Parsing Tweets into Universal Dependencies

Anonymous ACL submission

Abstract

1 Introduction

Analyzing the syntax of tweets is challenging for traditional NLP tools because most of the tweet texts are informal and noisy.

In this paper, we propose to parse the tweets in the convention of universal dependencies and built up the whole pipeline to parse the tweets from the raw text form.

Motivation notes: explain why UD is attractive (and more attractive than what previous work used, which was based on YM). one reason is that it might allow sharing of texts annotated in different genres (or even languages, cite Waleed). but universal annotation schemes are challenging! this paper contends with the challenges of applying UD to social media texts.

[Points that were in older text that I want to keep around: —NAS]

- Despite the informal nature of tweets, we argue that they basically follow the same grammar as their formal language counterpart.
- UD was designed to handle the variant syntactic phenomenons across different languages and we expect its 'context head' principle will bring convenience to the parsing of informal and even ungrammatical tweets. [Y/L I STILL FEEL WE SHOULD GIVE A REASON WHY WE CHOOSE UD. ONE EXAMPLE IS THAT TWITTER USERS TEND TO LEAVE OUT THE COPULA VERB AND IT WILL BE MESSY FOR THE 'FUNCTION HEAD' PRINCIPLE TO DEAL WITH THIS CASE. ONE OBSERVATION IS THAT INFORMAL TOKENS/SYNTAX HAPPENS MORE ON FUNCTION WORDS RATHER THAN CONTENT WORD.] [maybe we could argue that, if UD is designed to work for many languages, it should also be able to work for dialects, including AAVE, and informal registers. YM was not designed for cross-linguistic applicability. —NAS]

Contribution of this paper includes:

- We create a new version of tweet Treebank Tweebank 2.0
- We propose a neural network method to parse tweets into universal dependencies
- We study the adaptation of universal dependencies for analyzing tweets

[not clear to me: did we reannotate the POS corpus? was that automatic or manual? I wasn't sure what to do with the POS section below. —NAS]

2 Annotation

[need to introduce this section and explain the structure. I'm not crazy about what I have now, but I like it better than what we had before, which seemed to keep repeating the same ideas in different contexts. the flow still seems wrong; we start by talking about dependencies, then take a detour to POS, then come back to dependencies. —NAS]

2.1 Background: TWEEBANK

The annotation effort we describe stands in contrast to previous work by Kong et al. (2014). Their aim was the rapid development of a dependency parser for tweets, and to that end they contributed a new annotated corpus, TWEEBANK, consisting of 929 tweets. Their annotations added a layer to a portion of the data annotated with part-of-speech tags by Gimpel et al. (2011) and Owoputi et al. (2013). They also contributed a system for parsing; we defer discussion of their parser to §3.

Kong et al.'s rapid, small-scale annotation effort was heavily constrained. It was carried out mostly by non-native speakers, in a very short amount of time (a day). Driven both by the style of the text they sought to annotate and by exigency, some of their annotation conventions included:

- Allowing an annotator to exclude tokens from the dependency tree. A clear criterion for exclusion was not given, but many tokens were excluded because they were deemed "non-syntactic."
- Allowing an annotator to merge a multiword expression into a single node in the dependency tree, with no internal structure. Annotators were allowed to take the same step with noun phrases.
- Allowing multiple roots, since a single tweet might contain more than one sentence.

These conventions were justified on the grounds of making the annotation easier for non-experts, but they must be revisited in our effort to apply UD to tweets.

2.2 Part-of-Speech Annotation

Before turning to UD annotations, we (re)annotated the data with part-of-speech, for consistency with other UD efforts, which adopt the universal POS tagset of Petrov et al. (2012).

In some cases, conflicts arose between the UD English treebank conventions (UD_English) [1] [add cite —NAS] and the conventions of Gimpel et al. (2011) and Owoputi et al. (2013). In these cases, we always conformed to UD, enabling consistency (e.g., when we exploit the existing UD English treebank in our parser for tweets, §3). For example, the nominal URL in Figure 2 is tagged as *other* (X) and + is tagged as *symbol* (SYM) rather than *conjunction* (CCONJ).

Tokens that do not have a syntactic function (discussed at greater length in the next section) were usually annotated as *other* (X), except for emoticons, which are tagged as *symbol* (SYM), following UD_English.

Tokens that abbreviate multiple words, such as *idc* ("I don't care") are resolved to the POS of the syntactic head of the expression, following UD conventions (in this example, the head *care* is a verb, so *idc* is tagged as a verb). When the token is not phrasal, we use the POS of the left-most sub-phrase. For example, *mfw* ("my face when") is tagged as a noun (for *face*).

Compared to coarse-grained part-of-speech tagging effort of Gimpel et al. (2011), our approach simplifies some matters. For example, if a token is not considered syntactic by UD conventions, it gets an *other* (X) tag (Gimpel et al. had more extensive conventions). Other phenomena, like abbreviations, are more complicated for us, as discussed above; Gimpel et al. used a single part of speech for such expressions.

Another important difference is in tokenization. UD calls for more aggressive tokenization than that used by Gimpel et al. (2011), which followed **?**. In particular, they opted out of tokenizing contractions and possessives, introducing new parts of speech instead.[2] For us, these tokens must be split, but universal parts of speech can be applied.

2.3 Universal Dependencies Applied to Tweets

We adopt UD version 2 (de Marneffe et al., 2014) guidelines to annotate the syntax of tweets. In applying UD annotation conventions to tweets, the choices of Kong et al. (2014) must be revisited. We consider the key questions that arose in our annotation effort, and how we resolved them.

[for each thing we talk about here, might be nice to quantify it in our annotated corpus —NAS]

Acronym abbreviations. How should we syntactically analyze acronym tokens like *mfw* (abbreviating "my face when") and *rn* ("right now")? Should they be decomposed into their component words, and if so should those words be "normalized" into explicitly spelled out intermediate forms? [explain what we do, and include how this is similar to or different from Kong et al —NAS]

[maybe we owe citations to (Finin et al., 2010; Eisenstein, 2013) here? maybe better papers to cite —NAS]

Non-syntactic tokens. Figure 1 illustrates examples of tokens that have no syntactic function. Note that these generally have the *other* (X) part of speech.

[1] https://github.com/UniversalDependencies/UD_English

[2] These tags only accounted for 2.7% of tokens, leading to concerns about data sparseness in tagging and all downstream analyses.

It is important to note that these types may, in some contexts, have syntactic functions. For example, besides being a discourse marker, *RT* can abbreviate the verb `retweet`, and emoticons and hashtags may be used as content words within a sentence; see Figure 2. Therefore, the criteria for annotating a token as non-syntactic must be context-dependent. To our knowledge, this issue has not previously been addressed in UD annotation efforts [check! —NAS].

[I'm confused about (1) how we define non-syntactic tokens (what are the criteria) and (2) whether this is something we kick over to POS annotation —NAS]

Inspired by the way UD deals with *punctuation* (which is canonically non-syntactic), we adopt the following conventions:

- If a non-syntactic token is within a sentence that has a clear predicate, it will be attached to this predicate;
- If the whole sentence is made of a sequence of non-syntactic tokens, we attach all these tokens to the first one;
- non-syntactic tokens are mostly labeled as *discourse*, but URLs are always labeled as *list*, following the UD_English dataset.

[explain our conventions and how they relate to Kong and to UD. in the draft before, we said that we had conventions, and Kong didn't, but we didn't really say what ours were. —NAS]

[in latex source I've commented out a paragraph about truncated content. it's not clear to me what the nonsyntactic tokens are, when that happens —NAS]

Kong et al. (2014) proposed an additional preprocessing step, *token selection*, to remove non-syntactic tokens before parsing. [confusing. were they removed or just labeled as non-syntactic and then excluded from the tree? I think they were not actually removed, and were still available, e.g., for defining features of the input for parsing —NAS] In order to keep our annotation conventions in line with UD norms, we include non-syntactic tokens in the dependency analysis, with a special label [say what it is, and what they attach to, and clarify that the annotators didn't do this manually (I think) —NAS]

Retweet construction. Figure 1 shows an example of the retweet construction (*RT @coldplay :*). This might be treated as a verb phrase, with *RT* as a verb and the at-mention as its object. This solution would lead to an uninformative root word and, since this expression is idiomatic to Twitter, might create unnecessary confusion for downstream applications aiming to identify the main predicate(s) of a tweet. We therefore treat the whole expression as non-syntactic, including assigning the *other* (`X`) part of speech to both *RT* and *@coldplay*, attaching the at-mention to *RT* with the *discourse* label and the colon to *RT* with the *punct*(uation) label, and attaching *RT* to the predicate of the following sentence.

Constructions handled by UD. A number of constructions that are especially common in tweets are well handled by UD conventions: ellipsis, irregular word orders, and paratactic sentences not explicitly delineated by punctuation.

Vocative at-mentions. Another idiomatic construction on Twitter is an at-mention (followed by a colon [? —NAS]) as a sign that a tweet is a reply to a tweet by the mentioned user. We treat these at-mentions as vocative expressions, labeling them as *proper noun* (`PROPN`) and attaching them to the following predicate with the label *vocative*. [need an example. —NAS]

[I didn't touch anything below here —NAS]

2.4 TWEEBANK V2

Following the guideline mentioned above, we create a new annotated Twitter treebank, which we call TWEEBANK V2.

2.4.1 Data Collection

TWEEBANK V2 is built on the original data of TWEEBANK V1 (Kong et al., 2014, 639 tweets), along with additional 201 tweets sampled from Gimpel et al. (2011) and 2,500 tweets sampled from the Twitter stream from Feb. 2017 to Jul. 2017. The latter data source consists 147.4M English tweets which are filtered by *lang* attribute in the tweet JSON and *langid.py*[3] toolkit.

2.4.2 Annotation Process

The whole annotation process contains their stages. In the first stage, 18 researchers worked on the TWEEBANK V1 proportion and created the initial annotation in one day. This annotation was further refined by the authors of this paper to solve inconsistency between different annotators. In the

[3] https://github.com/saffsd/langid.py

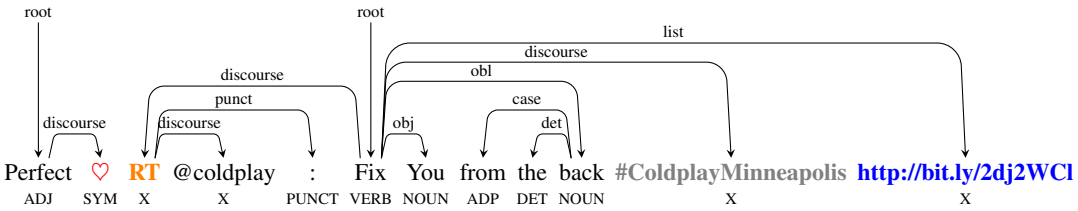

Figure 1: A example tweet contains major non-syntactic tokens, with sentiment emoticon, retweet marker, topical hashtag, and referential URL.

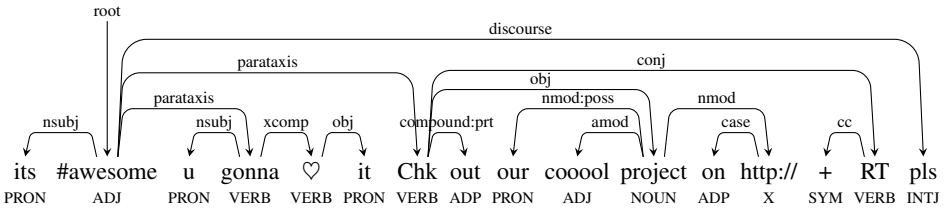

Figure 2: An example tweet with informal but syntactic tokens.

second stage, a postagger and parser are trained on the annotated data from the first stage, and automatically analyze the sampled 2,500 tweets and the authors of this paper manually correct these automatically annotated. Finally, an extra layer of word-level normalization was manually annotated.

We report the inter-annotator agreement between the annotators in the second stage. The agreement on POS is 95.16%, the unlabeled dependency agreement is 88.43% and the labeled dependency agreement is 83.89%. Further analysis shows the major disagreement on POS happens.

2.4.3 Statistics

[YL THIS PART IS NOT FINISHED] Tweebank V2 contains 3550 tweets in total, split into training set with 1639 tweets, development set with 710 tweets and test set with 1201 tweets. The training set contains all the 639 unique tweets in the training set of Tweebank V1 and another 1000 tweets in feb_jul_16 corpus. The development set contains 210 tweets of Tweebank V1 Dev and another 500 tweets in feb_jul_16 corpus. The test set contains all the 201 tweets in the test set of Tweebank V1 and another 1000 tweets in feb_jul_16 corpus.

We process all the raw tweets of these corpora from scratch, from tokenization and POS tagging to dependency annotation. As Tweebank V1 and the POS-tagged Twitter corpus of Gimpel et al. (2011) used a tweet tokenizer (O'Connor et al., 2010) that do not conform with UD guidelines, as we discussed before, we did not use any data of the intermediate steps from these corpora. The basic statistics of Tweebank V2 is shown in Table 1.

	Train	Dev	Test
tweets	1,639	709	1,201
tokens	24,753	11,742	19,112
types	7,579	4,165	6,228
parts	3,025	1,402	2,252

Table 1: Basic Statistics of Tweebank V2

2.4.4 Twitter Specific Statistics

Following the discussion of the Twitter specific linguistic phenomena, we obtain Table 2.

3 Pipeline

We present a pipeline model to parse tweets into universal dependency.

3.1 Tokenizer

Tokenization is non-trivial for Twitter domain (O'Connor et al., 2010). Due to the informal nature, tweets present cases which are unconventional for traditional toeknization module, like hashtags, at-mentions and emoticons. Some of them are even ambiguous without understanding their contexts. For example, the asterisk in *4*3* works as a symbol for times and indicates to segment this token into *4 * 3* but that in *SH*T* works as a mask and *SH*T* should not be segmented.

Instead of manually crafting rules, we propose to deal with tweets tokenization with a character-

				Train	Test	Dev	All	Per
token	syntactic	phrasal abbreviation		12	12	10	34	0.06%
	optionally syntactic	RT	syntactic	40	19	18	77	0.14%
			non-syntactic	600	500	286	1386	2.49%
		hashtag	syntactic	251	212	103	566	1.02 %
			non-syntactic	290	246	155	691	1.24%
		at-mentioned	syntactic	659	420	257	1336	2.40%
			non-syntactic	640	519	311	1470	2.64%
		emoticon/emoji	syntactic	59	31	24	114	2.05%
			non-syntactic	233	164	98	495	0.89%
	non-syntactic	truncated		111	110	50	271	0.49%
structure	retweet			596	498	282	1376	2.47%
	at-mentioned			485	305	175	965	1.74%

Table 2: Twitter Specific Statistics of Tweebank V2

System	F
Stanford Tokenizer	96.6
Twokenizer	94.4
UD pipe v1.2	94.7
UD pipe v1.2 (re-trained)	97.3
Our Tokenizer	**98.3**

Table 3: The tokenization result.

System	P
Stanford Postagger	65.6
UD pipe v1.2	78.7
UD pipe v1.2 (re-trained)	83.8
Our bi-LSTM	91.8
Owoputi et al. (2013)	**95.1**

Table 4: The POS tagging result on gold tokenization.

level bidirectional LSTM (bi-LSTM) sequence-labeling model. Our model takes the raw sentence and tag each character in this sentence as whether it is the beginning of a word.

3.2 POS-Tagger

We use the postagger of Owoputi et al. (2013). In our preliminary experiments, neural model didn't show superiority over the rich-feature linear model.

3.3 Parser

Time is a major concern on

3.4 Exploiting UD_English

4 Experiments

4.1 Tokenization Result

The tokenization results are shown in Table 3.

4.2 POS tagging Result

The POS-tagging results are shown in Table 4.

We also study the influence of tokenization on POS-tagging by evaluating the model's performance on automatically tokenized data. This result is shown in Table 5.

System	F
Stanford Tokenizer	92.0
Our Tokenizer	93.6

Table 5: The POS tagging result on automatic tokenization. The POS-tagger we used is from Owoput et al., 2013.

System	UAS	LAS
Our Parser		

Table 6: The POS tagging result.

4.3 Parsing Result

5 Related Work

[YL RELATED WORD IS NOT FINISHED.] Eisenstein (2013) reviewed NLP approaches for analyzing text on social media, especially for tweets and showed that there are two major directions for NLP community to handle the tweets, including normalization and domain adaptation. He also pointed out that normalization can be problematic because precisely defining the normalization task is difficult.

Kong et al. (2014) argues that the Penn Treebank approach to annotation is poorly suited to more informal genres of text, as some of the annotation challenges for tweets, including token selection, multiword expressions, multiple roots, and structure within noun phrases diverge significantly from conventional approaches. They believe that rapid, small scale annotation efforts performed by imperfectly-trained annotators should provide enough evidence to train an effective parser, given the rapidly changing nature of tweets (Eisenstein, 2013), the attested difficulties of domain adaptation for parsing (**?**), and the expense of creating

Penn Treebank-style annotations (**?**). Therefore, they build a new corpus of tweets (Tweebank), with conventions informed by the domain, using new syntactic annotations that can tackle all the forementioned problems annotated in a day by two dozen annotators, most of whom had only cursory training in the annotation scheme. Then, they modify the decoder of the TurboParser, a graph-based dependency parser, which is open-source and has been found to perform well on a range of parsing problems in different languages (**?**) to adapt to the Tweebank dataset, and incorporate new features such as Brown Clusters and Penn Treebank features and changes to specification in the output space into TurboParser.

Like "mfw" is usually followed by an adverbial clause and "ima" is usually followed by a clausal complement. It is not reasonable to treat them in the same part-of-speech. In this paper, when annotating the POS tagging for abbreviations, we first try to recover their original forms, then use the POS of the core-word as the POS for the abbreviation. Second, four special POS tags (S, L, M, Y) were designed to handle contraction words in Gimpel et al. (**?**). Major concern of designing such tags is to minimize the effort of tokenization. However, contractions of common nouns and pronouns are casted into the same category which increase the difficulty of distinguishing their syntactic function (say, there's and book'll are treated with the same syntactic function). What's more, only a small proportion of words can be categorized into these tags (2.7 % in total), which cast a doubt of the usefulness of these certain tags. In this paper, we believe such contraction can be properly handled by tokenization module, so we suggest to tokenize the contraction word and annotate POS tag accordingly. Besides the contraction that be conventionally tokenized, tweets also witness a set of unconventional contraction like iv (I've), whatis (what is). In this paper, we follow the same idea of annotation abbreviation to handle the unconventional contractions and use the POS of core word of the original form as their POS. Third, special POS was designed to handle emoticon in Gimpel et al. (**?**). However, in most cases, emoticon plays the same role as most of the symbolic tokens. In this paper, we follow the UD guideline to annotate the emoticon as symbol (SYM). At last, its arguable that some of the hashtags, URLs can work as a nominal in tweets. Whether treating them as the same part-of-speech or different ones according to their context is an open question. A preliminary survey on the standard UD English data shows that URL, email address are all tagged as the foreign language (X), so we also tag them as X and leave the disambiguation of their syntactic function to the annotation of parse tree.

6 Conclusion

References

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal stanford dependencies: A cross-linguistic typology. In *LREC*.

Jacob Eisenstein. 2013. What to do about bad language on the internet. In *Proc. of NAACL*. http://www.aclweb.org/anthology/N13-1037.

Tim Finin, William Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in twitter data with crowdsourcing. In *Proc. of NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proc. of ACL*. ACL.

Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A. Smith. 2014. A dependency parser for tweets. In *Proc. of EMNLP*. ACL.

Brendan O'Connor, Michel Krieger, and David Ahn. 2010. Tweetmotif: Exploratory search and topic summarization for twitter.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proc. of NAACL*. ACL.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proc. of LREC*. ELRA.