Token Boundaries or Named Entity Boundaries

Han-Cheol Cho, Naoaki Okazaki, Jun'ichi Tsujii
Tsujii lab., Department of Information Science and Technology, the University of Tokyo
{hccho, okazaki, tsujii}is.s.u-tokyo.ac.jp

1 Introduction

Named Entity Recognition (NER) is a task that recognizes mentions of specific entities of interest in text. These mentions could be names of people, organizations and locations[6], or terminologies in specialized domains (e.g. medical areas) such as gene and protein names[2, 5].

NER has been mostly formalized as a sequential labeling task which labels a sequence of tokens with a set of pre-defined tags. This formalization means that input sentence should be tokenized first. While it is a necessary step, tokenization has not drawn much attention from researchers probably because simple tokenization methods usually work well for NER tasks in general domain.

In biomedical domains, however, such tokenization methods may produce a token sequence having inconsistent token boundaries with named entity (NE) boundaries. The example[1] in Figure 1 shows the parts of sentences tokenized on non-alphanumeric characters[2]. GGP (gene or gene product) mentions are bold-faced.

- … determined that **H4**Ac16 is present along …
- … , CSN5 binds to oligo**ubiquitin** chains …
- … rat (r**minK**) or human **minK** ...

Fig. 1: GGP mention examples in text

In the above examples, the tokenization fails to produce proper tokens for NER. In such a situation, a NER system may recognize a whole token. These NEs having incorrect boundaries are problematic since they are mostly used as input to high level NLP applications such as event extraction.

In this paper, we propose a character-based NER system based on both character- and token-level features. By labeling characters, we can avoid inevitable boundary inconsistencies caused by tokenization. However, character-based NER has to make correct predictions on much longer sequences of characters than token-based NER to correctly recognize NEs of the same length. To tighten up the relations of the labels within a same token, we adopt the tokenization method used in the BANNER for feature generation. Token-level features will give uniform feature weights to the characters within in a same token, and therefore can strengthen them to have a more likely label sequence.

2 Proposed Method

The proposed method exploits features in both character-level and token-level. We designed two different feature sets for them. Figure 2 shows the system architecture of the proposed system. In the following sections, we explain each component in detail.

2.1 Tokenization Methods

The proposed NER system uses two tokenization methods. The simplest one is the character splitter which divides an input into a sequence of characters. The other one is adopted from the BANNER. The BANNER's tokenization method simply divide an input into tokens based on non-alphanumeric characters. For example, "2,3,7,8-tetrachlorodibenzo-p-dioxin" will be tokenized into thirteen tokens, "2",

[1] This example is excerpted from the Epigenetics and Post-translational Modifications event corpus of the BioNLP2011 shared task. The training and development data are used for evaluation too.

[2] This method is adopted from a publicly available NER system, BANNER. http://cbioc.eas.asu.edu/banner/

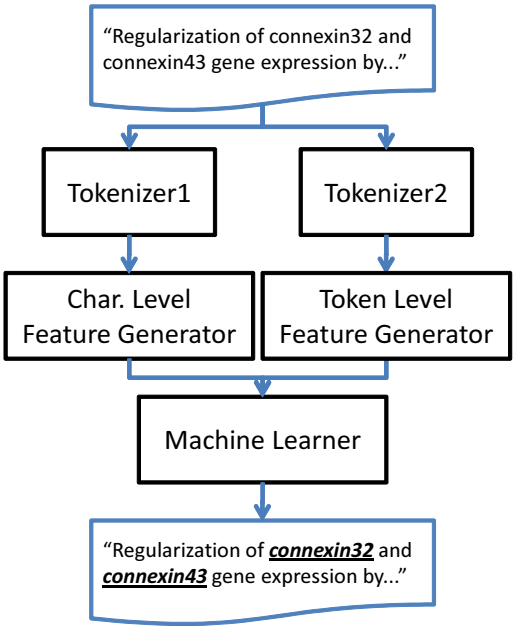

Fig. 2: The system architecture

",", "3", ",", "7", ",", "8", "-", "tetrachlorodibenzo", "-", "p", "-", "dioxin."

2.2 Feature Design

We designed features in two perspectives: character level features and token-level features. We will explain these features with the example sentence, "Regularization of connexin32 and connexin43 gene expression by...," where the current token position (t_{token}) is "connexin32" and the current character position (t_{char}) is the fifth character "e."

Table 1 shows the character level feature templates. The LEFT_SPACE (or RIGHT_SPACE) feature is true when the left (or right) character of the current position is a space. Therefore, the LEFT_SPACE and RIGHT_SPACE are both false in this example. Character N-grams (N is from 1 to 8) are generated within the context window, [t_{char}-7, t_{char}+7][3]. We names character N-gram features as c[begin][end]=[N-gram]. (e.g. c[-1][-1]=n, c[0][0]=e, ..., c[-7][-4]=of_co, ..., c[0][7]=exin32_a). There are also conflated character N-grams where continuous number parts are replaced with a single zero and non-alphanumeric character parts are substituted with the under bar symbol. These feature names begin with "cc" instead of "c". (e.g. cc[0][7]=exin0_a). For dictionary N-grams, we first apply a dictionary tagger which performs exact string matching with two string normalization heuristics[4]. We used two dictionaries compiled from a gene database, EntrezGene[5], and a medical term database, UMLS[6] Then, N-grams (N is 3 to 5) of the matched string are generated. For dictionary features, we abstracted the position information into three types, left(-), current(0) and right(+) positions. (e.g. D_GENE[-][-]=BII, D_GENE[-][-]=III, D_GENE[-][0]=III, ... D_GENE[+][+]=IIIII). Lexicalized dictionary features are also generated with conflated lexical features. (e.g. D_GENE[-][-]=BII/con, D_GENE[-][-]=III/onn, D_GENE[-][0]=III/nne, ... D_GENE[+][+]=IIIII/xin0).

token-level feature templates are shown in Table 2. We use uni-grams and bi-grams for all features within the [t_{token}-2,t_{token}+2] context window except dictionary N-grams which will be generated with the

Class	Description
Space	LEFT_SPACE, RIGHT_SPACE
Char N-grams	N = 1-8, W = 7
Dic. N-grams	N = 3-5, W = the length of a matched string
Dic. length	the length of matched string

Table 1: Char. level features (N: n-gram size, W: context window size)

Class	Description
Token N-grams	N = 1-2, W = 2
Lemma N-grams	N = 1-2, W = 2
POS N-grams	N = 1-2, W = 2
Lemma & POS N-grams	N = 1-2, W = 1-2
Dic. N-grams	N = 1-2, W = the length of a matched string
Dic. length	the length of matched string

Table 2: token-level features

[3]Space is not counted as window size.

[4]One or more numbers are replaced with a single zero and non-alphanumeric characters are substituted with the unber bar symbol

[5]http://www.ncbi.nlm.nih.gov/gene

[6]http://www.nlm.nih.gov/research/umls/

dictionary matched strings.

2.3 Machine Learning

For training our NER system, we used LibLinear[7]. We implemented a search algorithm which uses two previously predicted labels as features. In NER task, greedy search is known to show comparable performance to Viterbi search[4] while it provides great speed up for both training and tagging.

We trained three models: CHR, TOK and CHR+TOK. The CHR model, as a baseline system, is a character-based NER system which uses only character-level features. On the other hands, the TOK model is another baseline system that is a token-based NER system which uses only token-level features. The CHR+TOK model incorporates both character-level and token-level features into a single model, and performs character-based NER.

We used L2-regularized L2-loss support vector classification (dual) solver. Each model is trained with ten regularization parameter C values (0.01, 0.1, 1, 5, 10, 15, 20, 25, 30, 35). The performance comparison in Section 3 is done with the best performance for each model.

3 Evaluation

For evaluation, we use one of the BioNLP 2011 shared task corpora[8], the Epigenetics and Post-translational Modifications corpus. This corpus provides comprehensive GGP (gene or gene product) annotations on the given domain where the nomenclature is a very example of a community specific naming convention. Token-based NER often fails to recognize GGP mentions due to the inconsistent token boundaries. There are 2,499 GGP mentions in the development data[9], and 147 mentions have inconsistent token boundaries even when a fine-grained tokenization method used in the BANNER is applied.

3.1 Performance Evaluation

Table 3 shows the performance of the three models. As explained in Section 2.3, we applied ten different regularization values for each model and used the best scores for the performance comparison.

Model	Recall	Prec.	F1-score
CHR (c=0.1)	70.91%	86.44%	77.91%
TOK (c=0.1)	73.99%	84.51%	78.65%
CHR+TOK (c=1.0)	74.71%	87.32%	80.53%
NERSuite	75.0%	81.66%	78.24%

Table 3: Evaluation results

A baseline model, CHR, achieves a 77.91% F1-score. The other baseline model, TOK, achieves a 78.65% F1-score. Lastly, the CHR+TOK model improves its performance by a 2.62% from the CHR model and a 2.84% from the TOK model.

Next, we applied the in-house version of the NERSuite[10] to investigate how well a token-based NER system work on this corpus. The NERSuite is a CRF-based NER system specially tuned for biomedical NER tasks. (Due to the limited time, we could not test different regularization parameter values for this system. Therefore, the performance of the NERSuite should not be regarded as its best performance on the test data.)

3.2 Error Analysis

The CHR+TOK model shows higher performance than its baseline models. To investigate the reason of this improvement, we checked whether the CHR+TOK model properly recognizes NEs which cannot be detected by the TOK model correctly. The model successfully recognized named entities such as s**TSHR**, **H3**K4, m**OAT1**. (NEs are marked in boldfaced font). However, we found that some names entities are not recognized at all (e.g. **gap1**Delta, m**HP1alpha**) or recognized by token boundaries (e.g. **PrP**sc, h**AChE**).

The result is promising because many NEs are correctly recognized which could not be recognized properly by token-based NER systems. However, the features used in this research are relatively primitive. We think that it is necessary to design new

[7] http://www.csie.ntu.edu.tw/ cjlin/liblinear/

[8] https://sites.google.com/site/bionlpst/

[9] We use the development data for evaluation since the test data is currently not available.

[10] The in-house version of NERSuite uses dictionary features while the release version (http://www-tsujii.is.s.u-tokyo.ac.jp/nersuite/) is not able to use it yet.

features which are informative to decide NE boundaries within tokens.

4 Related Work

The use of prefixes, suffixes and character n-grams for NER task can be found in the previous work[3]. However, they use such features to relieve the unknown word problem in token-based NER. One of NER systems of LingPipe[11] is a character based HMM model. This model labels characters as our system, but it only uses character-level features since incorporating rich features into a generative model is often impossible.

In Japanese NER task, there is a work[1] which uses redundant word segmentation analysis information to handle word segmentation errors. While their approach is similar to ours in methodological viewpoint, it requires wide-coverage morphological analyzer(s). Since NER targets various domains, it could be difficult to prepare such a wide-coverage morphological analyzer.

5 Conclusion

Tokenization can be a tricky problem for NER in specialized domains such as biomedical areas. In this paper, we proposed a character-based NER system which can avoid the inconsistent token boundary problem inherited from tokenization stage. We also adopted a tokenization method to the feature generation step. These features could help the characters belonging to a same token to have a more plausible label sequence.

参考文献

[1] Masayuki Asahara and Yuji Matsumoto. Japanese named entity extraction with redundant morphological analysis. In *Proceedings of the NAACL-2003*, pp. 8–15, 2003.

[2] Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. Introduction to the bio-entity recognition task at jnlpba. In *Proceedings of International Joint Workshop on NLPBA '04*, pp. 70–75, 2004.

[3] Dan Klein, , Dan Klein, Joseph Smarr, and Christopher D. Manning. Named entity recognition with character-level models. In *Proceedings of CoNLL-2003*, 2003.

[4] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the CoNLL-09*, pp. 147–155, 2009.

[5] L Smith et al. Overview of biocreative ii gene mention recognition. *Genome Biology*, Vol. 9, No. Suppl 2, p. S2, 2008.

[6] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: language-independent named entity recognition. In *Proceedings of the 7th Conference on HLT-NAACL*, pp. 142–147, 2003.

[11] Pipehttp://alias-i.com/lingpipe/