Parsing Tweets into Universal Dependencies

First Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Second Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

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.

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

2 Data

2.1 Linguistic Phenomena of Twitter

Twitter, as an extreme example of informal domain, contains a collection of conversational languages (like abbreviation, informal contraction, and variant entity names) and twitter-specified markers (like retweet mark, and username). Despite the informal nature of tweets, we argue that they basically follow the same grammar as their formal language counterpart. The key issue lies in understanding the specific construction that doesn't carry any syntactic functions, and designed rules to handle the specific constructions in tweets. In the following section, we studied such constructions from both the token and structure level. [$^{\text{Y}}_{\text{L}}$ I DON'T LIKE THE NAME OF STRUCTURE LEVEL]

2.1.1 Token Level

Informal Syntactic Tokens. Informal tokens like phrase abbreviation (like `mfw`: my face when, `rn`: right now, and etc.) and orthographic variants (like `sooo cooool`) have drawn much attention in previous literals (Finin et al., 2010; Eisenstein, 2013) [$^{\text{Y}}_{\text{L}}$ NOT SURE ORTHOGRAPHIC VARIANTS IS A GOOD NAME, NEED REFERENCE.] However, in the sense of syntactic functions, they work in the same way as the ordinary words, which makes them requires less specialty in designing annotation conventions.

Non-syntactic Tokens. Besides the informal but syntactic tokens, there are a large collection of tokens that cannot be handled by the syntactic theory designed for standard text. A major part of these tokens consists of meta data in tweets, like the sentiment emotion, retweet mark, topical hashtag, and referential URL shown in Figure 1.

However, whether a token convey some syntactic function cannot be simply decided by its form. For example, besides being a discourse marker, `RT` can also works as a abbreviation of the verb `retweet`. Emoticons are sometimes used as a verbal part of the sentence. Twitter users have developed a casual habit of using hashtag, and even use hashtag as a adjective. The sentence in Figure 2 gives a comprehensive example of those informal but syntactic tokens.

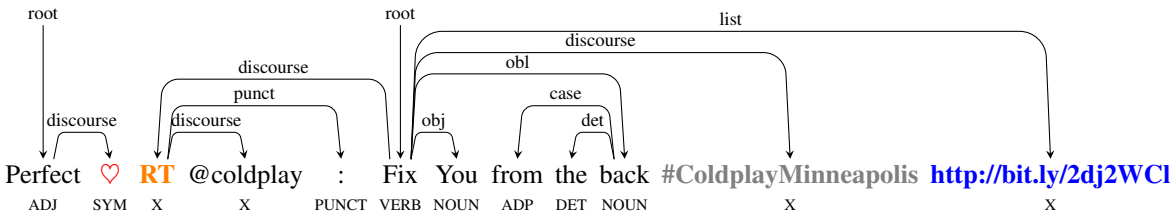

Figure 1: A example tweet contains major non-syntactic tokens, with sentiment emoticon, retweet marker, topical hashtag, and referential URL.

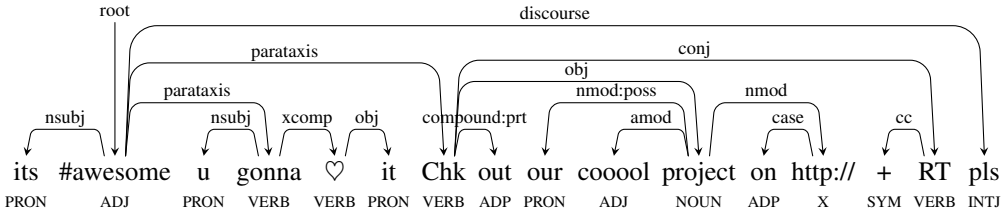

Figure 2: An example tweet with informal but syntactic tokens.

It shows whether a token makes a non-syntactic one is highly conditioned on its context.

A kind of token that doesn't carry clearly syntactic functions can be resulted by the 140 characters limits of tweets. Since last tokens that exceeding 140 characters limits are truncated in tweets which leaves a broken sentence, it is impossible to fully recover their original form and figure out their syntactics. In this paper, we propose to treat these truncated tokens, along with the other 4 kinds of tokens mentioned above, as ones without syntactic functions.

In the sense of dealing with the non-syntactic tokens, Kong et al. (2014) proposed an preprocessing step named *token selection* to remove non-syntactic tokens from the original tweets. However, concrete standard of non-syntactic tokens was not presented in their work. What's more, these unselected tokens are absent from downstream processing, including parsing. We differ from their work by proposing concrete standard of non-syntactic tokens and we include these non-syntactic tokens in our final analyzed dependencies by adding special attachment to them.

2.1.2 Structure Level

The syntactic constructions vary greatly across different tweets. Unconventional constructions which frequently occur include ellipsis, irregular word orders, and para-tactic sentences without the segmentation of punctuations [$^{\text{Y}}_{\text{L}}$I MOVE THE 3RD BULLET POINT HERE BECAUSE THIS CONSTRUCTION WAS HANDLED BY UD-GUIDELINES]. And these unconventional ones are more or less resolved by the universal dependencies. The unresolved cases are mainly resulted by the format of twitter. The *retweet* and *at-mention* are of two major constructions.

Retweet. For the *retweet* construction ('`RT @coldplay :`' in Figure 1), although it can be treated as a verbal phrase in which retweet marker works as a verb and the following at-mention is its objective, we argue that such convention lead to a uninformative root word. It also increase the difficulty for downstream tasks in figuring out the core predicate and arguments for one certain tweet. Thus, we treat the whole construction as a non-syntactic phrase. [$^{\text{Y}}_{\text{L}}$CORRECT ME IF I AM WRONG.]

At-mention. At-mention in the beginning of a tweet is the format of indicating the 'reply' action. In this paper, we treat the at-mentions as vocative expressions for their connections with the traditional vocative words, rather than treat them as non-syntactic metadata, which increase the complexity for analyzing a tweet. [$^{\text{Y}}_{\text{L}}$SINCE THE ORIGINAL

DISCUSSION MAINLY FOCUS ON TOKENS, I MOVE THEM TO THE FORMER SECTION.]

2.2 Part-of-Speech Annotation

We adopt the universal Part-of-Speech (POS) tagset (Petrov et al., 2012) to annotate the tweets. The POS annotation of a word in tweets is generally based on its syntactic distribution, which indicates that the POS of word changes on different context. Sticking to the syntactic distribution makes it possible that a '♡' symbol is tagged as a verb as shown in Figure 1. [$^{Y}_{L}$NEED A BETTER EXPLANATION.]

We need to note that in some certain cases, the distribution criteria doesn't holds. For example, the nominal URL in Figure 2 is tagged as *other* (X) and the conjunctor + symbol is tagged as *symbol* (SYM) rather than *conjunction* (CCONJ). Such violation is resulted by our attempt of conforming the annotation to the *universal dependencies - English dependency treebank* (UD_English), in which all the URLs are tagged as *other* and symbolic conjunctors are tagged as *symbol*.[1] By conforming our annotation to the UD_English data, hopefully, the tools we built on tweets can benefit from the dataset on broader domain. [$^{Y}_{L}$NEED WORDING]

Again, there are un-conventional linguistic constructions in tweets that syntactic distribution doesn't handle well. One of the cases is the non-syntactic tokens. We consider most of them as X, except for the sentiment emoticons, which would be tagged as SYM, in practice of matching the UD_English. Another case is the retweet construction which is considered as a non-syntactic phrase and both the RT and at-mention are tagged as X. For the at-mention, we tag them as proper noun (PROPN) in spirit of following the vocative example in UD_English.

The informal syntactic tokens can generally be handled by their distribution. However, one of the exception is the phrasal abbreviations in which one token carries multiple syntactic functions. In dealing with such phrases, we propose a routine that first recover their original forms, then, use the POS of their syntactic head word as the POS of the whole phrase. Therefore, *idk* (I don't care) is a verb because *care* is its syntactic head. Such routine works for most of the phrasal abbreviations. When it falls, we just use the POS of the left-most subphrase. [2]

Similar to our work, Gimpel et al. (2011) proposed a course grain twitter POS tagset that handles most of the token level phenomena. However, our work diverge theirs in several ways. Compared with the Gimpel's work, which assigns different tags to non-syntactic tokens in different categories, we simplify them into the X tag. At the same time, all the abbreviations in Gimpel et al. (2011) are tagged with one POS, which doesn't distinguish their syntactic functions. We argue that such approach oversimplify the abbreviations. One example is that *idc* (I don't care) and *rn* (right now) carries significantly different syntactic functions and should not be treated as the same. In this paper, we stick more to the syntactic distribution when handling such abbreviations and it allows them to function differently in different context.

The last difference between Gimpel et al. (2011) is resulted by the tokenization process. In Gimpel et al. (2011), they opted out the tokenization on contractions and possessives and introduce several tags for these un-tokenized words. As mentioned above, the goal of conforming our annotation to the UD_English makes us adopt the same the UD-styled tokenization. Thus, these tags for contractions and possessives are not necessary at all in our case. What's more, as Gimpel et al. (2011) mentioned, only a small proportion of tokens can be categorized into these tags (2.7 % in total), which casts a doubt of the usefulness of these tags.

2.3 Dependency Annotation

We follow the UD V2 syntax guideline and the English specific Universal Dependency Relations to annotate our data.

In token level phenomena, we adopt the following rules for non-syntactic tokens, denoted as x:

- If the sentence x belongs to has a clear predicate with syntactic function, x should not have

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

[2] Such failure can be resulted by the mis-match of phrase boundaries between the original form and the definition of 'phrase' in UD. For example, 'mfw' (my face when) doesn't make a phrase because 'when' belongs to the subordination that modifies 'my face'.

any dependent and is attached to this sentences predicate;

- If the sentence x belongs to consists of only non-syntactic tokens we attach x to the first token of the sentence;
- In parataxis case, when it is difficult to decide which sentence or phrase x should belong to, we always attach x to the previous predicate of the paratactic clause or phrase;
- x is mostly labelled as *discourse*, but URLs are always labelled as *list*;

The reason we annotate URLs differently is the same as in POS tagging, that UD_English labelled them as *list* when they are non-syntactic tokens and we want to keep the dataset conformed with it.

For syntactic tokens, they are considered the same as normal tokens and we just annotate them by following the UD guidelines.

we further adopt the following rules for structure level phenomena:

- Retweet structure: we attach `@user` to `RT` and `RT` to the predicate of the following sentence with *discourse* as labels.
- At-mentioned structure: we attach at-mentioned username to the predicate of the most relevant sentence with *vocative* label. If it is hard to decide which sentence would be the most relevant, we attach it to the previous sentence's predicate.
- Paratactic parts without punctuations: we follow the UD convention, set the first part as the main part of the sentence, and set its predicate as the main predicate of the sentence. Every predicate of the following parts should attach to the main predicate. If the main part and some following part are both phrases, the label is *list*, otherwise the label is *parataxis*. Note that we still have an exception for URLs, conforming with UD, which are always labelled as *list*, regardless of the type of the other part.

Kong et al. (2014) built a syntactic treebank of tweets called Tweebank using Gimpel et al.'s POS tagset. Apart from the difference in POS tagging, we found the following difference in both methods. First, Tweebank is an unlabelled dependency treebank. There is clearly a doubt about the usefulness of the Tweebank. We annotate the labelled dependencies by following the UD annotation guideline. Second, most of Tweebank was built in a day by two dozen annotators, most of whom had only cursory training in the annotation scheme (Kong et al., 2014). We think, after investigation, that despite the speed, the quality of data is still to be improved. We follow strictly the UD annotation guideline and only make twitter specific conventions when there is no suitable convention to adopt in UD guideline or no relevant data found in UD_English. Third, Kong et al. (2014) developed a first-order sequence model to filter out the non-syntactic tokens and punctuations, and exclude them before parsing. It is very different from most of the annotation conventions, especially for UD conventions. We believe that every token should be included in the dependency tree, whether they have syntactic functions or are to be evaluated in the end, as in most of the conventions, so we annotate all tokens. Last, Kong et al. (2014) adopt the "Yamada-Matsumoto" conventions (**?**), where auxiliary verbs are parents of main verbs, and prepositions are parents of their arguments, in contrast to UD conventions, where content words are put into the primacy, and dependency relations hold primarily between them, rather than being indirect relations mediated by function words. Auxiliary verbs and prepositions are all function words, and are attached to the most related content words in UD.

2.4 Tweebank V2

We present Tweebank V2, a version 2 of Tweebank (Kong et al., 2014).

The tweet source of Tweebank V2 consists of three parts. The first is the original Tweebank created by (Kong et al., 2014), denoted as Tweebank V1. There are 639 unique tweets in the training data of Tweebank V1, and 201 unique tweets in the data set. All of the unique tweets in Tweebank V1 are included in Tweebank V2. The second are 210 tweets from POS-tagged Twitter corpus of **?**), denoted as Tweebank V1 Dev. It was initially created as a development set of Tweebank V1 for hyperparameter tuning. Tweebank V1 Dev comprises were equally

sampled from two data set, OCT27 (tweets sampled from a particular day, October 27, 2010) and DAILY 547 (one random English tweet per day from January 2011 through June 2012, 547 tweets in total). We should note that all of the Tweebank V1 tweets were also drawn from OCT27 and DAILY547, and the new tweets we extracted are not in the Tweebank V1. The third are 2500 tweets from the Twitter streams from February 2017 to July 2017 from ArchiveTeam,[3] which we call Feb_Jul_16 corpus. The Twitter streams are in Spritzer version, which provides a 1% random sample of public tweets everyday, and the tweets are in twitter JSON format. There are xxx tweets in all the streams. We filtered them with *lang* attribute in *user* and used *langid.py*[4] to pick out English tweets, which ends in xxx tweets in English. Then we sampled 2500 tweets roughly equally from these months.

2.4.1 Basic Statistics

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 **?**) used a tweet tokenizer (**?**) and POS tagger (**?**) 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.

2.4.2 Twitter Specific Statistics

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

[3]https://archive.org

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

	Train	Dev	Test
tweets	1639	709	1201
tokens	24753	11742	19112
types	7579	4165	6228
parts	3025	1402	2252

Table 1: Basic Statistics of Tweebank V2

3 Pipeline

4 Model

5 Experiments

6 Related Work

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

				Train	Test	Dev	All	Per
token	syntactic	phrasal abbreviation						
	optionally syntactic	RT	syntactic non-syntactic					
		hashtag	syntactic non-syntactic					
		at-mentioned	syntactic non-syntactic					
		emoticon/emoji	syntactic non-syntactic					
	non-syntactic	truncated						
structure	retweet at-mentioned parataxis without punctuations							

Table 2: Twitter Specific Statistics of Tweebank V2

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.

7 Conclusion

References

Jacob Eisenstein. 2013. What to do about bad language on the internet. In *Proc. of NAACL*, June.

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, October.

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