PALMYRA 2.0: A Configurable Multilingual Platform Independent Tool for Morphology and Syntax Annotation

Dima Taji and **Nizar Habash**
Computational Approaches to Modeling Language (CAMeL) Lab
New York University Abu Dhabi, UAE
{dima.taji,nizar.habash}@nyu.edu

Abstract

We present PALMYRA 2.0, a graphical dependency-tree visualization and editing software. PALMYRA 2.0 is designed to be highly configurable to any dependency parsing representation, and to enable the annotation of a multitude of linguistic features. It uses an intuitive interface that relies on drag-and-drop utilities as well as pop-up menus and keyboard shortcuts that can be easily specified.

1 Introduction

The development of treebanks is central to research on automatic syntactic and morphological analysis. Treebanks can vary based on the syntactic representations they are in, and the languages they encode. There is a large number of syntactic representations that vary in terms of linguistic theories underlying them, and in terms of their file formats. The Universal Dependency (UD) representation (Nivre et al., 2016) aims to be applicable in all languages. UD is currently one of the most popular representations being available in 90 languages, but many other treebanks are designed for a limited number of languages. The Penn Treebank (PTB) (Marcus et al., 1993) is the most used representation for constituency treebanks, and it is available in a number of different languages including English (Marcus et al., 1993), Arabic (Maamouri et al., 2003), and Chinese (Xue et al., 2005). The Prague Dependency Treebank (PDT) (Böhmová et al., 2003) is another representation used to annotate treebanks in Czech (Böhmová et al., 2003) as well as Arabic (Hajič et al., 2004).

The annotation of syntactic trees with morphological features, especially for morphologically rich languages, is often done in a cascading manner: text is annotated first for morphology then for syntax. This method of annotation can have errors cascading from one level of annotation to the other. Having the annotation done in tandem, or at least having the morphological features displayed and editable when annotating syntax, will allow annotators to look at the annotations in a different light, bringing to their attention different readings that may have been missed otherwise.

The richness of a language's morphology is not necessarily related to the richness of its treebank representation. Arabic, for example, is a very morphologically rich language, and it has different treebanks, with different representations. The Penn Arabic Treebank (PATB) (Maamouri et al., 2003) uses a constituency representation that is rich in morphological features. The Columbia Arabic Treebank (CATiB) (Habash and Roth, 2009), on the other hand, is a dependency representation with poor morphology. The designers of CATiB claimed that the simplified morphology was intended among other choices to speed annotation up. This motivated us to create a tool that is configurable and can work for multiple languages and morphosyntactic representations.

The existence of these varying treebanks and different approaches to annotation naturally raise the need for a tool that can be used with the different representation schemes and languages. We sum up our desiderata for such a tool as follows:

- A tool that is platform independent and requires no special installation.
- A tool that is lightweight and can be used offline.

Proceedings of the Fourth Workshop on Universal Dependencies (UDW 2020), pages 168–177

- A tool that is open-source.
- A tool that is intuitive to use and requires minimal training.
- A tool that allows the text of the trees to be easily displayed and searched.
- A tool that can be used at any point in the annotation process.
- A tool that supports morphologically rich languages.
- A tool that is usable with various syntactic representations.
- A tool that is usable with various linguistic features.

In our previous work describing the creation of PALMYRA (Javed et al., 2018) (henceforth, PALMYRA 1.0), we addressed platform independence by building a web-based dependency syntax annotation tool that did not require any installation. PALMYRA 1.0 uses an intuitive drag-and-drop interface, and has the options of annotating using both buttons and shortcut keys. PALMYRA 1.0's most important feature is the ease of fixing tokenization errors caused by morphological analyzers or syntactic parsers, by allowing users to easily split and merge nodes in the displayed trees.

In the new iteration of PALMYRA (henceforth, PALMYRA 2.0),[1] we extend the tool's ability to be used with varying syntactic representations and linguistics features through the introduction of an adaptable configuration file. PALMYRA 2.0 can be configured to support annotations that go beyond POS tags and syntactic relations, and extend to other linguistic features such as morphological features.

When building PALMYRA 2.0 we only focused on syntactic dependency representations, but we imagine many of its design elements can be expanded to constituency trees.

We discuss some related work in Section 2. We then present our design specifications (Section 3), user interface (Section 4), and the configuration setup (Section 5).

2 Related Work

Many annotation tools have been created in the context of all the work done on syntactic parsing and morphological analysis. Most tools are designed for one type of annotation only, and have limited configurations. In this section, we review previous work that was done on developing tools for linguistic annotation, specifically syntactic annotation as well as joint syntax and morphology annotation. While the tools we present here all have great features, some features that we consider key for ease of annotation are also lacking.

Morphological Annotation Tools A few tools exist to carry out morphological annotations. CorA (Bollmann et al., 2014) is an annotation tool for 'non-standard' language texts in German that has options for normalization, lemmatization, and morphological tagging. DIWAN (Al-Shargi and Rambow, 2015) and MADARi (Obeid et al., 2018) are morphological annotation interfaces designed specifically for Arabic text. Wasim (Alosaimy and Atwell, 2018) is a web-based tool for morphological annotation of inflectional languages that has some support for editing tokenization. None of these tools can be used to annotate for features that are beyond the ones they are initially designed for.

Syntactic Annotation Tools The best known tool for dependency treebank creation is TrEd (Pajas, 2008), which is a graph visualization and manipulation program written in Perl. Although it can be configured to automate frequently repeated operations, TrEd does not have a simple option for word tokenization, and can be difficult to install and learn. EasyTree (Little and Tratz, 2016) is a light-weight tool designed to annotate dependency trees in browsers. It does not maintain sentence order of nodes, and it has no functionality for editing word tokenization. However, EasyTree is very intuitive to use, and it is the base on which PALMYRA is built. EasyTree's successor, CrowdTree (Tratz and Phan, 2018), is designed to support human-in-the-loop methods as well as crowdsourcing, using a TrEd-inspired interface. It has a backend servlet that can train a parsing model during the annotation process, which helps produce automatic annotations for the annotators to start from. However, CrowdTree does not support the annotation of any other linguistic features, or word tokenization.

[1] We use the term PALMYRA to refer to the tool generically; and only specify the version number when discussing version-specific features.

Joint Morphology and Syntax Annotation Tools BRAT (Stenetorp et al., 2012) is a web-based annotation tool that focuses on collaborative annotation. One of the features of BRAT is that it is not limited to one kind of annotation. It can be used to annotate syntax, event extraction, named entity detection, and coreference resolution, to mention a few. However, BRAT requires users to login to be able to annotate their text online, and it does not allow for web-based configuration of tag sets, and it does not support word tokenization. Another web-based tool is WebAnno (de Castilho et al., 2016) which allows the annotation of various layers of linguistic annotations, including semantic and syntactic annotations. But similar to BRAT, WebAnno does not support word tokenization. Finally, ConlluEditor (Heinecke, 2019) is an editor designed with UD in mind, allowing for the editing of syntactic information, as well as morphological features. It supports word tokenization editing, and displays multi-word tokens, empty nodes, and enhanced dependencies. However, ConlluEditor requires installation and a server setup.

3 Design Specifications

In this section, we discuss the key decisions that we followed in designing PALMYRA 2.0. We are aware that there are various specification that are desirable by different annotators, such as user management, automatic parsing, and inter-annotator agreement reports. However, there is a tradeoff between having those features and having a lightweight and fast tool that can run offline, which we prioritized.

Design and Implementation PALMYRA is a platform independent open-source software[2] that is written entirely in JavaScript, HTML, and CSS. This makes it usable with modern browsers without the need of any installation or setup.[3]

Input File The most commonly used format for dependency parsing currently is the CoNLL-U format, a revised version of CoNLL-X (Buchholz and Marsi, 2006), that was made popular for its use in UD (Nivre et al., 2016). Because we want PALMYRA 2.0 to be usable by people annotating in different dependency representations, we decided to adopt this well established format for our files. Furthermore, PALMYRA 2.0 can be used to display any languages that can be encoded in Unicode.

Entries in CoNLL-U files have ten columns that correspond to the following fields: ID, form, lemma, POS, XPOS, features, parent, relation, enhanced dependency, and miscellaneous. The POS field is what is used to populate the tree with POS tags. The XPOS field, if specified in the configuration files, can be edited alongside the morphological features.

Figure 1 shows two trees from different representations, in the same CoNLL-U file format. We decided to go with an example in Arabic, a morphologically rich language, for two reasons. First, we want to illustrate the usability of PALMYRA 2.0 with different representations, in this case CATiB (Figure 1 (a)) and UD (Figure 1 (b)). And secondly, we want to demonstrate a use case that requires word tokenization and rich morphological features as part of creating syntactic trees: in Figure 1, see tokens 1, 8 and 11 which are proclitics, and the many features in column 6.

In the current version of PALMYRA 2.0, the CoNLL-U enhanced dependency field and the miscellaneous field are not used in the visualization of the tree, and are not editable through the interface. However, if they appear in the input file, they will be saved by the tool and written in the output file. Multi-token words and hidden nodes are not supported in the current version. Comments in the CoNLL-U format are stored and written to the output file. However, we currently do not allow editing them. We plan to do these extensions in the future.

Output Files PALMYRA 2.0's main output is in the same CoNLL-U format as the input files. It also give the option of downloading the trees as PNG images.

Configuration File PALMYRA 2.0 uses JSON format for its configuration file. The configuration file is discussed in details in Section 5.

[2]PALMYRA's code is available at `https://github.com/CAMeL-Lab/palmyra`.

[3]Try PALMYRA online at `https://camel-lab.github.io/palmyra`.

Gloss	ID	Token	Lemma	POS	XPOS	Features	Parent	Relation	Depd	Misc
and	1	+وَ	وَ	PRT	_	gloss=and	2	MOD	_	_
was	2	كَانَ	كَان	VRB	_	gloss=be;was;were|asp=p|vox=a|mod=i|per=3|gen=m|num=s	0	--	_	_
people	3	أَهْلُ	أَهْل	NOM	_	gloss=family;people|gen=m|num=p|stt=c|cas=n|rat=r	2	SBJ	_	_
the-Earth	4	الأَرْضِ	أَرْض	NOM	_	gloss=earth;land|gen=f|num=s|stt=d|cas=g|rat=i	3	IDF	_	_
all	5	جَمِيعاً	جَمِيع	NOM	_	gloss=all|gen=m|num=p|stt=i|cas=a	3	MOD	_	_
speak	6	يَتَكَلَّمُونَ	تَكَلَّم	VRB	_	gloss=speak;talk|asp=i|vox=a|mod=i|per=3|gen=m|num=p	2	PRD	_	_
primarily	7	أَوَّلاً	أَوَّل	NOM	_	gloss=primarily|gen=m|num=s|stt=i|cas=a	6	MOD	_	_
with	8	+بِ	بِ	PRT	_	gloss=with;by	6	MOD	_	_
tongue	9	لِسَانٍ	لِسَان	NOM	_	gloss=tongue|gen=m|num=s|stt=i|cas=g|rat=i	8	OBJ	_	_
one	10	وَاحِدٍ	وَاحِد	NOM	_	gloss=one|gen=m|num=s|stt=i|cas=g	9	MOD	_	_
and	11	+وَ	وَ	PRT	_	gloss=and	9	MOD	_	_
language	12	لُغَةٍ	لُغَة	NOM	_	gloss=language|gen=f|num=s|stt=i|cas=g|rat=i	11	OBJ	_	_
one	13	وَاحِدَةٍ	وَاحِد	NOM	_	gloss=one|gen=f|num=s|stt=i|cas=g	12	MOD	_	_
.	14	.	.	PNX	_	_	2	MOD	_	_

(a) CATiB Tree in CoNLL-U Format

Gloss	ID	Token	Lemma	POS	XPOS	Features	Parent	Relation	Depd	Misc
and	1	+وَ	وَ	CCONJ	conj	_	6	cc	_	_
was	2	كَانَ	كَان	AUX	verb	Aspect=Perf|Gender=Masc|Num=Sing|Mood=Ind	6	cop	_	_
people	3	أَهْلُ	أَهْل	NOUN	noun	Gen=Masc|Num=Sing|Case=Nom|Definite=Cons	6	nsubj	_	_
the-Earth	4	الأَرْضِ	أَرْض	NOUN	noun	Gen=Fem|Num=Sing|Case=Gen|Definite=Def	3	nmod	_	_
all	5	جَمِيعاً	جَمِيع	ADJ	noun_quant	Case=Acc|Definite=Ind	3	amod	_	_
speak	6	يَتَكَلَّمُونَ	تَكَلَّم	VERB	verb	Aspect=Imp|Gender=Masc|Num=Plur|Mood=Ind	0	root	_	_
primarily	7	أَوَّلاً	أَوَّل	ADJ	adj	Case=Acc|Definite=Ind	6	amod	_	_
with	8	+بِ	بِ	PART	prep	_	9	case	_	_
tongue	9	لِسَانٍ	لِسَان	NOUN	noun	Gen=Masc|Num=Sing|Case=Gen|Definite=Ind	6	obj	_	_
one	10	وَاحِدٍ	وَاحِد	NUM	noun_num	Gen=Masc|Num=Sing|Case=Gen|Definite=Ind	9	nummod	_	_
and	11	+وَ	وَ	PART	conj	_	9	cc	_	_
language	12	لُغَةٍ	لُغَة	NOUN	noun	Gen=Fem|Num=Sing|Case=Gen|Definite=Ind	9	conj	_	_
one	13	وَاحِدَةٍ	وَاحِد	NUM	noun_num	Gen=Fem|Num=Sing|Case=Gen|Definite=Ind	12	nummod	_	_
.	14	.	.	PUNCT	punct	_	6	punct	_	_

(b) UD Tree in CoNLL-U Format

Figure 1: Two CoNLL-U trees represention of the same sentence وَكَانَ أَهْلُ الأَرْضِ جَمِيعاً يَتَكَلَّمُونَ أَوَّلاً بِلِسَانٍ وَاحِدٍ وَلُغَةٍ وَاحِدَةٍ. *wakAna Âhlu AlÂarDi jamiyçAã yatakal∼amuwna Âw∼alAã bilisAnĩ wAHidĩ walugaħĩ wAhidaħĩ* 'And the whole earth was of one language, and of one speech.': (a) CATiB representation, with features in a format similar to MADAMIRA (Pasha et al., 2014), and (b) UD representation, with UD morphological features. The first column on the left of the figure is added to display the English glosses of the Arabic tokens for ease of reading.

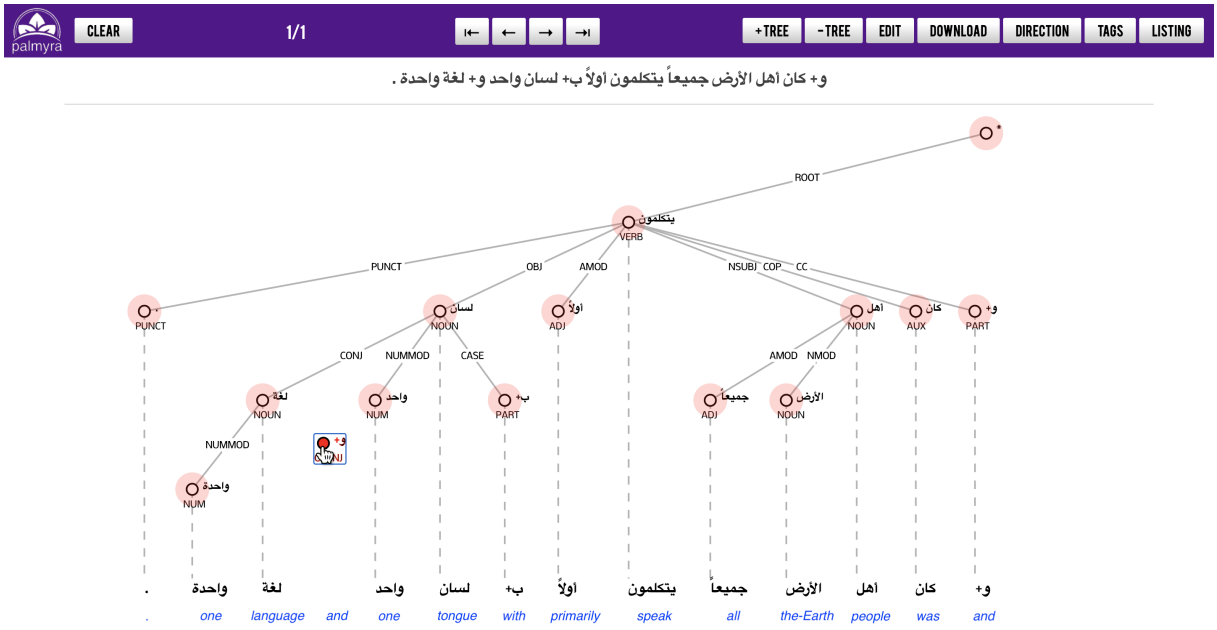


Figure 2: PALMYRA 2.0 interface while editing a tree using the drag-and-drop functionality. The tree shown in this example is the visualization of the UD tree shown in Figure 1 (b). The translation in blue at the bottom of the image is added for clarification purposes, and should be read from right to left.

4 PALMYRA 2.0's User Interface

In this section, we present the details of the interface. This section has some information that was first presented in PALMYRA 1.0 (Javed et al., 2018), however, some repetition is required to illustrate the work we built on top of it.

Dependency Tree Display Figure 2 presents a dependency tree being edited. The sentence's tokens are projected at the bottom, and the tree nodes are aligned directly above them. The POS tag for each word is shown directly below it, and the relation is displayed half-way on the edge connecting the node to its parents. The display direction can be changed to be left-to-right or right-to-left using the DIRECTION button. It can also be automatically configured, as we will show in Section 5.

The screenshot in Figure 2 is taken midway through dragging and dropping a node. The 'drop zone' of each node is indicated by a red halo around it. When a node is attached to a new parent, the order of the words in the sentence, and in the tree, remains the same. PALMYRA also has tree zoom-in and zoom-out options.

Tree Navigation and Editing Users who prefer to use the keyboard to edit the trees can do that with ease. Once a node is selected, users can use arrows to move from one node to the other. The selected node is always highlighted to provide visual feedback to the user.

Editing POS tags and relation labels can be done by either using configurable keyboard shortcuts, or using sub-menus that provide button lists of all available annotation options. The tagging sub-menus are invoked by clicking on the TAGS button. The sub-menu button lists are populated dynamically from the configuration file, and can be grouped in whichever way the user feels relevant to their annotations task. Figure 4 (a) shows an example of these sub-menus for UD annotation. Switching between annotating POS tags and relation labels is done using the Tab key.

Additions, Deletions, and Mergers In PALMYRA 2.0, the addition and deletion of nodes is done using the Edit mode, which can be initiated by clicking on the EDIT button. In this mode, icons are displayed on the tree to indicate addition, deletion, and left and right mergers. Figure 3 displays the tree in sentence token editing mode. The clickable red "x" sign and green "+" sign are used to delete and insert nodes, respectively. The default POS tag and relation of the new node are configurable. Merging neighboring

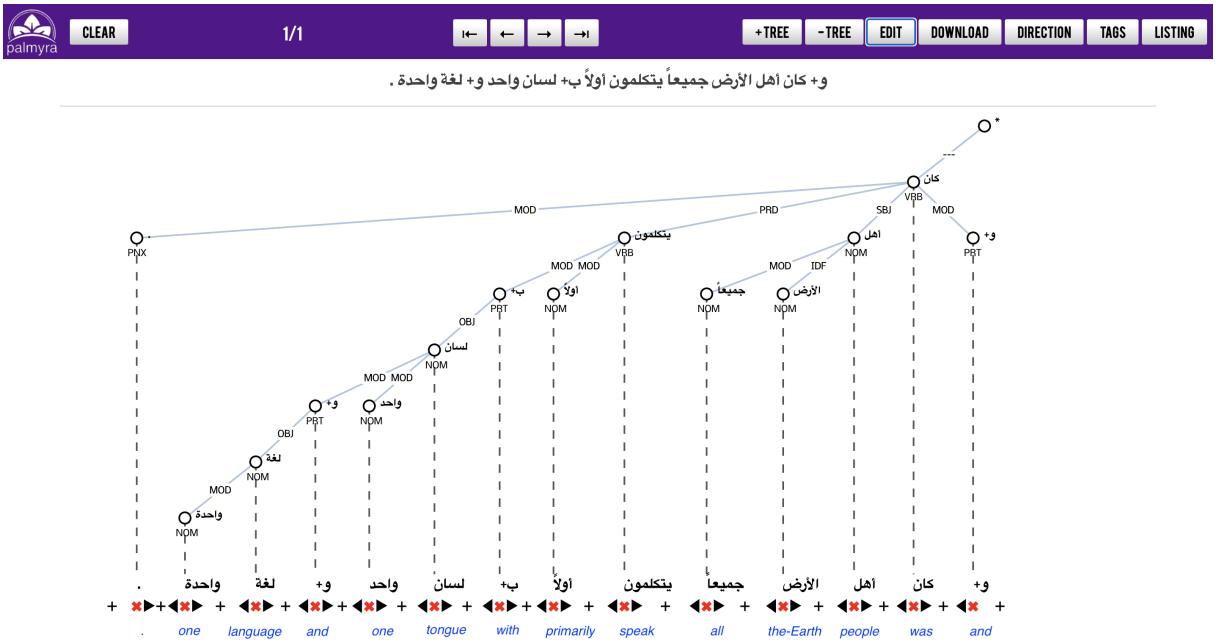


Figure 3: PALMYRA 2.0 interface in sentence token editing mode. The tree shown in this example is the visualization of the CATiB tree shown in Figure 1 (a). The translation in blue at the bottom of the image is added for clarification purposes, and should be read from right to left.

words is done through the clickable left and right arrows. The node under which the arrow is clicked is deleted, and its "name", i.e. the word itself, is added to the "name" of the respective neighbor node. The children of the merged node are assigned to the node it was merged with.

The user can also use the button +TREE to add a new tree to the end of the the file. The tree starts as a root node only, and the user can add new nodes using the word insertion functionality described above. The -TREE button deletes the tree that is currently in focus.

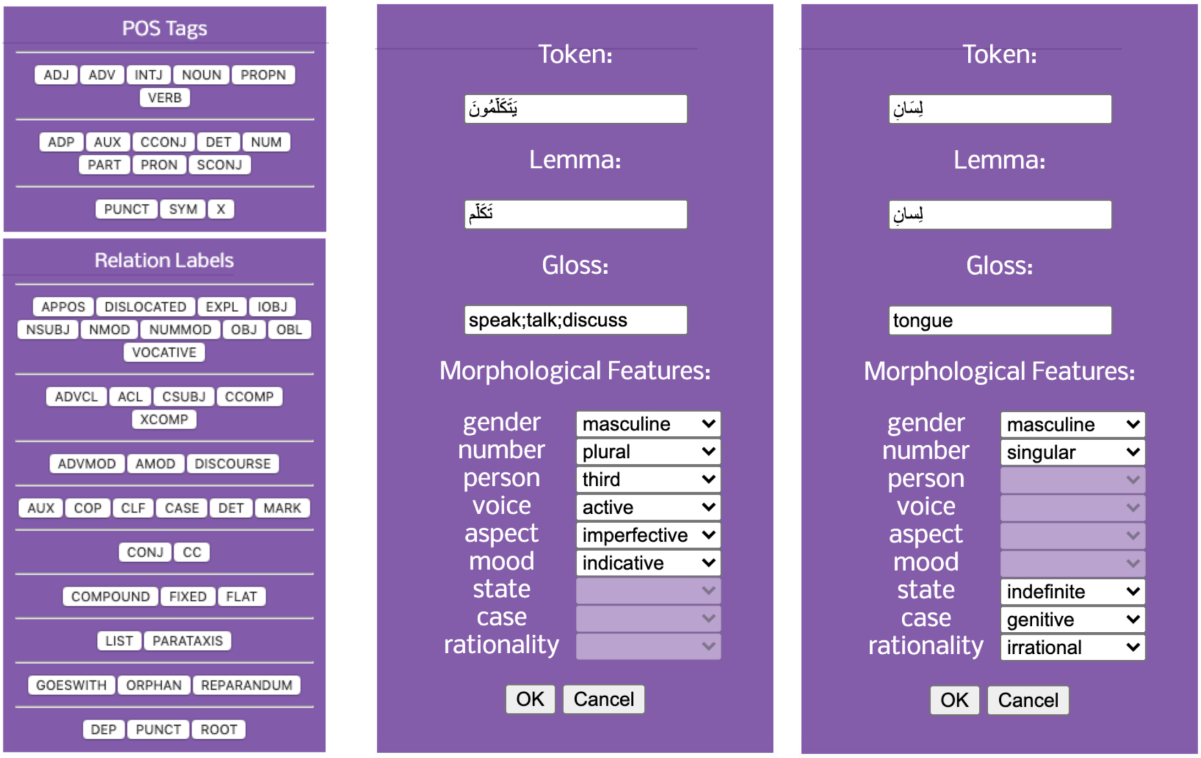
Word and Feature Editing Clicking on one of the words projected at the bottom of the tree allows the users to edit that word and its features. This includes the token, which users can edit to fix typographical errors, or add spaces to split the token into multiple tokens. Additionally, PALMYRA 2.0 gives the option of editing lemmas, and the features that appear in the sixth column of the CoNLL-U file.

Figure 4 shows the menu that pops up when a word is clicked. Users can use the configuration file to allow or disallow some features from being annotated with specific POSs. Figure 4 (b) shows the menu with a verb selected, where the features *state*, *case*, and *rationality* are disabled, and Figure 4 (c) shows the same menu when a noun is selected where *person*, *voice*, *aspect*, and *mood* are disabled.

Text Listing PALMYRA 2.0 can show a listing of the sentences that are in the file that is being displayed. Figure 5 presents the box that appears when clicking on the LISTING button, showing all the sentences that are in the uploaded file. Because PALMYRA 2.0 is a light tool, our approach is to exploit the existing search utility in web browsers to our advantage. The text in this box is searchable with the search utility, and the users can easily navigate to the sentence they desire by clicking on it in this box. If a specific key is specified in the configuration file, the text displayed in the listing box will come from the text in the comments with that key. If the key is not in the configuration, or if a tree does not have a comment with that key value, the text displayed in the listing box will be the concatenation of the sentence's tokens.

5 Configuration

PALMYRA 2.0's configuration file is a JSON file. The configuration file allows users to specify options related to the following features:



(a) (b) (c)

Figure 4: (a) Selection menus for POS tags (top) and relation labels (bottom); (b) and (c) Word and feature editing interfaces, showing the difference between editing a verb (b) and editing a noun (c).

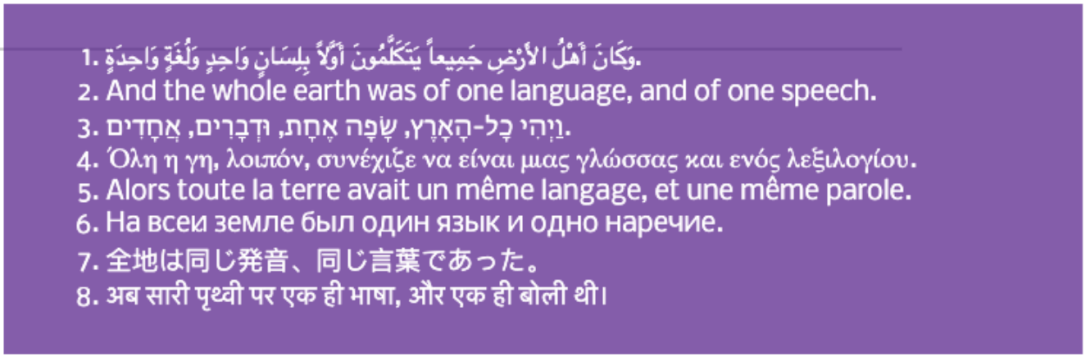


Figure 5: The listing of the sentences that are in the file being displayed. All are translations in different languages of the same sentences.

Orientation PALMYRA 2.0 can display dependency trees in right-to-left and left-to-right orientations. The user can set their preferred orientation through the configuration file.

Text Listing The text displayed in the listing box by default comes from treating the tokens in the tree as independent words. PALMYRA 2.0 can be configured to populate the listing box from a specific comment in the CoNLL-U file.

Display Features The features that are displayed by default on syntax trees are the tokens' POS tags, and their relations to their parents. The possible POS tags and relation labels that can be used for annotation are defined in the configuration file. Figure 6 (a) and (b) show a partial definition of POS tags and relations, respectively, in a UD configuration file. The *label* defines the name of the label that will be displayed on the tree. The *key* defines the shortcut that is used to select that label. Clicking on a shortcut key multiple time will circulate among all the tags that have the same key value, in the order in

```
"pos":{
 "values":[
  {"label":"adj",
           "key":"a", "group":"1"},
  {"label":"adv",
           "key":"a", "group":"1"},
  {"label":"cconj",
           "key":"c","group":"2"},
  ... ]}
```

(a)

```
"features":[
 {"name":"case",
  "type":"list",
  "values":["nominative","accusative",
            "genitive"]},
 {"name":"gender",
  "type":"list",
  "values":["feminine","masculine"]},
 {"name":"gloss",
  "type":"lexical"},
  ... ]
```

(c)

```
"relation":{
 "values":[
  {"label":"det",
           "key":"d", "group":"4"},
  {"label":"dep",
           "key":"d","group":"9"},
  ... ]}
```

(b)

```
"default_features":[
 {"pos":"NOM",
  "features":[
    {"name":"case","value":"nominative"},
    {"name":"gender","value":"masculine"},
    ... ]},
 {"pos":"VRB",
  "features":[
   {"name":"case","value":"N/A"},
   {"name":"gender","value":"masculine"},
    ... ]},
 ...]
```

(d)

Figure 6: Excerpts of the configuration file showing partial definition of (a) the POS tags and (b) the relation labels that will be displayed on the tree, as well as (c) the features and values that can be annotated and (d) the associated default feature values.

which they appear in the configuration file. This applies to both POS tags and relation labels. The *group* is used to group specific keys together in the pop-up menu. Figure 4 (a) shows the pop-up menu for the basic UD POS tags and relation labels, with specific tags and labels grouped together.

Linguistic Annotations CoNLL-U files use the sixth column for representing token features. PALMYRA 2.0's configuration file allows users to specify which features they are interested in annotating. Each feature can be one of two types: list or lexical. A lexical feature is edited using a text box, with no limitation on its value. A list feature can be edited through a drop-down list that is predefined in the configuration file. Figure 6 (c) shows a partial definition of three features; *case* and *gender*, both of which are list features that can take a limited set of specific values, and *gloss* which is a lexical feature. Figure 4 shows the linguistic features' pop-up menu that is populated from the full configuration file.

Default Values Users can define default feature values that are connected to specific POS tags. Figure 6 (d) shows a partial definition of default features. A *NOM* POS tag per this configuration will have a default *case* value of 'nominative', unless it is annotated otherwise in the input file. A *VRB* POS tag, on the other hand, cannot have a *case* feature value, so it is declared to be 'N/A'. Figure 4 shows the difference between the linguistic pop-up menus for a verb (b) and a noun (c).

Configuration file examples and examples of corresponding dependency tree files are provided in the PALMYRA website (see footnote 3).

6 Conclusion and Future Work

In this paper, we described PALMYRA 2.0, a platform independent graphical software for dependency tree visualization and editing. The features of PALMYRA 2.0 allow it to be easily configured to work with

any syntactic dependency representation. They also allow for editing linguistic features through simple drop-down menus and text fields. Being built entirely using standard web technologies, PALMYRA 2.0 runs on all major web browsers.

PALMYRA 2.0's configuration file allows it to be used for annotating features that do not commonly appear in syntactic dependency treebanks. The flexibility of not limiting the linguistic annotations to one kind of features makes the repurposing of this tool to annotate, for example, sentiment datasets, or NER gazetteers, a functionality that can be easily integrated into the tool.

We plan to continue maintaining and enhancing PALMYRA 2.0's capabilities, such as supporting multi-token words, hidden node representations, and enhanced dependencies annotation. We also plan on adding support for additional linguistic annotations, and sentence splitting. Inter-annotator agreement, user management, and automatic parsing are also features we are looking to add to PALMYRA to increase its usability in large annotation projects, particularly since we will be using PALMYRA 2.0 heavily as part of a large-scale treebanking annotation project in the near future.

Acknowledgments

This project is funded by a New York University Abu Dhabi Research Enhancement Fund. We would like to thank Jamila El-Gizuli, Ossama Obeid, Arfath Pasha for helpful conversations and interface design and implementation advice. We thank the anonymous reviewers for their feedback and comments.

References

Faisal Al-Shargi and Owen Rambow. 2015. Diwan: A dialectal word annotation tool for Arabic. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, pages 49–58, Beijing, China.

Abdulrahman Alosaimy and Eric Atwell. 2018. Web-based annotation tool for inflectional language resources. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká, 2003. *The Prague Dependency Treebank*, pages 103–127. Springer Netherlands, Dordrecht.

Marcel Bollmann, Florian Petran, Stefanie Dipper, and Julia Krasselt. 2014. Cora: A web-based annotation tool for historical and other non-standard language data. In *Proceedings of the 8th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)*, pages 86–90.

Sabine Buchholz and Erwin Marsi. 2006. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, pages 149–164, New York City, New York.

Richard Eckart de Castilho, Eva Mujdricza-Maydt, Seid Muhie Yimam, Silvana Hartmann, Iryna Gurevych, Anette Frank, and Chris Biemann. 2016. A web-based tool for the integrated annotation of semantic and syntactic structures. In *Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH)*, pages 76–84.

Nizar Habash and Ryan Roth. 2009. CATiB: The Columbia Arabic Treebank. In *Proceedings of the Joint Conference of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 221–224, Suntec, Singapore.

Jan Hajič, Otakar Smrž, Petr Zemánek, Jan Šnaidauf, and Emanuel Beška. 2004. Prague Arabic Dependency Treebank: Development in Data and Tools. In *Proceedings of the International Conference on Arabic Language Resources and Tools*, pages 110–117, Cairo, Egypt. ELDA.

Johannes Heinecke. 2019. Conllueditor: a fully graphical editor for universal dependencies treebank files. In *Proceedings of the Third Workshop on Universal Dependencies (UDW, SyntaxFest 2019)*, pages 87–93.

Talha Javed, Nizar Habash, and Dima Taji. 2018. Palmyra: A Platform Independent Dependency Annotation Tool for Morphologically Rich Languages. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Alexa Little and Stephen Tratz. 2016. Easytree: A graphical tool for dependency tree annotation. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Portorož, Slovenia.

Mohamed Maamouri, Ann Bies, Hubert Jin, and Tim Buckwalter. 2003. Arabic treebank: Part 1 v 2.0. Linguistic Data Consortium (LDC2003T06).

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Portorož, Slovenia.

Ossama Obeid, Salam Khalifa, Nizar Habash, Houda Bouamor, Wajdi Zaghouani, and Kemal Oflazer. 2018. MADARi: A Web Interface for Joint Arabic Morphological Annotation and Spelling Correction. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Petr Pajas. 2008. Tred: Tree editor. http://ufal.mff.cuni.cz/ pajas/tred.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 1094–1101, Reykjavik, Iceland.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. Brat: A web-based tool for nlp-assisted text annotation. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 102–107, Avignon, France.

Stephen Tratz and Nhien Phan. 2018. A web-based system for crowd-in-the-loop dependency treebanking. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(02):207–238.