# A Dependency Treebank for Classical Arabic Poetry

**Sharefah Al-Ghamdi**
King Saud University
sharefah@ksu.edu.sa

**Hend Al-Khalifa**
King Saud University
hendk@ksu.edu.sa

**Abdulmalik Al-Salman**
King Saud University
salman@ksu.edu.sa

## Abstract

This paper introduces the first syntactically annotated corpus for Classical Arabic poetry, a morphologically rich ancient Arabic text. The paper describes how the dependency treebank was prepared, focusing on some issues dealing with Classical Arabic poems in which syntactic constructions require special attention. We also present the results of the baseline experiments on Classical Arabic poetry dependency parsing with this treebank.

## 1 Introduction

With the massive development of natural language processing (NLP) applications and tools, treebanks (TB) (syntactically parsed text corpora) are considered an essential basic language resource. The existence of a treebank is the first step toward parser creation and evaluation for any natural language. Unfortunately, classical Arabic (CA) has only one treebank, which is for the Holy Quran text (Dukes and Buckwalter, 2010). This motivated us to contribute to the Arabic NLP resources by constructing the first Arabic Poetry Treebank (ArPoT).

CA (aka Quranic Arabic) is the standardized literary form of the Arabic language; It consists of the Holy Quran text and literary texts such as poetry, elevated prose, and oratory. However, it differs in its vocabulary and phraseology from the Modern Standard Arabic (MSA) that came with the prevalence of literacy, universal education, journalism, and written media. Moreover, CA poems are characterized by symmetry, eloquence, and rhetoric (Zwettler, 1978; Ahmed and Trausan-Matu, 2017). To maintain the rhyme and rhythm of poems, poets would violate the grammatical requirements showing, called the Poetic Necessity (Najjar, 2012). Thus, this work explores the dependency syntactic analysis of CA poems, and we expect that it would be a starting point for further studies on CA poetry parsing.

For our annotation scheme, we have chosen the part of speech (POS) tag sets, dependency labels and guidelines released by Habash et al. (2009), which have been applied during constructing Columbia Arabic Treebank (CATiB). We selected this schema based on two considerations. First, it is closer to the traditional Arabic grammar; however, it maintains the ability to do a future conversion to other different representations such as Universal dependency (UD) (Habash et al., 2009; Taji et al., 2017 ). Second, there is a publicly available parser that trained on Columbia Arabic Treebank, which we used in the initial annotation step. So that it would simplify and speed up the development process.

This paper describes the annotation process and outlines some of the issues and interesting phenomena found during the annotation of ArPoT. The rest of the paper is structured as follows: Section 2 briefly reviews the Arabic treebanks. Section 3 introduces the dataset that has been used to construct the ArPoT. Next, the annotation process is described in Section 4. Then, Section 5 discusses the challenges and issues we had tackled. Finally, we present the results of the baseline parsing experiments on our treebank in Section 6, and conclude the paper with future work in Section 7.

## 2 Related Work

Most of the well-known syntactic Arabic TBs are constructed for MSA, such as: constituency Penn Arabic Treebank (PATB) by Maamouri et al. (2004), Prague Arabic Dependency Treebank (PADT) by Hajic et al. (2004) and dependency Columbia Arabic Treebank (CATiB) by Habash et al. (2009). For

This work is licensed under a Creative Commons Attribution 4.0 International License. License details: http://creativecommons.org/licenses/by/4.0/.

CA, Quranic Arabic Dependency Treebank (QADT) of the Holy Quran text by Dukes et al. (2010) is the only known TB. Its linguistic framework is termed a hybrid dependency-phrase structure grammar and focuses more on visualizing the grammatical annotation. The syntactic layer of QADT covers 37,578 words (~ 49% of the full Quranic text) (Dukes and Habash, 2011).

In addition to the above, several TBs for Arabic dialects have been produced, such as: Levantine Arabic Treebank (LATB) (Maamouri et al., 2006), Egyptian Arabic Treebank (Maamouri et al., 2014), and dependency treebank of Arabic tweets (Albogamy et al., 2017). However, there is no Arabic poetry Treebank that has been created yet.

Example 1: يا عَينِ جُـودي بِالدموع المُستَهِلَّاتِ السـوَافِحْ

*yaA ðayni juwudiy bi Alddmwuʕ Almusthil~aAti AsswaAifH.*

"Oh eye, be generous with shedding and pouring tears"

Figure 1. Classical Arabic verse

## 3 Dataset Preparation

### 3.1 Poems collection

Poems in ArPoT have been collected initially from Arabic literary poems websites such as ADAB[1] and ALDIWAN[2]. They offer thousands of written poems for transmitted oral poetry from the earliest pre-Islamic era until today. For this work, we only focus on Classical poetry, which commonly refers to old oral poems transmitted from the early (6th to 13th) centuries. The selected verses are diverse; they are from more than 775 poems for 34 different Classical eras poets. Our final corpus contains 2685 verses (35,459 tokens).

Classical verses consist of two parts that follow the metric rule, which is not the case of modern free poetry verses. Figure 1 shows example 1 for Classical verse along with its transliteration[3] and English translation. In addition, Table 1 lists a word for word glosses for all examples of CA verses that used in this work.

| Word | Gloss | Word | Gloss | Word | Gloss | Word | Gloss |
|---|---|---|---|---|---|---|---|
| **Example 1** | | **Example 3** | | **Example 4** | | **Example 5** | |
| يا | oh | و | and | فإن | if | خلاءين | empty |
| عين | eye | رب | rare | أبك | I cry | بعد | after |
| جودي | be generous | شمعة | candle | قومي | my people | الحلم | meekness |
| ب | with | مزقت | tears up | يا | oh | و | and |
| الدموع | tears | ثوب | garment | نوار | proper name | الجهل | rudeness |
| المستهلات | shedding | الظلام | darkness | ف | so | في | in |
| السوافح | pouring | ب | with | إني | I | هما | them |
| **Example 2** | | ما | that | أرى | see | و | and |
| فيضا | flood | بثت | spread | مسجدي | mosques | بعد | after |
| كما | as | من | of | هم | their | عبابي | roar |
| فاض | flood | النور | light | من | of | الندى | rain |
| الغروب | pails | في | in | هم | them | المتدافع | rush |
| المترعات | generous | الأرجاء | surroundings | ك | as | - | - |
| من | from | متسعا | widely | البلاقع | desolate home | - | - |
| النواضح | camels | | | | | | |

Table 1. A word for word glosses for all examples of CA verses.

[1] https://www.adab.com/

[2] https://www.aldiwan.net/

[3] All Arabic transpirations are according to (Habash et al., 2007) transliteration scheme.

### 3.2 Preprocessing

In this stage, we have prepared the poetry text for annotation. After verses had been scraped from the webpages into text files, we concatenated the two parts of each verse using our implemented java code. Then, the spelling mistakes were corrected manually. During this phase, we removed the identical verses which are accidentally repeated on the websites. Also, there were some verses that were clearly broken and had several missing words shown as dots. The syntactic structure analysis for such verses was not able, so we removed them from the dataset. The "التطويل/ *Atatweel*/ Kashedah" has been removed as well. Since the verses are from transmitted old oral classical poems, the punctuation is uncommon and very rare. Therefore, the punctuation has been eliminated in this dataset.

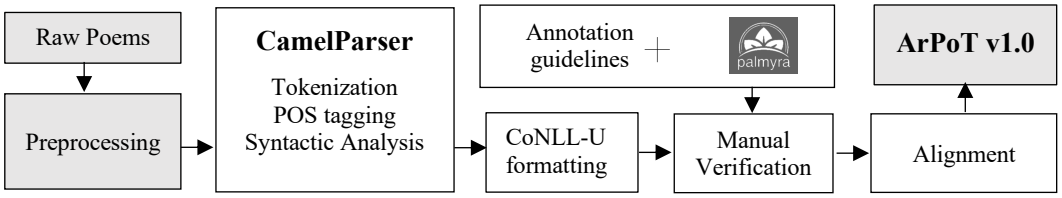



Figure 2. Annotation Process of ArPoT

## 4 Annotation process

To maintain the annotation process cost (in terms of money and time), we considered the strategy of automatic annotation followed by manual correction instead of creating the Arabic Poetry Treebank from scratch. Figure 2 shows the flowchart for the annotation steps.

### 4.1 Initial automatic annotation

After reviewing the dependency parsers for the Arabic language, we chose the CamelParser (Shahrour et al., 2016) for the initial automatic annotation. It is a publicly available system for Arabic syntactic dependency analysis that is trained on CATiB (Habash et al., 2009). Although it was developed on MSA, its initial parsing shortened the annotation process. It applies the tokenization and POS tagging with reasonable accuracy, and it constructs the syntactic trees we provide to the annotators for manual corrections.

### 4.2 File Format transformation

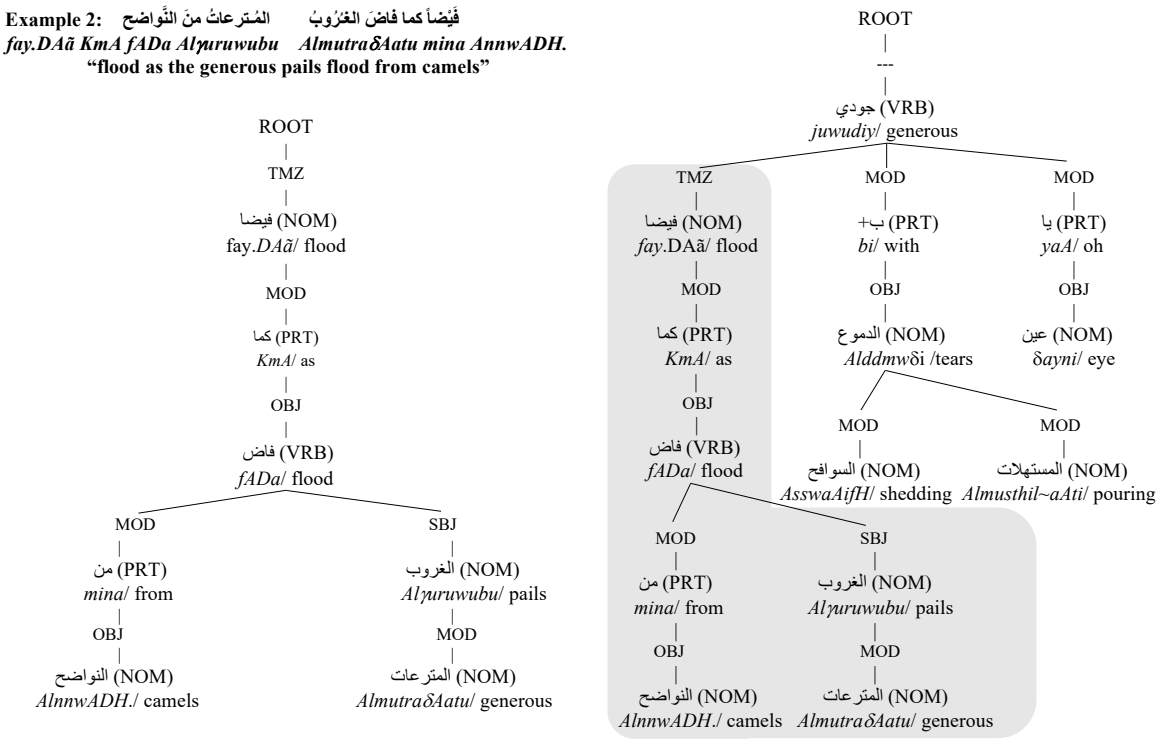
The CamelParser offers the output in different formats. However, we decided to produce a valid CoNLL-U format that can train most of the current parsers and tree visualization tools.

### 4.3 Manual Verification

While CamelParser was trained on MSA corpus, it handles the CA poems with tokenization, POS tagging, and dependency relation labeling errors. The manual correction phase starts with correcting the tokenization errors to give the ability to calculate the Inter Annotation Agreement (IAA) between the annotators. Three paid annotators have carried out this phase. They were Arabic native speakers and linguistic experts. PALMYRA, a graphical dependency tree visualization and editing software, has been used for this step (Javed et al., 2018; Taji and Habash, 2020). The manual correction was completed within four months.

CamelParser's tokenization was incorrect for around 52% of words. Thus, to report its accuracy on the CA poems, we compared the verses that have true tokenization with the final gold annotated verses which were verified by annotators. The result gave us 55% Exact Match (EM) – the percentage of tokens with correct POS tags, heads and relation labels.

We used the Kappa coefficient for IAA between annotators (Cohen, 1960). The first part of the data, which covers ~ 83% of the corpus, was revised by two full-time annotators with a 0.97 kappa value on 10% of this part. To check the agreement, the second part of the data, which covers ~17% of the corpus plus 10% of the first part, has been revised by a third annotator. The result of IAA was 0.85 for the




(a) Before Alignment (b) After Alignment (Example 1 and 2)

Figure 3. The Alignment for two contiguous verses that have dependency relation in between.

kappa coefficient; then, after the second round of revision, the IAA increased to 0.96. The small size of the data and the few tags included in the guidelines positively affected the agreement score. Moreover, the CATiB annotator's manual provided to the annotators decreased the disagreement cases.

### 4.4 Alignment

Like the Quranic text, CA poetry consists of verses, which might be one complete sentence. However, the verse may act as a modifier for prior or posterior verse so that the complete sentence would be in two, three or more verses. Although sentence boundary detection is essential for NLP, there is no available system that could detect the sentence boundaries of the CA poetry. Therefore, we concatenated the verses' dependency trees for the same sentence during the alignment phase. Moreover, delaying the alignment step after the manual verification has simplified the visualization during the correction, while large trees after alignment become more complicated.

During the manual verification, we added a syntactic label to the root in case it has a relation with another verse and recorded the index of the parent token. Then, in the alignment phase, we just connected the related verses to produce one complete sentence in one syntactic tree. This broad tree shows the whole meaning that the verses will provide. For example, Figure 3 (a) shows the dependency tree for verse example 2 which is the subsequent of verse example 1 in the same poem (shown in Figure 1). The head token of example 2 syntactic tree has TMZ "تمييز/ tamyiz/ specification" relation with the word "جودي/ juwudy/ be generous" in the verse example 1. After the alignment for these contiguous verses to form a complete sentence, the connected tree for verse example 2 is shown with gray shade in Figure 3 (b).

## 5 CA Poetry Annotation Issues

Although the main guiding principle followed during the construction of ArPoT v1.0 serves as a general guideline, some syntactic structure issues and phenomena of CA poetry have been encountered. In the following, we present two categories of issues along with the solution strategies we applied.

**Example 3:** و رُبَّ (*) شمعةٍ مزّقت ثوبَ الظلامِ بما بثَّت من النّورِ في الأرجاءِ متسعا

*wa šamδahī maz~aqat θawba AĎĎalaAami bi maA baθ~at mina Annwri fi AlÂrjaA'i mutasaδaA*

**"Rarely that a candle tears up the darkness garment, with its light that has been spread widely in the surroundings"**

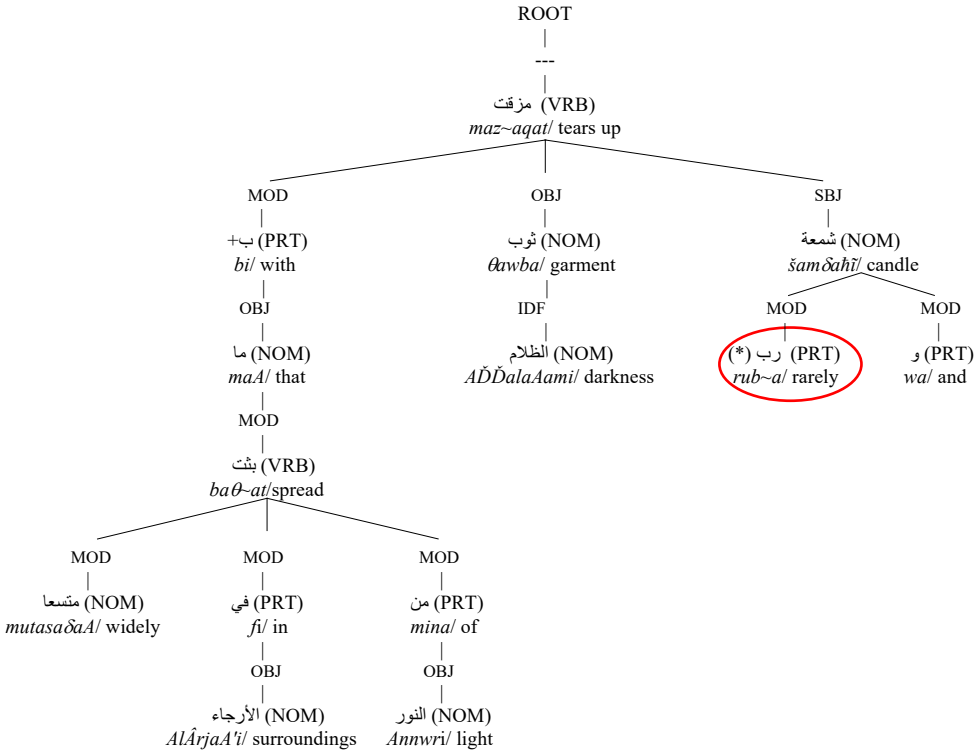



Figure 4. Dependency tree for the verse example 3 that shows the elision case.

## 5.1 Elision and Reconstruction

Linguistic deletion or elision (الحذف/ AlHaðf) is a common syntax feature in Classical Arabic language, mainly in Quranic text and poetry, where a major element of the sentence is omitted but often implied and recovered based on contextual clues (Suleiman, 1990). On the other hand, the process of allowing implicit syntactic roles to be made explicit is known as reconstructing (التقدير/ Altaqdir). Adding the ellipse to the sentence structure through reconstruction provides new information or meaning which unable to clarify except with (التقدير/ Altaqdir). Thus, we followed Dukes and Buckwalter (2010) in their treatment of elision cases by showing the empty nodes in the syntactic tree. In ArPoT, only 0.6% of the tokens are ellipses. During the manual verification, annotators added those dropped words manually to the treebank in the form (word (*)).

Ellipsis in ArPoT includes different categories such as: verbs, subject of nominal sentences, and particles deletion. For example, the deleted preposition (رُبَّ/ rub~a) has been added to the verse syntactic tree of verse example 3 as shown in Figure 4. In this example, (رُبَّ/ rub~a) gives the meaning of (التقليل/ taqliyl/ reduction), which means it is rare that one candle can give that much light.

The preposition (رُبَّ/ rub~a) is obviously used in CA. In the Arabic language, it is known as a semi-extra preposition (حرف شبيه بالزائد). This means that it illustrates the sentence's meaning, but it does not relate to its object like other original prepositions. Thus, we attached it under its object with MOD relation.

**Example 4:** فَإنْ أبْكِ قَوْمي يا نَوَارُ فإنّني أرَى مَسْجِدَيهِمْ مِنهمُ كالبَلاقِعِ

***faĂn Âbki qawmiy yaA naw~aAru faĂn~aniy Âraý masjidayhim minhumu kaAlbalaAaqiδi***

**"If I cry my people, oh Nawwar, that because I see their mosques as desolate home"**

**Example 5:** خَلاءَينِ بَعدَ الحِلْمِ وَالجَهلِ فيهما وَ بَعْدَ عُبابيِّ النّدَى المُتَدافِعِ

***xalaA'ayni baδda AlHilmi wa Aljahli fihimaA wa baδda δubabiy~ Annadý AlmutadaAfiδi***

**"I see them empty, after meekness and rudeness there, and after the roar of heavy rain"**

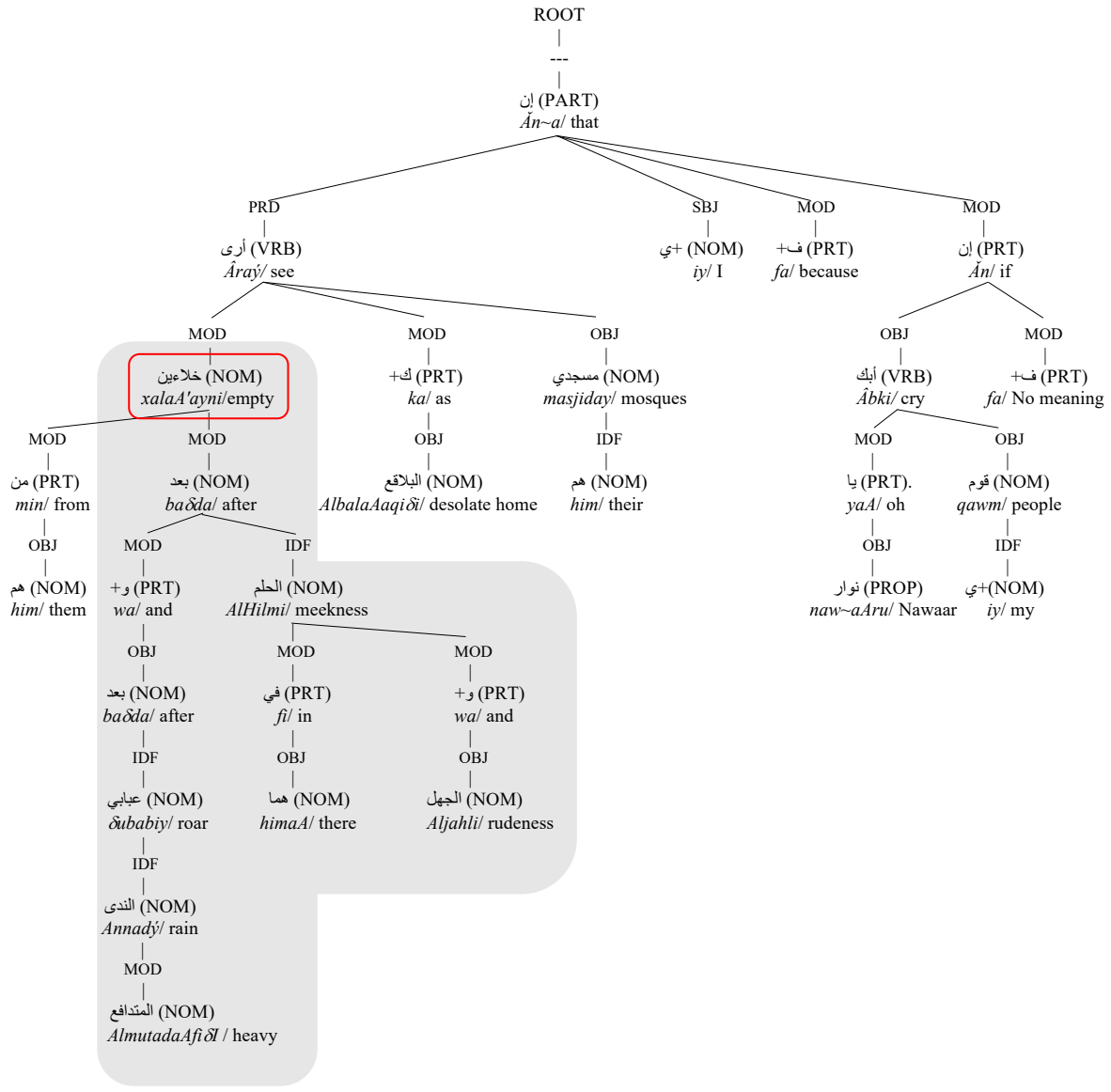



Figure 5. Dependency tree for two verses with dual syntactic relations

## 5.2 Broken and Complex Structure

As mentioned earlier in this paper, the selected poems were transmitted from an earlier era, using ancient CA. Since then, Arabic books have been published for each poet to collect and interpret their poems which guide the annotators during the manual verification work. These references show that some transmitted verses are broken, with missing parts or words. Also, some poems were incomplete.

For example, the poem starts with a verse that should be dependent on another unavailable previous verse. Therefore, broken and incomplete verses have been excluded from the corpus.

Although most of the related verses were sequential, we found more complicated cases that brought us to the alignment step in the annotation process. For example, the two contiguous verses (example 4 and 5) shown in Figure 5 have dual relations in both directions. Each verse includes a token headed by a parent token in the other verse. To illustrate the relations, we shaded all tokens in the dependency tree for the verse example 5. Its first word (خلاءين/ xalaA'ayni / empty), bordered by a red line, headed by a token and it heads another token that both are in the verse example 4. Placing the two verses in one syntactic tree shows the full structure that cannot be represented by an individual tree for each verse.

## 6 Evaluation

To test the effectiveness of the proposed annotations, we carried out some parsing experiments using dependency parsing models that adapted two different neural-based architectures. They achieved remarkable accuracies in dependency parsing for multilingual treebanks. The first model is the novel left-to-right dependency parser based on pointer networks developed by Fernández-González and Gómez-Rodríguez (2019). The second is the accurate and straightforward sequence tagging parser for Vacareanu et al. (2020).

We have split the ArPoT v1.0 randomly, dedicating 80% of the dataset for training. Due to the small size of the treebank and for a more confident result, 12% was used for testing and 8% for development. Words in this version are without "التشكيل/ *Taskeel*/ Diacritics". We are planning to include them in the future. The treebank is available here: https://github.com/arpot-ksu.

| **Model** | **Method** | **UAS** | **LAS** |
|---|---|---|---|
| Fernández-González and Gómez-Rodríguez (2019) | Transition based | 81.52 | 75.25 |
| Vacareanu et al. (2020) | Labeling | 78.43 | 70.95 |

Table 2: Evaluation results on the ArPoT 1.0 test set for the two neural-based parsing models[4].

The parsing results are found in Table 2. We used the standard metrics for dependency parsing, Labelled Attachment Score (LAS) and Unlabeled Attachment Score (UAS). The reported scores are the average of three runs.

The accuracy of the transition-based pointer networks model is UAS of 81.52% and LAS of 75.25%, whereas the tagging model obtains a UAS of 78.43% and LAS of 70.95%. Overall, the results are promising for small treebank such as ArPoT. However, a more in-depth error analysis would be necessary to better understand the challenges of parsing models and provide an accurate analysis of CA poetry.

## 7 Conclusion and Future Work

This work described the first syntactically annotated corpus for Classical Arabic poetry. The treebank consists of 35,460 tokens. In addition to the annotation process, this paper discussed some issues during the development of the ArPoT treebank. We also posed an initial set of experiments with two neural-based parsing systems that show the appropriate settings of our treebank.

Future work plans will include more verses in our treebank and conduct a comparison study with other MSA treebanks. Also, we intend to further investigate the dependency parsing approaches on CA poetry. Besides, ArPoT might help in building a sentence boundary detection tool, which would be beneficial in our research.

[4] For both parsers we used the predefined settings.

## Reference


Ahmed, Munef Abdullah, and Stefan Trausan-Matu. 2017. Using Natural Language Processing for Analyzing Arabic Poetry Rhythm. In *16th Networking in Education and Research RoEduNet International Conference,* 1–5, Targu-Mures, Romania. IEEE.

Albogamy, Fahad, Allan Ramsay, and Hanady Ahmed. 2017. Arabic tweets treebanking and parsing: A bootstrapping approach. In *Proceedings of the Third Arabic Natural Language Processing Workshop*, pp. 94-99.

Cohen, Jacob. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement,* 20 (1):37-46.

Dukes, Kais, and Nizar Habash. 2011. One-step statistical parsing of hybrid dependency-constituency syntactic representations. In *Proceedings of the 12th International Conference on Parsing Technologies*, pp. 92-103.

Dukes, Kais, and Tim Buckwalter. 2010. A dependency treebank of the Quran using traditional Arabic grammar. In *2010 the 7th International Conference on Informatics and Systems (INFOS)*, pp. 1-7. IEEE.

Fernández-González, Daniel, and Carlos Gómez-Rodríguez. 2019. Left-to-Right Dependency Parsing with Pointer Networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* pp. 710-716.

Habash, Nizar, Abdelhadi Soudi, and Timothy Buckwalter. 2007. On Arabic transliteration. In *Arabic computational morphology*, pp. 15-22. Springer, Dordrecht.

Habash, Nizar, Reem Faraj, and Ryan Roth. 2009. Syntactic Annotation in the Columbia Arabic Treebank. In *2nd International Conference on Arabic Language Resources and Tools MEDAR*, Cairo, Egypt.

Hajic, Jan, Otakar Smrz, Petr Zemánek, Jan Šnaidauf, and Emanuel Beška. 2004. Prague Arabic Dependency Treebank: Development in Data and Tools. In *The NEMLAR International Conference on Arabic Language Resources and Tools*, 110–117.

Javed, Talha, Nizar Habash, and Dima Taji. 2018. Palmyra: A platform independent dependency annotation tool for morphologically rich languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Maamouri, Mohamed, Ann Bies, Seth Kulick, Michael Ciul, Nizar Habash, and Ramy Eskander. 2014. Developing an Egyptian Arabic Treebank: Impact of Dialectal Morphology on Annotation and Tool Development. In *LREC*, pp. 2348-2354.

Maamouri, Mohamed, Ann Bies, Tim Buckwalter, Mona Diab, Nizar Habash, Owen Rambow, and Dalila Tabessi. 2006. Developing and using a pilot dialectal Arabic treebank. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation* (LREC'06).

Maamouri, Mohamed, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In *NEMLAR conference on Arabic language resources and tools*, Vol. 27, 466-467.

Ma, Xuezhe, Zecong Hu, Jingzhou Liu, Nanyun Peng, Graham Neubig, and Eduard Hovy. 2018. Stack-Pointer Networks for Dependency Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* pp. 1403-1414.

Najjar, Manal. 2012. Poetic Necessity Between the Syntax of a Sentence and the Syntax of a Text. *GSTF Journal of Law and Social Sciences (JLSS),* 2(1):322.

Shahrour, Anas, Salam Khalifa, Dima Taji, and Nizar Habash. 2016. Camelparser: A system for Arabic syntactic analysis and morphological disambiguation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pp. 228-232.

Suleiman, Saleh M 1990. The semantic functions of object deletion in classical Arabic. *Language Sciences*, 12(2-3): 255-266

Taji, Dima, and Nizar Habash. 2020. PALMYRA 2.0: A Configurable Multilingual Platform Independent Tool for Morphology and Syntax Annotation. In *Proceedings of the Fourth Workshop on Universal Dependencies (UDW 2020)*, pp. 168-177.

Taji, Dima, Nizar Habash, and Daniel Zeman. 2017. Universal dependencies for Arabic. In Proceedings of the *Third Arabic Natural Language Processing Workshop*, pp. 166-176.

Vacareanu, Robert, George Caique Gouveia Barbosa, Marco A. Valenzuela-Escarcega, and Mihai Surdeanu. 2020. Parsing as tagging. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 5225-5231.

Zwettler, Michael. 1978. *Oral tradition of classical Arabic poetry: its character and implications*. The Ohio State University Press.