TECHNICAL REPORT

No. CCLS-09-01

Title: CATiB: The Columbia Arabic Treebank

Authors: Nizar Habash and Ryan Roth

CATiB: The Columbia Arabic Treebank

Nizar Habash and Ryan Roth
Center for Computational Learning Systems
Columbia University, New York, USA
{habash,ryanr}@ccls.columbia.edu

Abstract

The Columbia Arabic Treebank (CATiB) is a resource for Arabic parsing. CATiB contrasts with previous efforts on Arabic treebanking and treebanking of morphologically rich languages in that it encodes less linguistic information in the interest of speedier annotation of large amounts of text. This paper describes CATiB's representation and annotation procedure, and reports on achieved inter-annotator agreement and annotation speed.

1 Introduction

Collections of manually-annotated morphological and syntactic analyses of sentences (treebanks) are an important resource for building syntactic models for statistical parsing or syntax-aware approaches to applications such as machine translation. Rich treebank annotations have also been used for a variety of applications such as tokenization, diacritization, POS tagging, morphological disambiguation, base phrase chunking, and semantic role labeling. Under time restrictions, the creation of a treebank faces a tradeoff between linguistic depth and treebank size, especially for morpho-syntactically complex languages such as Arabic or Czech. Linguistic depth provides the advantage of providing many linguistic features that may be useful for a variety of applications. This comes at the cost of slower annotation as a result of longer guidelines and more intense annotator training. As a result, the deeper the annotation, the slower the annotation process and the smaller the size of the treebank. And consequently, the less data there is to train tools that can benefit from more data.

In this paper, we present the Columbia Arabic Treebank (CATiB), a resource built with faster annotation speed and shallower linguistic depth in mind.

2 Previous Work

Much work has been done in the area of building treebanks, most prominent amongst is the English PennTreebank (Marcus et al., 1993). In the case of Arabic, two important treebanking efforts exist: the Penn Arabic Treebank (PATB) (Maamouri et al., 2004) and the Prague Arabic Dependency Treebank (PADT) (Smrž and Hajič, 2006). The main difference between these two resources is the linguistic representation: PATB uses phrase structure and PADT uses dependency representation. Both of these efforts employ complex and very rich linguistic representations that require a lot of human training (on the order of 6 months to a year per annotator). Due to space limitations, we compare our work with the PATB. A comparison with PADT appears in (Habash et al., 2009).

3 CATiB: Columbia Arabic Treebank

3.1 Motivation

CATiB contrasts with PATB and PADT in putting an emphasis on faster production with some constraints on linguistic depth. Two basic ideas inspire the CATiB approach. First, CATiB avoids annotation of redundant linguistic information. For example, nominal case markers in Arabic have been shown to be automatically determinable from syntax and word morphology and needn't be manually annotated (Habash et al., 2007a). However,there is some information in CATiB that is not easily recoverable, such as phrasal co-indexation and full lemma disambiguation. Second, CATiB uses a linguistic representation and terminology inspired by Arabic's long tradition of syntactic studies. This makes it easier to train annotators without being restricted to hire annotators who have degrees in linguistics. CATiB uses an intuitive dependency representation and relational labels inspired by Arabic grammar such as tamyiz (specification) and idafa (possessive construction) in addition to universal predicate-argument structure labels such as subject, object and modifier.

3.2 Syntactic Representation

CATiB uses the same basic tokenization scheme used by PATB and PADT. However, the CATiB POS tag set is much smaller. Whereas in practice PATB uses 420 tags (on tokenized words) specifying every aspect of Arabic word morphology such as definiteness, gender, number, person, mood, voice and case, CATiB uses 6 POS tags: **NOM** (non-proper nominals including nouns, pronouns, adjectives and adverbs), **PROP** (proper nouns), **VRB** (verbs), **VRB-PASS** (passive-voice verbs), **PRT** (particles such as prepositions or conjunctions) and **PNX** (punctuation). The eight CATiB relation labels are: **SBJ** (subject of verb or topic of simple nominal sentence), **OBJ** (object of verb, preposition, or deverbal noun), **TPC** (topic in complex nominal sentences containing an explicit pronominal referent), **PRD** (predicate marking the complement of the extended copular constructions for *kAn*[1] كان واخواتها and *An* ان واخواتها), **IDF** (relation between the possessor [dependent] to the possessed [head] in the idafa/possesive nominal construction), **TMZ** (relation of the specifier [dependent] to the specified [head] in the tamyiz/specification nominal constructions), **MOD** (general modifier of verbs or nouns), and **—** (marking *flatness* inside constructions such as first-last proper name sequences). This relation label set is much smaller than the twenty or so dash-tags used in PATB to mark syntactic and semantic functions. No empty categories and no phrase co-indexation are made explicit. No semantic relations (such as time and place) are encoded.

Figure 1 compares the representation of the same sentence in PATB and CATiB. A detailed comparison of PATB and CATiB annotations will appear in a future publication. A detailed discussion of CATiB guidelines and further comparison with PATB appears in (Habash et al., 2009).

[1]Arabic transliterations are provided in the Habash-Soudi-Buckwalter transliteration scheme (Habash et al., 2007b).

3.3 Annotation Procedure

Although CATiB is independent of previous annotation projects, it builds on existing resources and lessons learned. For instance, CATiB's pipeline uses PATB-trained tools for tokenization, POS-tagging and parsing. We also use an annotation interface developed in coordination with the PADT (see below). Using these tools allowed annotation production to start quickly after the project began. In the first month of the project, we began writing the annotation manual (guided by the wonderfully detailed manual of the PATB for coverage). During this period, we also reconfigured and tested tools and automatic pipelines and conducted interviews to hire annotators. Annotator training took place over two months (150 hrs/annotator on average). By the end of training, the manual was stabilized and the production phase started. We hit our production target of 200K words within five months (including the Holy month of Ramadan, where the annotation speed was much lower).

Below we describe our pipeline in more detail including the different resources we used.

Data Preparation The data to annotate is split into *batches* of 3-5 documents each, with each document containing 15-20 sentences (400-600 tokens). Each annotator can work on one batch at a time. This procedure and the size of the batches was determined to be optimal both for the software we used and the annotators' productivity.To track the annotation quality, several key documents are selected for inter-annotator agreement (IAA) checks. The IAA documents are chosen to cover a range of news sources and to be of average document size. These documents (collectively about 10% of the planned token volume) are seeded throughout the batches. Every annotator eventually annotates each one of the IAA documents, but are never told which documents are for IAA.

Tokenization and POS Tagging We use the MADA&TOKAN toolkit (Habash and Rambow, 2005) for POS tagging and tokenization. Tokeniza-

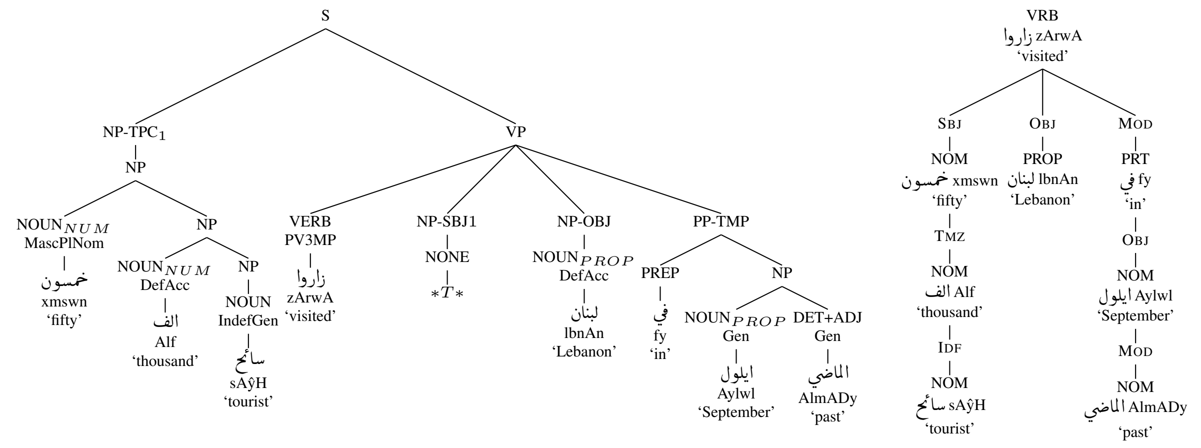

Figure 1: Comparing phrase structure in the Penn Arabic Treebank (left) to CATiB (right) for the sentence خمسون الف سائح زاروا لبنان في ايلول الماضي *xmswn Alf sAŷH zArwA lbnAn fy Aylwl AlmADy* '50 thousand tourists visited Lebanon last September.'

tion F-score is 99.7%. and POS tagging accuracy (on the CATiB POS tagset; with gold tokenization) is above 97.7%. To reduce tokenization errors that would make annotation difficult, all the tokenizations are manually checked and corrected by the annotation supervisor. New POS tags are assigned manually for each (and only) corrected tokenization. Full POS tag correction is done as part of the Annotation step (see below). The speed of this step is well over 6K tokens/hour.

Parsing Initial dependency parsing in CATiB is conducted using the Malt Parser software (Nivre et al., 2007). An initial parsing model was built using an automatic constituency-to-dependency conversion of a section of PATB *Part 3* (PATB3-Train, 339K tokens). The quality of the automatic conversion step is measured against a hand-annotated version of an automatically converted held-out section of PATB3 (PATB3-Dev, 31K tokens). The results are 87.2%, 93.16% and 83.2% for attachment (**ATT**), label (**LAB**) and labeled attachment (**LABATT**) accuracies, respectively. These numbers are 95%, 98% and 94% (respectively) of the IAA scores on that set. Details of the conversion process will be discussed in a separate publication. At the production midpoint another parsing model was trained by adding all the CATiB annotations generated up to that point (513K tokens total). An evaluation of the parser against the CATiB version of PATB3-Dev shows the **ATT**, **LAB** and **LABATT** accuracies are 81.7%, 91.1% and 77.4% respectively.

Annotation CATiB uses the TrEd tool (Pajas, 2008) as a visual interface for annotation. The parsed trees are converted to TrEd format and delievered to the annotators. The annotators are asked to only correct the tree structure, POS and labels. Once annotated (i.e. corrected), the documents were returned to be packaged for release.

Our five annotators and their supervisor are all educated native Arabic speakers. Annotators are hired on a part-time basis and are not required to be onsite. The annotation files are exchanged electronically. This arrangement allows more annotators to participate, and reduced logistical problems. However, having no full-time annotators limits the overall weekly annotation rate.

4 Results

Data Sets CATiB annotated data is taken from the following LDC-provided resources:[2] LDC2007E46, LDC2007E87, GALE-DEV07, MT05 test set, MT06 test set, and PATB (part 3). Headlines, datelines and bylines are parsed, but not annotated and some sentences are excluded for excessive (>300 tokens) length and formatting problems. Collectively, over 272K tokens (227K words) of data were annotated, not counting IAA duplications, omissions, headlines, and annotations done for purposes of de-

[2]http://www.ldc.upenn.edu/

IAA Set	Sents	POS	ATT	LAB	LABATT
PATB3-Dev	All	98.6	91.5	95.3	88.8
	≤ 40	98.7	91.7	94.7	88.6
PROD	All	97.6	89.2	93.0	85.0
	≤ 40	97.7	91.5	94.1	87.7

Table 1: Average pairwise inter-annotator agreement accuracies for 5 annotators. The **Sents** column indicates which sentences were evaluated, based on token length. The sizes of the sets are 2.4K (PATB3-Dev) and 3.8K (PROD) tokens.

IAA File	Toks/hr	POS	ATT	LAB	LABATT
HI	398	97.0	94.7	96.1	91.2
HI-S	956	97.0	97.8	97.9	95.7
LO	476	98.3	88.8	91.7	82.3
LO-S	944	97.7	91.0	93.8	85.8

Table 2: Highest and lowest average pairwise inter-annotator agreement accuracies for 5 annotators achieved on a single document – before and after serial annotation. The "-S" suffix indicates the result after the second annotation.

velopment. These packaged data sets will be made publicly available through the LDC.

Annotator Speeds Our POS and syntax annotation rate is 540 tokens/hour (with some reaching rates as high as 715 tokens/hour). However, due to the current part-time arrangement, annotators worked an average of only 6 hours/week, which meant that data was annotated at an average rate of 15K tokens/week.

Basic Inter-Annotator Agreement We present IAA scores for **ATT**, **LAB** and **LABATT** on IAA subsets from two data sets in Table 1: PATB3-Dev is based on an automatically converted PATB set (introduced in Section 3.3) and PROD refers to all the new CATiB data. We compare the IAA scores for all sentences and for sentences of token length $\leq$ 40 tokens. The IAA scores in PROD are lower that PATB3-Dev, this is understandable given that the error rate of the conversion from a manual annotation (starting point of PATB3-Dev) is lower than parsing (starting point for PROD). Length seems to make a big difference in performance for PROD, but less so for PATB3-Dev, which makes sense given their origins. Annotation training did not include very long sentences. Excluding long sentences during production was not possible because the data has a high proportion of very long sentences: for PROD set, 41% of sentences had $\geq$ 40 tokens and they constituted over 61% of all tokens.

The best reported IAA number for PATB is 94.3% F-measure after extensive efforts (Maamouri et al., 2008). This number does not include dashtags, empty categories or indices. Our numbers cannot be directly compared to their number because of the different metrics used for different representations.

Serial Inter-Annotator Agreement We test the value of *serial annotation*, a procedure in which the output of annotation is passed again as input to another annotator in an attempt to improve it. The IAA documents with the highest (HI, 333 tokens) and lowest (LO, 350 tokens) agreement scores in PROD are selected. The results, shown in Table 2, indicate that serial annotation is very helpful reducing **LABATT** error by 20-50%. The reduction in the LO is not as large as that in HI unfortunately. The second round of annotation is almost twice as fast as the first round. The overall reduction in speed (end-to-end) is only around 30%.

Disagreement Analysis We conducted an error analysis of the basic-annotation disagreements in HI and LO. The two sets differ in sentence length, source and genre: HI has 28 tokens/sentence and contains AFP general news, while LO has 58 tokens/sentence and contains Xinhua financial news. The most common POS disagreement in both sets is NOM/PROP confusion, a common issue in Arabic POS tagging in general. The most common attachment disagreements in LO are as follows: prepositional phrase (PP) and nominal modifiers (8% of the words had at least one dissenting annotation), complex constructions (dates, proper nouns, numbers and currencies) (6%), subordination/coordination (4%), among others. The respective proportions for HI are 5%, 5% and 1%. Label disagreements are mostly in nominal modification (MOD/TMZ/IDF/—) (LO 10%, HI 5% of the words had at least one dissenting annotation).

The differences in error analysis between HI and LO seem to primarily correlate with length difference and less with genre and source differences.

5 Conclusion and Future Work

We presented CATiB, a treebank for Arabic built with faster annotation speed and lighter linguistic depth in mind. In the future, we plan to extend our annotation guidelines focusing on longer sentences and specific phenomena and introduce serial annotation as a standard part of the annotation pipeline.

Acknowledgements

We would like to thank all the annotators who worked very hard on creating CATiB. We would like to thank Owen Rambow and Otakar Smrž for helpful discussions. We would like to thank the GALE Banks committee members for their support and guidance. This work has been supported by Defense Advanced Research Projects Agency Contract No. HR0011-08-C-0110. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

References

N. Habash, R. Faraj and R. Roth. 2009. Syntactic Annotation in the Columbia Arabic Treebank. In *Proc.* of Arabic Language Resources and Tools (MEDAR).

N. Habash and O. Rambow. 2005. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proc.* of ACL'05.

N. Habash, R. Gabbard, O. Rambow, S. Kulick, and M. Marcus. 2007a. Determining case in Arabic: Learning complex linguistic behavior requires complex linguistic features. In *Proc.* of EMNLP'07.

N. Habash, A. Soudi, and T. Buckwalter. 2007b. On Arabic Transliteration. In A. van den Bosch and A. Soudi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Springer.

M. Maamouri, A. Bies, and T. Buckwalter. 2004. The Penn Arabic Treebank: Building a large-scale annotated Arabic corpus. In *Proc.* of NEMLAR'04.

M. Maamouri, A. Bies, and S. Kulick. 2008. Enhancing the Arabic Treebank: A Collaborative Effort toward New Annotation Guidelines. In *Proc.* of LREC'08.

M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2).

J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kubler, S. Marinov, and E. Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2).

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

O. Smrž and J. Hajič. 2006. The Other Arabic Treebank: Prague Dependencies and Functions. In A. Farghaly, editor, *Arabic Computational Linguistics: Current Implementations*. CSLI Publications.