Parsing Japanese Tweets into Universal Dependencies

Hayate Iso, Kaoru Ito, Hiroyuki Nagai, Taro Okahisa and Eiji Aramaki

Nara Institute of Science and Technology

Summary

- Despite the importance of the Japanese tweets, little attention has been focused on the Japanese tweet processing.
- Developed the compatible guidelines with existing Japanese and English tweet UD corpus and manually annotated 700 tweets from scratch to build Japanese tweet UD corpus, TWEEBANK-JA.
- Built the Japanese tweet processing tool, TWPIPE-JA.
- To achieve the fast and accurate pipeline processing, we adopted the feature caching for a joint model that enables 1.7 times faster processing while preserving the pipeline performance.

Building TWEEBANK-JA corpus

- Our focus is to create the Japanese tweet corpus that is compatible with the existing UD corpus.
- Thus, we have carefully transformed the existing Japanese formal text [1, 2] and English tweet[3] annotation guidelines into Japanese tweets.

Word Boundary

- For the word unit, we generally adopt the short unit word (SUW), a minimal language unit that has a morphological function as a word unit [4] except for the Twitter-specific word definition about *Emojis*, *Kaomojis* (eastern emoticons), and hashtags.

- **Emoji & Kaomoji**
  - Regarding the SUW, we generally split multiple *Emoji* sequences into single *Emojis* (e.g., 😂😂😂to 😂/😂/😂).
  - However, sometimes multiple *Emojis* are combined together to represent a single meaning (e.g., 🏃💨means "run in haste") or in other cases *Emojis* are used as parts of the *Kaomoji* (e.g., 🍵(_˘-˘_🍵_)ｵﾁｬﾝ means "a person serving green tea").
  - In such cases, we treat these sequences as a single SUW.
- **Hash-Tag**
  - Throughout our annotations, we found that a single hashtag often contains multiple SUWs in Japanese language. For example, the hashtag "#戒めの投稿" contains four SUWs, "#/戒め/の/投稿."
  - However, we observed only one example that used the hashtag that contains the multiple SUWs used syntactically. This decomposition would lead the parsing errors. Therefore, we treated each hashtag as the SUW.

Part-of-Speech Tag

- Regarding the part-of-speech annotations for the tweet-specific tokens, we referred to [3] (e.g., PROPN for at-mention, X for RT, URL, and Hashtag, SYM for emoticon).
- When they are used syntactically, we annotated the corresponding part-of-speech tag.

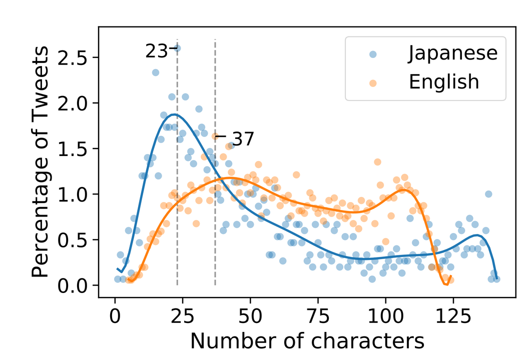
Dependency Structure

- We followed the tweet-specific dependency structure defined by [3, 5] (e.g., multiroot for single tweet, RT construction, vocative at-mention).
- For URLs and SYM, however, we annotated the dep label because of UD_JAPANESE-BCCWJ.

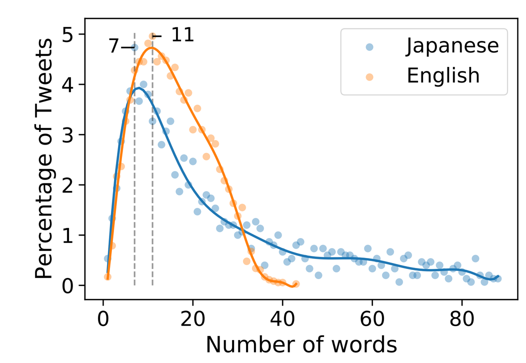
Dataset and Its Statistics

	TRAIN	DEV	TEST	TOTAL
tweets	500	100	100	700
words	13,190	2,471	2,668	18,329
characters	31,226	6,135	6,184	43,545

Table: Statistics of TWEEBANK-JA. The number of words is comparable scale with [5].



(a) Dist. of # characters



(b) Dist. of # words

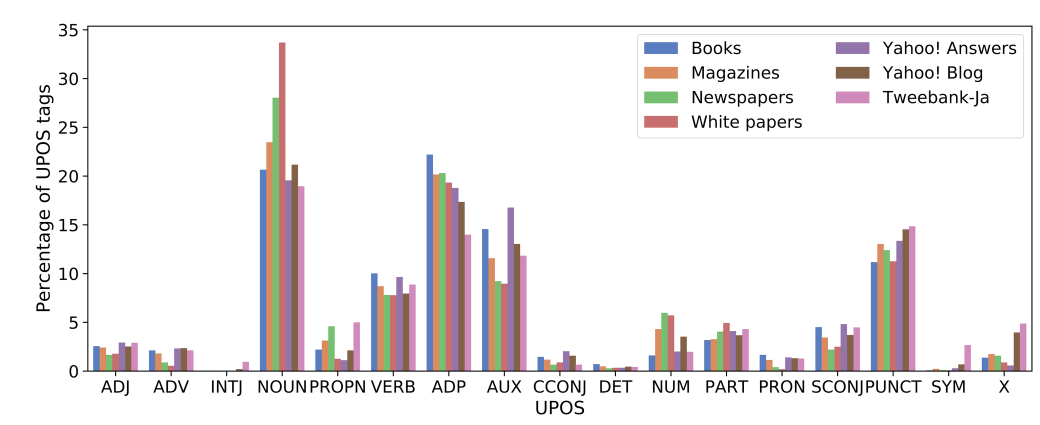


Figure: Dist. of UPOS tags in Japanese UD corpus. The # of INTJ, PROPN, SYM, X is larger than other corpus and ADP is smaller.

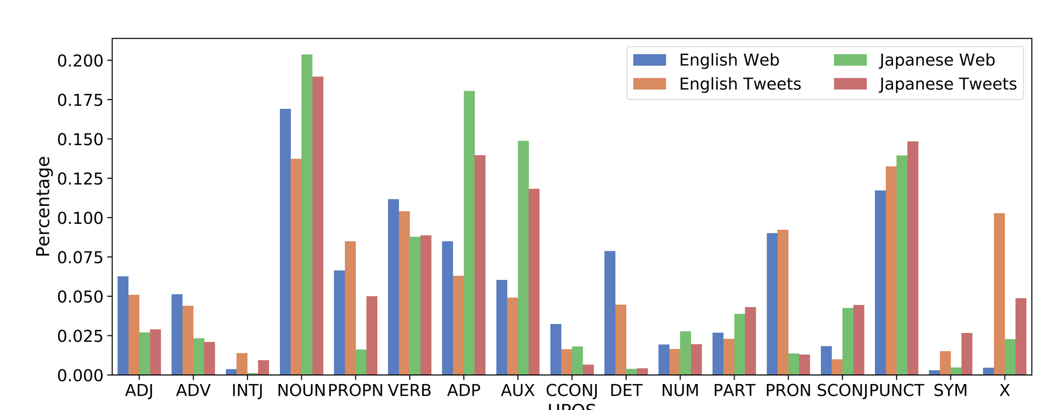


Figure: Dist. of UPOS tags in UD web corpus.

Building Blocks for TWPIPE-JA

Word Segmentations

- We chose the character-level sequential tagging with conditional random fields [CRF; 6] for the word segmentation.
- We used the character sequence, the character-type feature, and the character-level treebank embedding [7], which models the treebank-wise differences for the inputs.

Table: Segmentor comparison on the TWEEBANK-JA test set.

System	F_1
UDPipe	84.7
KyTea	90.2
our BiLSTM-CRF segmentor	91.6
- Type embdding	89.3
- Treebank embedding	91.4

Part-of-Speech Tagging

- We chose word-level sequential tagging with CRF for Part-of-Speech Tagging.
- We used character encoding by BiLSTM and word-level treebank embedding [7].

Table: POS tagger comparison on gold-standard words in the TWEEBANK-JA test set.

System	Accuracy
our BiLSTM-CRF	87.7
- Character encoding	85.5
- Treebank embedding	87.6

Dependency Parsing

- For a fast and accurate dependency parsing, we used the greedy transition-based system [8, 9], where the complexity is linear in the length of sentence n, $O(n)$.
- Specifically, we adopted the BiLSTM parser [10, 11].

Table: Dependency Parser comparison on gold-standard words and pos-tags in the TWEEBANK-JA test set except for last line. The last result is on gold-standard words and predicted-pos-tags

System	UAS	LAS
our BiLSTM-Parser	87.8	78.8
- PoS embedding	76.8	65.0
- Character encoding	87.2	78.1
- Treebank embedding	86.4	76.2
+ Predicted PoS tags	80.8	68.4

Multitask Pipeline

- Although we empirically found the importance of character-level information for the word-level tasks, both PoS tagging and dependency parsing, these process could be bottlenecked.
- To obtain a faster pipeline model, we stacked all of the pipeline models with a task hierarchy [12].
- During decoding, this model could cache the previously extracted feature values and the cached features are used for the next task inputs. Subsequently, the deep joint could resolve the feature extraction bottleneck with this simple technique that we call *feature caching*.

Pipeline	Score	Joint	Disjoint
Word Segmentation	F_1	91.8	91.6
PoS tagging	F_1	81.4	81.6
Dependency Parsing	*LAS* F_1	58.8	58.6
Tweets into UD	Kw/s	1.9	1.1
Total Model size	#Param	4.5M	9.2M

Table: Pipeline evaluation and speed comparison between the Joint and Disjoint pipeline model. The "Kw/s" indicates the parsing speed evaluated by thousands of words the model processed per second. The "#Param" indicates the number of parameters for all of pipeline models.

References

[1] Masayuki Asahara, Hiroshi Kanayama, Takaaki Tanaka, Yusuke Miyao, Sumire Uematsu, Shinsuke Mori, Yuji Matsumoto, Mai Omura, and Yugo Murawaki. Universal Dependencies Version 2 for Japanese. In *LREC*, 2018.

[2] Takaaki Tanaka, Yusuke Miyao, Masayuki Asahara, Sumire Uematsu, Hiroshi Kanayama, Shinsuke Mori, and Yuji Matsumoto. Universal Dependencies for Japanese. In *LREC*, 2016.

[3] Yijia Liu, Yi Zhu, Wanxiang Che, Bing Qin, Nathan Schneider, and Noah A. Smith. Parsing Tweets into Universal Dependencies. In *NAACL-HLT*, 2018.

[4] Kikuo Maekawa, Makoto Yamazaki, Toshinobu Ogiso, Takehiko Maruyama, Hideki Ogura, Wakako Kashino, Hanae Koiso, Masaya Yamaguchi, Makiro Tanaka, and Yasuharu Den. Balanced corpus of contemporary written Japanese. In *LREC*, 2014.

[5] Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A Smith. A Dependency Parser for Tweets. In *EMNLP*, 2014.

[6] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[7] Sara Stymne, Miryam de Lhoneux, Aaron Smith, and Joakim Nivre. Parser Training with Heterogeneous Treebanks . In *ACL*, 2018.

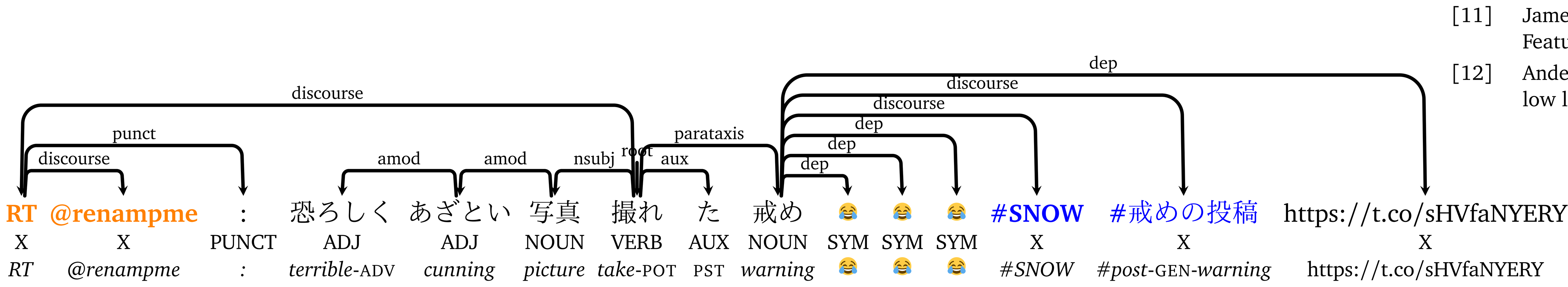
[8] Hiroyasu Yamada and Yuji Matsumoto. Statistical dependency analysis with support vector machines. In *IWPT*, 2003.

[9] Joakim Nivre. An efficient algorithm for projective dependency parsing. In *IWPT*, 2003.

[10] Eliyahu Kiperwasser and Yoav Goldberg. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*, 2016.

[11] James Cross and Liang Huang. Incremental Parsing with Minimal Features Using Bi-Directional LSTM. In *ACL*, 2016.

[12] Anders Søgaard and Yoav Goldberg. Deep multi-task learning with low level tasks supervised at lower layers. In *ACL*, 2016.



"RT @renampme: I have taken a terribly cunnning picture warning 😂😂😂#SNOW #post to warn https://t.co/sHVfaNYERY"