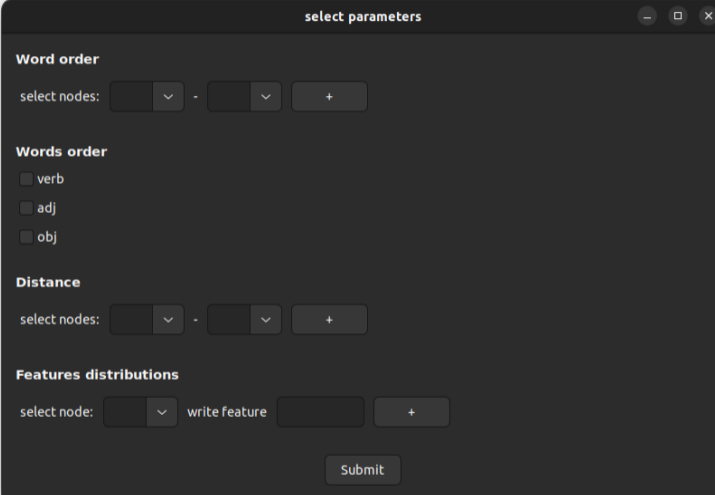
Extract statistics

UDeasy allows to extract statistical information from the results. To do so, we have to select what we want to get from the available options.



Extract statistics

Once selected, we will fill in the fields to get information about frequencies/ordering/cooccurrences of the selected tokens.



The image shows a dark-themed dialog box titled "select parameters" with standard window controls (minimize, maximize, close) in the top right corner. The dialog is organized into four sections:

- Word order:** Labeled "select nodes:", it contains two dropdown menus, a minus sign "-", and a plus sign "+" button.
- Words order:** Contains three unchecked checkboxes labeled "verb", "adj", and "obj".
- Distance:** Labeled "select nodes:", it contains two dropdown menus, a minus sign "-", and a plus sign "+" button.
- Features distributions:** Labeled "select node:", it contains a dropdown menu, the text "write feature", a text input field, and a plus sign "+" button.

A "Submit" button is located at the bottom center of the dialog.

The background features a light pink color scheme with stylized architectural elements. On the left and right sides, there are two sets of columns supporting a horizontal beam, with a large arch above each set. The word "Tutorial" is centered within the arch on the left.

Tutorial

In this tutorial we will deal with the English GUM treebank (9124 sentences, 166659 tokens) and a sample of the Catalan Ancora treebank (9000 sentences, 307555 tokens) both available in Universal Dependencies. Questions:

- 1 is English an SVO language?
- 2 which words are modified by the lemma *nice*?
- 3 does English use relative clauses more on the subject or on the object?
 - how distant is the relative clause from its head?
- 4 is Catalan a pro-drop language?

The files for this tutorial are available here:

<https://unipv-larl.github.io/udeasy/tutorials.html>.

Is English an SVO language?

To answer this question we first have to design a query targeting some patterns from the treebank and then analyze the ordering of the elements within such patterns.

- we are looking for three nodes and we will refer to them as *subj*, *verb* and *obj*
- we restrict the results to the nominal subjects and objects (we exclude pronouns)
- we specify the dependency relations
- and the syntactic relations among the target nodes
- we choose a visualization option

Which words are modified by the lemma *nice*?

To answer this question we first have to design a query targeting some patterns from the treebank and then analyze some features of such elements.

- we are looking for two nodes and we'll refer to them as *word* and *nice*
- for the node *word* we don't specify any feature
- for the node *nice* we specify the lemma
- syntactic relation between the two nodes
- we choose a visualization option

Does English use relative clauses more on the subject or on the object?

To answer this question we have to design a query that meets these criteria and then analyze the features of one of the nodes involved in the patterns:

- two nodes (we'll refer to them as *subj-obj* and *rel*)
- we specify the features of the node *rel* (`deprel`)
- and of the node *subj-obj* (`upos` and `deprel`)
- we define the syntactic structure
- we choose a visualization option

Is Catalan a pro-drop language?

To answer this question we first have to design a query in which one of the target nodes is optional:

- we are looking for two nodes: we will refer to them as *subj* (optional) and *verb*
- we specify the features of the nodes (*deprel*, *upos*, *VerbForm*, *PronType*)
- we define the syntactic relations
- we choose a visualization option

Thank you for your attention!

✉ luca.brigadavilla@unibg.it

👤 bavagliladri

👤 unipv-larl

🔗 UDeasy website