UDMorph: Morphosyntactically Tagged UD Corpora

Maarten Janssen
Charles University, Faculty of Mathematics and Physics
Prague, Czechia
janssen@ufal.mff.cuni.cz

Abstract

UDMorph provides an infrastructure parallel to that provided by UD for annotated corpus data that follow the UD guidelines, but do not provide dependency relations: a place where new annotated data-sets can be deposited, and where existing data-sets can be found and downloaded. It also provides a corpus creation environment to easily create annotated data for additional languages and variants. And it provides a REST and GUI interface to a growing collection taggers with a CoNLL-U output, currently for around 150 different languages, where taggers for new data-sets in UDMorph are automatically added.

Keywords: Universal Dependencies, Annotated Corpora, POS Tagging

1. Introduction

Linguistic annotations greatly increase the usability of corpora, especially for morphologically rich languages. Lemmas allow searching for words independently of the form it happens to be used in, part-of-speech tags and syntactic relations make it possible to search for constructions in the text. And named-entity labels allow going through the names in a text. Such linguistic annotations are automatically assigned by means of NLP pipelines, but those are only available for a limited amount of typically major languages. This puts researchers working on languages for which no NLP tools exist at a clear disadvantage, and perpetuates the use of a limited selection of languages for linguistic research.

However, most modern, statistically based NLP tools are completely language independent. The only thing that is missing to train NLP tools for new languages is training data - manually annotated example texts to train computational models on the language in question. And there is a steady development of training data and computational tools for new languages.

But there is a range of different problems in using these new tools and training data. The first is accessibility. There are many descriptions of data and tools in international journals, but the resources they describe are often not accessible anywhere, or are impossible to find. A second problem is that often, the resources are built either by people with a either linguistic background or with a computational background (but not both), while the skill-sets required to create a coherent and high-quality gold standard annotated data-set differ greatly from the skill-sets required to use those data to create accurate NLP tools. And a third problem is that there is a large amount of variation in the type of annotation provided in the different resources, making it difficult to work with multiple languages or use different data-sets within the same tool or resource.

The Universal Dependencies (UD) project[1] provides a solution for all three of these problems. UD makes resources available by providing a Git repository organisation where new treebanks can be deposited and maintained. And it makes them easy to find by providing a website listing all available treebanks with a description of the language and the status of the treebank. UD distinguishes the linguistic skills from the computational skills by separating the treebanks stored in the repository from the computational tools trained on them - UD provides a new release every half a year, and there are several computational tools that use the new releases to train computational models for all sufficiently large treebanks within the new release. And UD provides a single annotation scheme for any language in the world - the part-of-speech tags, morphosyntactic features and dependency relations are designed to be language independent, and are stored in the same format for all languages.

By providing a working infrastructure, UD has led to the availability of NLP tools in many more languages than were previously available. For instance UDPIPE (Straka and Straková, 2017) trained on version 2.12 of UD provides a dependency parser in over 70 different languages, and there are treebank data in over 100 languages. These data make UD not only language independent in theory, but proves that it is really usable in practice for a very wide range of languages - wide enough to merit the claim that it can be used for any language. And it is actively developed, meaning it can be adopted to new linguistic aspects not yet accounted for. On top of that the UD community provides a wide array of tools to work with, and has an active community that can help out in developing new treebanks.

But for many linguists, UD has one major draw-

[1] https://universaldependencies.org/

LREC-COLING 2024, pages 16933–16940

back - it is an infrastructure for dependency parsed treebanks. And creating a treebank is hard work, and not always what people are looking for. Creating a POS annotated and lemmatised data-set is often much less complicated, and sufficient for the purpose at hand. But POS annotated data-sets cannot be deposited to the UD infrastructure, and many of the tools provided by UD will not work on data without dependency relations. They use UD because it provides a well-establish language independent framework, with often example data available in languages similar to the one under development. But such resources cannot be hosted on the UD infrastructure since they lack dependency relations.

UDMorph attempts to remedy this drawback by providing an infrastructure parallel or at least similar to that provided by UD for data that follow the UD guidelines, but do not provide dependency relations. And in doing so aims to provide a place where new annotated data-sets can be deposited, existing data-sets are easy to find and available for training NLP tools, which in turn will hopefully lead to an NLP pipeline for an increasing number of languages and variants. And it provides an online interface and REST API access to use those tools in a format that is compatible with the output of UD tools like UDPIPE. Additionally, it provides a web-based tool that makes it easy for people to develop training data in new languages, and makes the training data in the UDMorph infrastructure accessible as searchable corpora.

The UDMorph web-site is hosted as a service by the LINDAT infrastructure: https://lindat.mff.cuni.cz/services/teitok-live/udmorph/, which provides links to the Git organisation, the tagging interface, and the searchable corpora. From the landing page there is a link to an overview of the current number of available taggers, corpora, and repositories.

2. Taggers

The ultimate goal of UDMorph is to enable and provide NLP pipelines in as many languages and language variants as possible. That includes not only languages, but also dialects, historical variants, and non-standard language domains. For this, it provides a web-based interface where people can submit texts in a growing number of languages, and get annotated results. The output is provided in CoNLL-U format, where the first lines indicate the tagger and the model that was used. It also provides a graphical output where the annotations for each word are displayed in a popup when hovering the mouse over it. At the time of writing, there are taggers for almost 150 different languages in the online interface of UDMorph, which is about twice as many as those provided by other UD based NLP tools. The taggers are not only provided as a web interface, but also as a REST service, so that the tagger output that can be directly integrated in an NLP pipeline.

The interface, as well as the REST service, provides a list of the languages for which a tagger is currently available, in which each language is identified by a name, as well as its ISO 639-3 code. For ease of use, it also provides an ISO 639-5 language family to group languages. For each language, it indicates which tagger is used for tagging, as well as the model that is used, and the features that are provided by the tagger.

There are two types of REST models used by the UDMorph online interface. There are external, existing REST services, mostly for that languages for which a UDPIPE model exists. For external REST services, the input data are sent to the REST service in question and only displayed locally. And there are REST models provided by UDMorph itself, that will run a tagger on the LINDAT servers. Those taggers that run on the LINDAT server can be further divided in three types - (1) those that in fact run on an external REST service but are executed via the LINDAT server to harmonize the output. (2) Those that are using existing, installable taggers that run in a locally installed version on the server. And (3) new taggers that use models that are trained on the UDMorph data-sets.

The local installation of taggers does not only provide tool developers to have a place where they can deploy their tagger, but also allows casual users to use the tagger without having to install it locally. The local taggers seldomly use CoNLL-U as their output format, and often do not use UD tags. Therefore, the output provided by the REST service is often not the direct output of the tagger, but rather the output converted to CoNLL-U. Where needed, the tags are also automatically converted to UD, with the original tag provided as a non-UD part-of-speech tag (XPOS), and the UPOS and FEATS columns provide the automatically converted data.

In principle, all taggers in UDMorph provide a lemma, a UPOS and FEATS, and optionally an XPOS. However, taggers that do not provide all these data are also included while no more complete tools are available. The language overview page lists which fields are provided by the tagger, with a star behind the UPOS and FEATS if they are not provided natively in UD, but rather were automatically converted, as in the case of locally installed non-UD taggers. Automatic conversion does not (always) lead to fully UD compliant output since the tags cannot always be correctly mapped. There are tags that lack information for proper automatic mapping - for instance many taggers do not distinguish between subordinate and coordi-

nate conjunction, meaning the UPOS column can only provide a less specific tag CONJ until it is disambiguated in context - a tag that is not a valid in UD. There are tags whose mapping is potentially incorrect in UD, for instance many tag-sets use a broader notion of auxiliary verb than what is used in UD meaning that some auxiliary verbs should have been tagged as verbs in UD.

In order to facilitate testing the different taggers, the GUI interface provides the option to load an example sentence for each language. For this it uses the first article of the Universal Declaration of Human Rights (UDHR), which is available in around 500 languages. Even with that many translations, the UDHR is not available for all languages for which there is a tagger, and if there is, it sometimes uses a different orthography. Therefore, there is the option for users to provide additional translations, especially for languages for which there is a UDMorph taggers. Also the collection of UDHR translations will be made available via the UDMorph repository.

The tagger interface is not intended to provide stable models, but rather always provides the most accurate UD based tagger available. If better or more complete taggers become available, they will replace the older version so that the output of the tagger service is always the best available output. In principle we will attempt to allow calling older versions explicitly, but especially third-party taggers might break in the future. For people relying on a stable tagger output, the models are always made available for download where possible, so that people can use them locally and have control over the version of the model they are using.

3. Data-sets

The more direct goal of UDMorph is to provide gold standard training data, parallel to what the UD infrastructure does, but then for data-sets that do not provide dependencies. Data-sets enable the NLP community to train new taggers with new techniques, which with the current advances in deep learning is bound to lead to increasingly accurate taggers. The guidelines for the data-set are exactly the same as for UD treebanks.

The data-sets in UDMorph are provided in a manner very similar to how they are provided in UD: as Git repositories in CoNLL-U format in a dedicated Git organisation[2], with metadata about the content and constitution of the corpus. The parallel set-up and shared data structure should enable developers to easily include UDMorph data into training pipelines developed for UD treebanks (and that do not rely on dependencies).

[2] https://github.com/UDMorph/

UDMorph provides access to all data-sets, including those data-sets that are not yet complete data-sets in UD style. And it aims to release periodic releases of all complete and correct data-sets that have a sufficiently free licence. As in the case of taggers, data-sets should at the bare minimum provide a form and a UPOS, and complete data-sets should provide FEATS, LEMMA, and optionally XPOS.

Some data-sets were automatically converted from non-UD source data, and do not necessarily (yet) fully follow the UD guidelines. Such non-compliance will be reflected in the metadata of the data-set, with only fully compliant data-sets included in the UDMorph release. An additional issue in data-sets converted from external sources is that many traditional POS data do not contain information about word spacing. For such data, the UD style spacing information (NoSpaceAfter) was added manually where expected, so that the data-sets can be used in modern-day NLP pipelines. Modern tool-chains often include a tokenization step, and tokenization badly fails if the word-spacing is missing in the training data. However, the automatically inserted NoSpaceAfter data do not necessarily reflect the spacing in the original data. So apart from the status of the tags and lemmas, the metadata also provide information about the status of the word spacing in each data-set.

There are data-sets in UDMorph that are not maintained at the UDMorph repository, but merely hosted there, much more than in the case of UD. Firstly, there are the data-sets that are converted from external sources, and if those original source are still being maintained, all contributions to such corpora should be made to the original, which typically is itself available as a Git repository. Secondly, UDMorph provides a graphical environment to allow people to more easily create new data-sets for new languages. For those corpora, the CoNLL-U data-set are exported from the data format used in the graphical interface, and hence also for those corpora, new contributions should not be added to the CoNLL-U data-set, but rather using the graphical interface.

UDMorph data-sets can optionally contain information in the MISC column, which can be of a range of different types, as prescribed in the UD guidelines. The most prominent types of miscellaneous information in annotated data-sets for less resourced languages tends to be named entities, as well as (English) glosses to make data more easy to interpret.

Despite the fact that the release of data-sets is a primary goal of the UDMorph project, there is currently only a handful of data-sets available as Git repositories. The reasons for that is that even though there are more data-sets on the server,

those data-sets were incorporated from external sources. Even though those sources are in principle all open source, we are still first looking for the confirmation by the original authors that their data can be made available in a modified format as a UDMorph data-set. The long-term objective is to have only a limited number of data-sets that can only be used internally (they are currently only used to train a tagger), since their limited functionality does not really justify the effort needed to convert them. The hope is that all current internal data-sets will gradually be distributed as Git repositories.

4. Corpora

Where permitted by the licence and/or the author, the UDMorph data-sets are also made available as searchable corpora on the UDMorph site, using the TEITOK corpus platform (Janssen, 2016). TEITOK is an open source platform that can be installed anywhere from the repository page[3].

The corpora in TEITOK are set-up in manner parallel to the UD corpora in TEITOK[4]. There are three different types of ways in which the TEITOK corpora were created. (1) External sources that are not yet in CoNLL-U have been first converted from their original format to a TEITOK corpus, and from there exported to CoNLL-U. (2) external sources that are in CoNLL-U from the start are only imported into TEITOK. And (3) there are corpora that have been generated directly in TEITOK using UDMorph as a corpus creation tool.

TEITOK internally uses TEI/XML, with the CoNLL-U attributes modeled as attributes over the token nodes. Although this is quite different from CoNLL-U, the information is fully compatible and can be losslessly converted in both directions (baring non-UD information that might be included in the corpus).

TEITOK makes the gold standard corpus data searchable using the Corpus Query Language of the Corpus WorkBench (Evert and Hardie, 2011). And it makes the corpus text visible as a readable document. The landing page for each corpus not only allows searching though the corpus, but also provides all relevant information about the corpus, in order to make sure it is made clear where the data came from and who was involved for its creation.

The TEITOK corpus can contain more information than what is exported to the CoNLL-U format. The reason for that is that if the original data contain additional information that can easily be incorporated and made searchable in TEITOK, it would be a missed opportunity to not include that information in the corpus. This is the same as the reason why UD treebanks can contain a non-UD POS tag as an XPOS, despite the fact that the XPOS has in principle little to do with UD. A example of such non-UD information can be found in the HUN-NERKOR corpus (Simon and Vadász, 2021), which contains a lemmatized form (the emLemma) using a different style of lemmatization from what UD requires. In parallel to the XPOS, this information is stored as an attribute XLEMMA in the TEITOK corpus for this resource, which can be used in searching and is diplayed in the interface[5]. And there might be other types of information in other external data-sets, such as entity linking information, morphological information as Interlinear Glossed Text, as well as more structured metadata about the source files than UD tends to provide.

The potential presence of additional information in the TEITOK corpus is one of the motivations behind using TEITOK as an intermediate format when converting existing data-sets to CoNLL-U. At least at the moment, such information is not included in the CoNLL-U files, since it would not add to the usability of the data-set for POS tagging and lemmatization. So converting directly to CoNLL-U would mean those data would not be available in the searchable corpus either, while in the current set-up using TEITOK as an intermediate platform it is.

The CoNLL-U files generated from TEITOK corpora can include data that are not standard in UD. The most prominent example is the inclusion of the token ID of the token in the TEITOK/XML that corresponds to the CoNLL-U line. The reason for that is that if for some reason corrections are made to the CoNLL-U export, the token IDs allow reincorporating those changes into the TEITOK/XML files.

5. Corpus Creation Tool

On the UDMorph web-site, TEITOK is not only provided as a corpus search tool, but also as a corpus creation tool to make the process of generating new data-sets easier. And we intend to use TEITOK in courses and hackatons, both on-site and virtual, to guide people along in the creation of new annotated data-sets.

TEITOK was created on the basis of tool called CorpusWiki (Janssen, 2013), which was a tool to generate gold standard POS annotated corpora, sharing its objective hence with UDMorph. TEITOK has since developed considerably, and has proven a popular tool for creating and correcting POS annotated corpora such as EModSar

[3] https://gitlab.com/maartenes/teitok

[4] https://lindat.mff.cuni.cz/services/teitok/ud212/

[5] https://lindat.mff.cuni.cz/services/teitok-live/udmorph/hun-nerkor/index.php?action=file&id=news/globalvoices_1.xml

(Puddu and Talamo, 2020) and CoDiAJe[6], especially for researchers with a limited computational background. It provides a visual environment for viewing and correcting token-level annotations like POS and lemma, and also provides several methods to speed up the process of correcting such annotations.

In order to build a new annotated corpus in TEITOK from scratch for a new language, the UDMorph interface provides the same bootstrapping set-up that was used in CorpusWiki and more recently in UDWiki (Janssen, 2021). In this set-up, the very first file in a new corpus has to be manually annotated. And as such, it is highly recommendable to start with a short text, ideally with a limited vocabulary. While annotating, the system will suggest the previously used tag for word-forms that have been used before. And for yet unknown words, it will allow you to select one of the universal part-of-speech (UPOS) tags, and depending on the UPOS tag allow selecting morphosyntactic features (FEATS).

Once the first text is fully annotated, it can be used to generate a tagger, which can then be used to automatically annotate a second text. Since the training data are still very limited the accuracy of that initial parse will be quite low. Still, correcting tags in a badly annotated text is quicker than having to manually annotated from scratch. And by retraining the tagger with every additional manually correct file, the tagger will gradually get better and the process will increasingly speed up.

In order to use the bootstrapping technique in which the tagger is frequently retrained, the training has to be fast and light-weight, meaning that taggers based on large language models cannot be used in this manner. Therefore, the system provide a choice between two older taggers: UDPIPE version 1, as well as NeoTag (Janssen, 2012), which was the tagger initially developed for CorpusWiki.

As mentioned, the TEITOK interface provides various options to make corrections in an efficient way, apart from correcting individual errors in a simple HTML form. The most prominent of those is the option to search for specific occurrences in the corpus using CQL, and then automatically or manually correct all those occurrences (see (Janssen et al., 2017)). This can be used to quickly correct known errors of the tagger, often triggered by spotting an individual error. For instance, if the word *hammer* in a very clearly verbal context would be tagged as a noun by the tagged, that might mean that it did not occur as a verb in the training data, and all occurrences of *hammer* should be revised. The interface makes it possible to restrict the number of cases that need to be revised using CQL, in the case of *hammer* for instance by only looking for the occurrences that to not follow a determiner. With the full CQL query, the interface present the full list of occurrences, where it is then possible to either change the POS or lemma to a given value, or create an editable list where each occurrence can be edited separately.

The UDMorph environment will provide three additional options as a starting point. The first is the option to use a multilingual large language model. Given an informal test of the accuracy on some languages, the current accuracy seems to be so low that is would only help in the first few, or maybe only in the very first file, after which it would be outperformed by tagger trained on the local data, even if that tagger is not fully up-to-date. But technology in this area is advancing rapidly, and perhaps could improve even more when UDMorph itself is used to train a multilingual tagger on a growing number of languages. These options still have to be implemented, so no experiences can be reported on how well it works in practice.

The second option is to start using an inflected lexicon. There are considerable amounts of languages for which there is an inflected lexicon, but no disambiguated texts using that lexicon. Lexicons that can be found in frameworks like UniMorph[7] or OntoLex-Morph (Chiarcos et al., 2022). We are gradually releasing conversion tools to convert inflected lexicons in various format to a format that provides UD style information. UD style lexicons can then be used in the corpus creation interface: when tagging a new unknown word, the interface will allow the user to select between the existing entries in the lexicon. This means that with a lexicon, the tasks of annotating the first text(s) consists of disambiguating between the various options provided by the lexicon, rather than tagging by hand. This not only speeds up the process but furthermore avoids coding errors. We aim to add lexicons in the published data where possible, in order to allow developers to use a lexicon to improve the accuracy of taggers based on the UDMorph data-sets.

And the last option is to use an existing UDMorph tagger for a language that either does not (yet) fully follow the UD guidelines, or does not yet provide all the fields. In such cases, the existing tagger can be used to provide all the data it can provide, so that the process of creating a more complete data-set would consist of correcting the errors and filling in the missing data. If for instance only a UPOS is provided by the tagger, that would imply manually lemmatizing, as well as providing the FEATS. And once completed, the full data-sets created out of a partial taggers and data-sets can then be included in the next UDMorph release. In this manner, UD-

[6] http://corptedig-glif.upf.edu/teitok/codiaje/

[7] https://unimorph.github.io/

Morph aims to gradually phase out taggers and data-sets that are not UD native.

One of the objectives of UDMorph is that a (complete) UDMorph tagger can be used in a similar way to automatically provide the morphosyntactic annotation for a new corpus, or to start directly from the annotated corpus, to allow people to manually add dependency relations to generate a full treebank. In that respect, one of the objectives of UDMorph is to make itself redundant by gradually have the data-sets converted into full treebanks that can be hosted on the UD infrastructure.

And of course it is not necessary to start from scratch with a new corpus, there might be existing data to start from. The corpus creation tool can be used to extend, complete and correct corpora based on partial data-sets directly instead of creating a new corpus based on their tagger output. And new corpora can be created out of existing annotated data-sets for that purpose as well. Given the CQL based manner mentioned above, the TEITOK interface can help in the process of correcting and completing. For instance to distinguish subordinate and coordinate conjunction in corpora that do not yet make that distinction, the interface makes it easy to say first change all conjunctions to SCONJ, and then go through the (typically limited list of) coordinate conjunctions one by one to convert conjunctions like *and* and *or* to CCONJ instead.

6. Contributing

UDMorph will of course only work with the participation of researchers around the world. It is an initiative to provide services to host and use annotated data-sets created by other people, and not a large project to create new annotated data. So we very much invite anybody to contribute additional data-sets to the initiative, whether they be existing resources or newly created ones.

An important objective is to make it easy to contribute to the UDMorph infrastructure, and to make sure contributions to UDMorph are properly cited and provide attribution citations. Since UDMorph has various objectives and types of corpora, there are also various ways to contribute.

The best way to contribute is to create or provide (and then subsequently maintain) a fully annotated data-set following the UD guidelines. A data-set with a sufficiently open licence that can be included in the next release. Such full data-sets will be published as a Git repository in the UDMorph workspace, and will be made available in the future as a HuggingFace data-set. We will train a tagger on the data-set and include it in the tagger service, both via the GUI and via the REST API. It will be made available as a searchable TEITOK corpus. And hopefully it will be picked up by researchers to generate increasingly accurate NLP pipelines.

To provide such full annotated data-sets, researchers should contact us so that we can generate a Git repository to host the data. Like in the case of a new UD treebank, it needs an ISO code of the language, with potentially an addition to indicate a specific dialect, region, style, or period, as well as a name that should be used for the language. And then there are two ways to have that repository be set-up.

The first option is to have a resource that is maintained as a TEITOK corpus for which you will be the main administrator allowing you to make change to the data, and from which the CoNLL-U files in the repository will be generated. As mentioned in the previous section, TEITOK provides several ways to efficiently adapt existing annotated corpora to UD compliant data-sets.

The second possible set-up for a full data-set is to maintain the Git itself and modify the CoNLL-U files with any editing tools you prefer. In those cases, you can be an administrator of the repository in the UDMorph repository itself, or the UDMorph repository can be a clone of a repository that is maintained elsewhere.

We are also happy to host data-sets converted from existing repositories that are either following UD but not provided in CoNLL-U, or in a standard that can be converted to UD. In the latter case, we would furthermore need a table or script to convert the data. In those cases, the data-set will largely be treated as a the full data-sets above, except that data that are not yet completed and corrected will not be included in the release.

At least in the initial phase, we will also be happy to convert existing data-sets that cannot be made available, in order to train a tagger on such data that can then hopefully be used to later generate a full UDMorph data-set in the set-up sketched in the previous section.

We are also happy to include available taggers that cannot be provided as data-sets for languages not yet available in UDMorph. In the best case scenario those are taggers that provide UD output in CoNLL-U, provided as a REST server. But we can also install taggers locally that either provide UD output or can be converted to UD output. Non-UD taggers or locally installed taggers would ideally lead over time to full data-sets that can be released.

And finally, anybody using the UDMorph data or other data to create new taggers or tagger models that (significantly) outperform the taggers currently provided in UDMorph and can be run (ideally as a REST service), we would be more than happy to replace the currently used tagger by its improved counterpart. As mentioned, the goal of the online tagger interface is to provide the most accurate data available.

7. Conclusion

UDMorph provides an infrastructure to help towards a better coverage of NLP tools for more languages, following the multilingual design of the UD annotation guidelines. It does this by providing an infrastructure parallel to that for UD treebanks where annotated data-sets following the UD guidelines can be hosted, without requiring them to be full treebanks. It furthermore provides an online interface where taggers with a CoNLL-U output in UD style can be used online. And it provides a corpus creation environment that makes it easier to create new data-sets for new languages or variants.

As an infrastructure initiative, UDMorph will only work if people contribute to it, but it already provides a considerable amount of services based on data we were able to gather ourselves. An initial internet sweep has shown that there are quite a few annotated data-sets out there for languages without NLP tools at the moment, and a significant amount of those is using are using UD guidelines, such as the Masakhane initiative for African languages[8]. There are also quite a few published papers that describe taggers that do not seem to be available by any means, potentially either because of the lack of a proper way to deploy the tagger, or because of a lack of proper attribution. By providing both an infrastructure and a way to acknowledge the efforts of the people doing the actual work, we hope to convince research to make more of such resources available.

With such data, as well as by using UD treebanks that are still too small to train a parser, but lead to decent results for the simpler task of tagging and lemmatization, UDMorph already provides around twice as many languages as other tagging or parsing tools available at the moment. And with a community effort, helped by the graphical tools and support we provide, the hope is to have this number grow considerably.

And as has been reported by the experience of ClassLA (Ljubešić et al., 2021), annotated data-sets tend to be larger than treebanks, and larger training data lead to better accuracy. So potentially UDMorph can provide more accurate data even for languages for which there already is a full UD treebank.

8. Acknowledgements

This work has been supported by the Ministry of Education, Youth and Sports of the Czech Republic, Project No. LM2023062 LINDAT/CLARIAH-CZ

[8] https://www.masakhane.io/

9. Bibliographical References

Christian Chiarcos, Christian Fäth, and Maxim Ionov. 2022. Unifying morphology resources with ontolex-morph: a case study in german. In *Proceedings of the 13th Language Resources and Evaluation Conference, 20-25 June 2022, Marseille, France*, pages 4842 – 4850.

Stefan Evert and Andrew Hardie. 2011. Twenty-first century corpus workbench: Updating a query architecture for the new millennium. In *Corpus Linguistics 2011*.

Maarten Janssen. 2012. Neotag: a POS tagger for grammatical neologism detection. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, pages 2118–2124. European Language Resources Association (ELRA).

Maarten Janssen. 2013. POS tagging and less resources languages individuated features in corpuswiki. In *Human Language Technology. Challenges for Computer Science and Linguistics - 6th Language and Technology Conference, LTC 2013, Poznań, Poland, December 7-9, 2013. Revised Selected Papers*, volume 9561 of *Lecture Notes in Computer Science*, pages 411–419. Springer.

Maarten Janssen. 2016. TEITOK: Text-faithful annotated corpora. *Proceedings of the 10th International Conference on Language Resources and Evaluation, LREC 2016*, pages 4037–4043.

Maarten Janssen. 2021. UDWiki: guided creation and exploitation of UD treebanks. In *Proceedings of the Fifth Workshop on Universal Dependencies (UDW, SyntaxFest 2021)*, pages 84–95, Sofia, Bulgaria. Association for Computational Linguistics.

Maarten Janssen, Josep Ausensi, and Josep Fontana. 2017. Improving POS tagging in old spanish using TEITOK. In *Proceedings of the NoDaLiDa 2017 Workshop on Processing Historical Language, Gothenburg, Sweden, May 22, 2017*, pages 2–6. Linköping University Electronic Press.

Nikola Ljubešić, Taja Kuzman, Tomaž Erjavec, and Petya Osenova. 2021. Tour de CLARIN: The CLARIN knowledge centre for south slavic languages (CLASSLA).

Nicoletta Puddu and Luigi Talamo. 2020. Emodsar: A corpus of early modern sardinian texts.

Eszter Simon and Noémi Vadász. 2021. Introducing nytk-nerkor, A gold standard hungarian named entity annotated corpus. In *Text, Speech, and Dialogue - 24th International Conference, TSD 2021, Olomouc, Czech Republic, September 6-9, 2021, Proceedings*, volume 12848 of *Lecture Notes in Computer Science*, pages 222–234. Springer.

Milan Straka and Jana Straková. 2017. Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.