cost
EUROPEAN COOPERATION
IN SCIENCE & TECHNOLOGY

Funded by
the European Union

UniDive Training School Course 3.3

# Corpus Format Validation

UniDive

Daniel Zeman, Agata Savary

zeman@ufal.mff.cuni.cz
https://unidive.lisn.upsaclay.fr/

# Outline

1. File formats (CoNLL-U, CUPT)
2. CoNLL-U validator
3. PARSEME validator
4. UD/PARSEME consistency

# CoNLL-U Format

```
# sent_id = s1
# text = Es unterscheidet sich vom westlichen Teil des Landes.
ID   FORM           LEMMA          UPOS   XPOS  FEATS  HEAD  DEPREL   DEPS  MISC
1    Es             es             PRON   _     _      2     nsubj    _     _
2    unterscheidet  unterscheiden  VERB   _     _      0     root     _     _
3    sich           sich           PRON   _     _      2     expl:pv  _     _
4-5  vom            _              _      _     _      _     _        _     _
4    von            von            ADP    _     _      7     case     _     _
5    dem            der            DET    _     _      7     det      _     _
6    westlichen     westlich       ADJ    _     _      7     amod     _     _
7    Teil           Teil           NOUN   _     _      2     obl      _     _
8    des            der            DET    _     _      9     det      _     _
9    Landes         Land           NOUN   _     _      7     nmod     _     SpaceAfter=No
10   .              .              PUNCT  _     _      2     punct    _     _
```

- CoNLL = *Conference on Natural Language Learning* (they organize shared tasks as satellite events)
- CoNLL* formats typically look like tables

# CoNLL-U Format

```
# sent_id = s1
# text = Es unterscheidet sich vom westlichen Teil des Landes.
ID   FORM          LEMMA          UPOS   XPOS  FEATS  HEAD  DEPREL   DEPS  MISC
1    Es            es             PRON   _     _      2     nsubj    _     _
2    unterscheidet unterscheiden  VERB   _     _      0     root     _     _
3                                        _     _      2     expl:pv  _     _
4-5  vom           _              _      _     _      _     _        _     _
4    von           von            ADP    _     _      7     case     _     _
5    dem           der            DET    _     _      7     det      _
6    westlichen    westlich       ADJ    _     _      7     amod
7    Teil          Teil           NOUN   _     _      2     obl      _     _
8    des           der            DET    _     _      9     det      _     _
                   Land           NOUN   _     _      7     nmod     _     SpaceAfter=No
10                 .              PUNCT  _     _      2     punct    _     _
```

The headers are not part of the file

Use TAB to separate columns

LF line breaks

Empty line

- CoNLL = *Conference on Natural Language Learning* (they organize shared tasks as satellite events)
- CoNLL* formats typically look like tables

# CoNLL-U Format

```
# sent_id = s1
# text = Es unterscheidet sich vom westlichen Teil des Landes.
# text_en = It is different from the western part of the country.
1    Es             es             PRON   _  Case=Nom|Gender=Neut|Number=Sing|Person=3|PronType=
2    unterscheidet  unterscheiden  VERB   _  Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=F
3    sich           sich           PRON   _  Case=Acc|Person=3|PronType=Prs|Reflex=Yes
4-5  vom            _              _      _  _
4    von            von            ADP    _  _
5    dem            der            DET    _  Case=Dat|Definite=Def|Gender=Masc|Number=Sing|PronT
6    westlichen     westlich       ADJ    _  Case=Dat|Degree=Pos|Gender=Masc|Number=Sing
7    Teil           Teil           NOUN   _  Case=Dat|Gender=Masc|Number=Sing
8    des            der            DET    _  Case=Gen|Definite=Def|Gender=Neut|Number=Sing|PronT
9    Landes         Land           NOUN   _  Case=Gen|Gender=Neut|Number=Sing
10   .              .              PUNCT  _  _
```

- CoNLL-U file format specification:
  - https://universaldependencies.org/format.html

# Official Validator

- A Python script you can run locally on your computer
  - If you have the Python language installed

# Official Validator

- A Python script you can run locally on your computer
  - If you have the Python language installed
- It will be run automatically on the server whenever you push changes to the `dev` branch of your repo
  - `https://quest.ms.mff.cuni.cz/udvalidator/cgi-bin/unidep/validation-report.pl`

# Official Validator

- A Python script you can run locally on your computer
  - If you have the Python language installed
- It will be run automatically on the server whenever you push changes to the `dev` branch of your repo
  - `https://quest.ms.mff.cuni.cz/udvalidator/cgi-bin/unidep/validation-report.pl`

- **Always check the online validation results** even if you validate locally
  - Online validation runs other scripts to check for more errors
  - The main script may have a newer version online
  - Data that don't pass online validation **will not be released!**
    - (Being visible in `dev` branch on GitHub does not mean being released!)

# Running Locally (Optional)

- Get Python
  - Linux / Mac: You may already have it
    - In Bash, try running `which python` or `which python3`
    - You need Python 3+, not Python 2 (try running `python --version`)
  - Windows: Google “download python windows”
    - `https://www.python.org/downloads/windows/`
    - Download and install it

# Running Locally (Optional)

- Get Python
  - Linux / Mac: You may already have it
    - In Bash, try running `which python` or `which python3`
    - You need Python 3+, not Python 2 (try running `python --version`)
  - Windows: Google “download python windows”
    - `https://www.python.org/downloads/windows/`
    - Download and install it
- Get one additional “library” (= extra functionality) for Python
  - `python -m pip install regex`

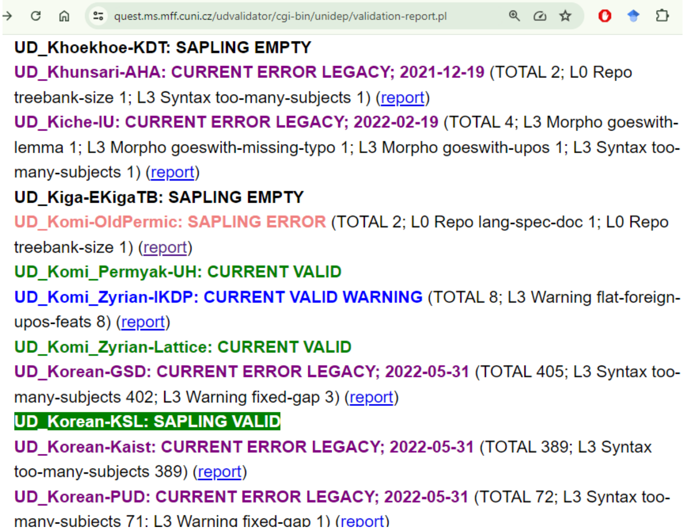
# Running Locally (Optional)

- Get Python
  - Linux / Mac: You may already have it
    - In Bash, try running `which python` or `which python3`
    - You need Python 3+, not Python 2 (try running `python --version`)
  - Windows: Google “download python windows”
    - `https://www.python.org/downloads/windows/`
    - Download and install it
- Get one additional “library” (= extra functionality) for Python
  - `python -m pip install regex`
- Get the validator
  - Clone the tools repository from UD GitHub
  - In that repo, look for `validate.py`
  - Run `python /path/to/tools/validate.py --help`

# Running on Server (Automatic)

- GitHub is set up to run it when you push to `dev`
- Small treebanks: Results available instantly
- Large treebanks: You may have to wait up to 20 minutes

- Write to Dan if you believe it’s not working


quest.ms.mff.cuni.cz/udvalidator/cgi-bin/unidep/validation-report.pl

**UD_Khoekhoe-KDT: SAPLING EMPTY**
**UD_Khunsari-AHA: CURRENT ERROR LEGACY; 2021-12-19** (TOTAL 2; L0 Repo treebank-size 1; L3 Syntax too-many-subjects 1) (report)
**UD_Kiche-IU: CURRENT ERROR LEGACY; 2022-02-19** (TOTAL 4; L3 Morpho goeswith-lemma 1; L3 Morpho goeswith-missing-typo 1; L3 Morpho goeswith-upos 1; L3 Syntax too-many-subjects 1) (report)
**UD_Kiga-EKigaTB: SAPLING EMPTY**
**UD_Komi-OldPermic: SAPLING ERROR** (TOTAL 2; L0 Repo lang-spec-doc 1; L0 Repo treebank-size 1) (report)
**UD_Komi_Permyak-UH: CURRENT VALID**
**UD_Komi_Zyrian-IKDP: CURRENT VALID WARNING** (TOTAL 8; L3 Warning flat-foreign-upos-feats 8) (report)
**UD_Komi_Zyrian-Lattice: CURRENT VALID**
**UD_Korean-GSD: CURRENT ERROR LEGACY; 2022-05-31** (TOTAL 405; L3 Syntax too-many-subjects 402; L3 Warning fixed-gap 3) (report)
**UD_Korean-KSL: SAPLING VALID**
**UD_Korean-Kaist: CURRENT ERROR LEGACY; 2022-05-31** (TOTAL 389; L3 Syntax too-many-subjects 389) (report)
**UD_Korean-PUD: CURRENT ERROR LEGACY; 2022-05-31** (TOTAL 72; L3 Syntax too-many-subjects 71; L3 Warning fixed-gap 1) (report)

# Levels of Validation

- **Level 1:** Technical backbone
  - 10 columns, empty line after each sentence, Unicode normalization…
  - ID column: correct sequence of numbers
- **Level 2:** UD basics
  - UPOS column: one of the 17 known tags
  - FEATS, DEPREL: *looks like* a UD label
  - HEAD: valid numbers, no cycles
  - Every sentence has `# sent_id` and `# text` metadata
- **Level 3:** UD guidelines
  - E.g., `conj` and its subtypes must go left to right
- **Level 4:** Language-specific basics
  - Possible exceptions for “words with spaces”
- **Level 5:** Language-specific guidelines
  - E.g. list of approved lemmas of auxiliaries and copulas

- Level 6:
  - Optional, on demand only: validation of entities and coreference in MISC

# Levels of Validation

- **Level 1:** Technical backbone
  - 10 columns, empty line after each sentence, Unicode normalization…
  - ID column: correct sequence of numbers
- **Level 2:** UD basics
  - UPOS column: one of the 17 known tags
  - FEATS, DEPREL: *looks like* a UD label
  - HEAD: valid numbers, no cycles
  - Every sentence has `# sent_id` and `# text` metadata
- **Level 3:** UD guidelines
  - E.g., `conj` and its subtypes must go left to right
- **Level 4:** Language-specific basics
  - Possible exceptions for “words with spaces”
- **Level 5:** Language-specific guidelines
  - E.g. list of approved lemmas of auxiliaries and copulas

- **Level 0** (online only, separate script):
  - Contents of your repo, file naming, README file
  - Is the language documented on the UD website?

# Mandatory vs. Optional

- To be released, treebank must validate on levels 0–5!

# Mandatory vs. Optional

- To be released, treebank must validate on levels 0–5!
- Some messages are warnings, not errors (they do not block releasing)

# More Columns / Fewer Columns: CoNLL-U Plus

```
# global.columns = ID FORM UPOS HEAD DEPREL MISC PARSEME:MWE
# source_sent_id = conllu http://hdl.handle.net/11234/1-2837 UD_German-GSD/de_gsd-ud-train.conll
# sent_id = train-s1682
# text = Der CDU-Politiker strebt einen einheitlichen Wohnungsmarkt an, auf dem sich die Preise
1   Der            DET    2  det           _              *
2   CDU            PROPN  4  compound      SpaceAfter=No  *
3   -              PUNCT  2  punct         SpaceAfter=No  *
4   Politiker      NOUN   5  nsubj         _              *
5   strebt         VERB   0  root          _              2:VPC.full
6   einen          DET    8  det           _              *
7   einheitlichen  ADJ    8  amod          _              *
8   Wohnungsmarkt  NOUN   5  obj           _              *
9   an             ADP    5  compound:prt  SpaceAfter=No  2
10  ,              PUNCT  5  punct         _              *
…
```

- CoNLL-U Plus file format specification:
  - https://universaldependencies.org/ext-format.html

# CUPT format: the PARSEME instance of CoNLL-U Plus

```
# global.columns = ID FORM UPOS HEAD DEPREL MISC PARSEME:MWE
# source_sent_id = http://hdl.handle.net/11234/1-2837 UD_German-GSD/de_gsd-ud-train.conllu trai
# text = Der CDU-Politiker strebt einen einheitlichen Wohnungsmarkt an, auf dem sich die Preise
1   Der            DET    2  det           _              *
2   CDU            PROPN  4  compound      SpaceAfter=No  *
3   -              PUNCT  2  punct         SpaceAfter=No  *
4   Politiker      NOUN   5  nsubj         _              *
5   strebt         VERB   0  root          _              2:VPC.full
6   einen          DET    8  det           _              *
7   einheitlichen  ADJ    8  amod          _              *
8   Wohnungsmarkt  NOUN   5  obj           _              *
9   an             ADP    5  compound:prt  SpaceAfter=No  2
10  ,              PUNCT  5  punct         _              *
…
```

- CUPT file format specification:
  - ▸ https://multiword.sourceforge.net/PHITE.php?sitesig=CONF&page=CONF_04_LAW-MWE-CxG_2018___lb__COLING__rb__&subpage=CONF_45_Format_specification

# CUPT format validator

- Uses the **CoNLL validator** to check UD compatibility
- Like for CoNLL-U, there is a **2-level validation**
  - **Level 1**:
    - checks lines and columns but not their contents
    - checks the existence of `global.columns` in the first line
    - invokes the CoNLLL-U validator level 1
  - **Level 2**:
    - invokes the CoNLL-U validator to check morphosyntax
    - checks the 11th column for integrity
- Available at `https://gitlab.com/parseme/utilities` under `st-organizers/release-preparation/CI-CD/`**`parseme_validate.py`**
- Documented at `https://gitlab.com/parseme/corpora/-/wikis/PARSEME-tools#file-format-validation`

# CUPT format validation exercise

- The CUPT validator can only be used in **command line** or behind the git of your PARSEME repository
- To test it in **command line**
  - finish the Gitlab exerices from lecture 1 (until push)
  - do this exercise: `https://gitlab.com/parseme/unidive-training-school#exercise-for-lecture-3-parseme-validator`

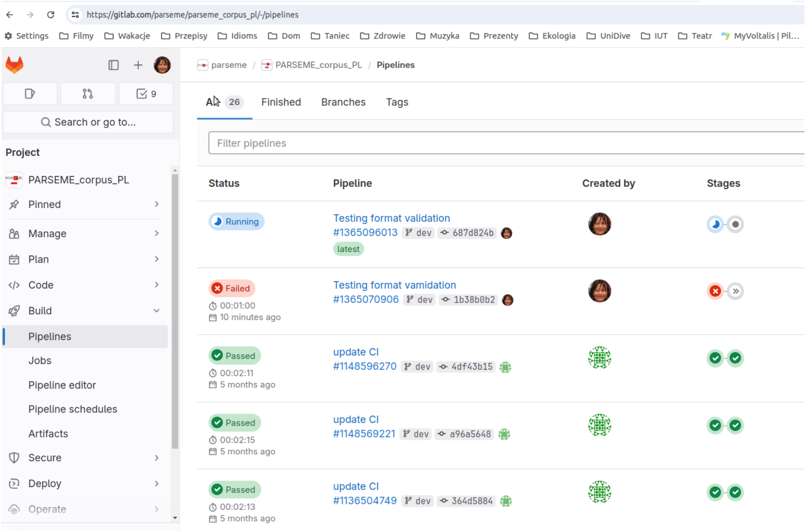
# CUPT format validation behind Gitlab

- The CUPT validator is installed in each **PARSEME language repository**
- If you **push** to the **development branch** (`dev`), this validator is run automatically on the server in a **pipeline**
- The **outcome of the pipeline** is visible on the server and is sent by email to the user
- This mechanism is called **continuous integration** (part of CI/CD: continuous integration/continuous deployment)

# CUPT format validation behind Gitlab – a demo

- We go to a local Polish PARSEME repository
- We switch to the `dev` branch
- We edit a file so that its **format** becomes **invalid**
- We add, commit and **push**
- We observe the pipeline in the browser at check which stages failed
- We correct the files, add, commit and push again

# CUPT format validation behind Gitlab – pipelines on the server

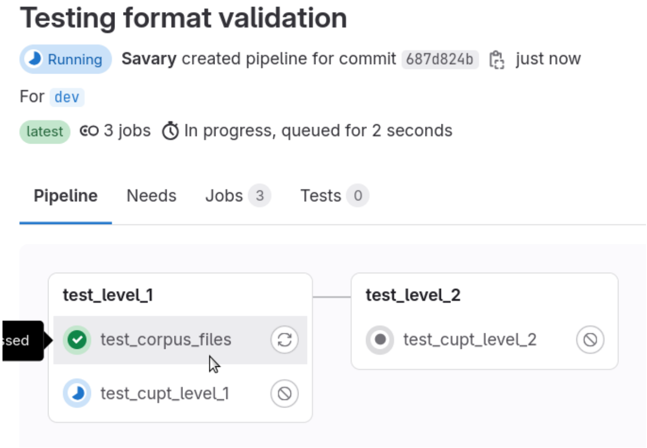


https://gitlab.com/parseme/parseme_corpus_pl/-/pipelines

parseme / PARSEME_corpus_PL / Pipelines

| Status | Pipeline | Created by | Stages |
|---|---|---|---|
| Running | Testing format validation<br>#1365096013 dev 687d824b<br>latest | | |
| Failed<br>00:01:00<br>10 minutes ago | Testing format vamidation<br>#1365070906 dev 1b38b0b2 | | |
| Passed<br>00:02:11<br>5 months ago | update CI<br>#1148596270 dev 4df43b15 | | ✓ ✓ |
| Passed<br>00:02:15<br>5 months ago | update CI<br>#1148569221 dev a96a5648 | | ✓ ✓ |
| Passed<br>00:02:13<br>5 months ago | update CI<br>#1136504749 dev 364d5884 | | ✓ ✓ |

# CUPT format validation behind Gitlab – a pipeline is running



Testing format validation

Running **Savary** created pipeline for commit `687d824b` just now

For dev

latest 3 jobs In progress, queued for 2 seconds

Pipeline Needs Jobs 3 Tests 0

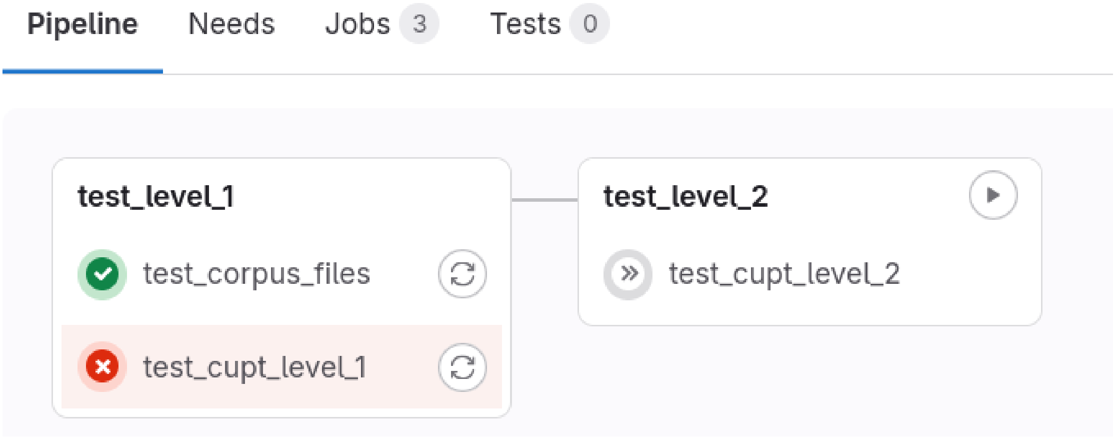
test_level_1

- ✓ test_corpus_files
- test_cupt_level_1

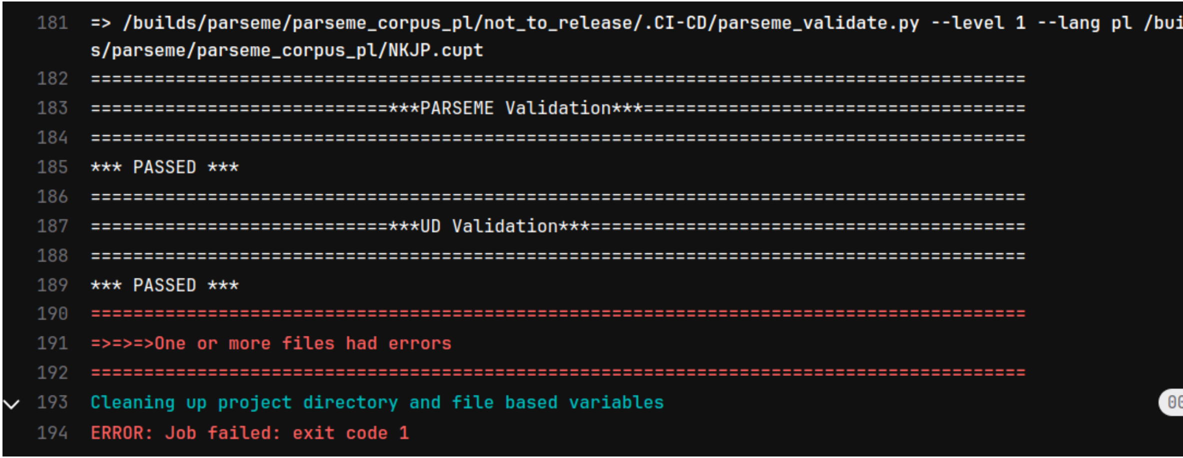
test_level_2

- ● test_cupt_level_2

# CUPT format validation behind Gitlab – a pipeline fails



# CUPT format validation behind Gitlab – pipeline errors



```
=> /builds/parseme/parseme_corpus_pl/not_to_release/.CI-CD/parseme_validate.py --level 1 --lang pl /bui
s/parseme/parseme_corpus_pl/NKJP.cupt
========================================================================================
============================***PARSEME Validation***====================================
========================================================================================
*** PASSED ***
========================================================================================
============================***UD Validation***=========================================
========================================================================================
*** PASSED ***
========================================================================================
=>=>=>One or more files had errors
========================================================================================
Cleaning up project directory and file based variables                                      00
ERROR: Job failed: exit code 1
```

# UD-PARSEME consistency

- The first 10 columns of a CUPT file are UD-compatible
- They may come from:
  - a UD treebank
  - automatic parsing with UDPipe (a parser trained on UD)
- When a new UD treebank is released, the PARSEME CUPT corpora become outdated
- The `lang-leaders/morphosyntax-update/`**`reannotate-morphosyntax.sh`** script in `https://gitlab.com/parseme/utilities` allows us to update a CUPT file with the latest UD treebanks
- The update is mostly automatic
- If tokenization chnaged, the script switches to interactive mode
- For a sample run, see the manual at `https://gitlab.com/parseme/corpora/-/wikis/Updating-morphosyntactic-annotations`

# Further reading

- UD release checklist
  `https://universaldependencies.org/release_checklist.html`
  (especially the Validation section)

- PARSEME tools wiki
  `https://gitlab.com/parseme/corpora/-/wikis/parseme-tools`