mhirano at the FinSBD Task: Pointwise Prediction Based on Multi-layer Perceptron for Sentence Boundary Detection

Masanori HIRANO[1*], **Hiroki SAKAJI**[1], **Kiyoshi IZUMI**[1], **Hiroyasu MATSUSHIMA**[1]
[1]School of Engineering, The University of Tokyo
hirano@g.ecc.u-tokyo.ac.jp, {sakaji, izumi, matsushima}@sys.t.u-tokyo.ac.jp

Abstract

This paper proposes a pointwise prediction for a sentence boundary detection task. The proposed pointwise prediction is combined with our original word embedding method and three-layered perceptron. It predicts whether the targeted words have the role of the beginning/end of a sentence or not by using word features around the targeted words. We tested our model by changing some parameters in our model and then ensembled these models with various parameters. Consequently, the ensembled model achieved 0.88 and 0.84 averaged f1-score by testing the data both in English and French, and it also obtained 0.84 in English and 0.86 in French as the final results of this shared task. In addition, we developed a baseline model, that is, a rule-based prediction model, for comparison. The result shows that the proposed pointwise prediction model outperformed the rule-based prediction model in any index.

1 Introduction

This paper presents the application technique[1] of the pointwise prediction to a shared task of Sentence Boundary Detection in PDF Noisy Text in the Financial Domain for the FinSBD 2019 shared task [Ait Azzi *et al.*, 2019][2].

We address the sentence boundary detection problem in PDF Noisy Text using a type of approach referred to as "pointwise" prediction. Pointwise prediction is an approach used to make every single independent decision at each point by using only the features around a single point. In this task, a pointwise prediction indicates predicting whether each word has a role as the beginning/end of the sentence or not by using only the features around that word. This type of approach was also used in Japanese morphological analysis, which is a task to detect boundaries of the smallest meaningful units because Japanese text has no spacing between each word. The pointwise prediction for Japanese morphological analysis [Neubig and Mori, 2010; Neubig *et al.*, 2011][3] achieved high results. The advantages of this pointwise prediction are its robustness and adaptiveness. Presently, many prediction techniques based on machine learning exist. Machine learning, specifically recurrent neural network (RNN), remarkably depends on features, and when one of the features is wrong, the results through these machine learning can also be wrong. However, the effect of the wrong feature or missing feature information is supposed to be local when this type of pointwise prediction is used. In addition, prediction using machine learning techniques apart from the pointwise prediction, particularly RNN, requires many training data, but the training data in this task are limited. Nevertheless, by using the pointwise prediction, the training data become more abundant than that of other machine learning techniques because the number of sentences is limited and much less than the number of candidates for sentence boundaries. We supposed that the pointwise prediction has some advantages on this prediction task, that is, sentence boundary detection with noise.

For this shared task, we submitted two test predictions, one of which is the result of our pointwise model and the other one is that of a simple rule-based model only. Here, we focused on describing the first one, and the latter is treated as a baseline model. In addition, we used the script that was distributed by the organizer of this shared task for evaluations.

2 Pointwise Prediction Model

First, we explain our pointwise prediction model. Figure 1 shows the model outline.

The proposed prediction model uses words around the target point, which is to be classified into the beginning of a sentence, the end of a sentence, and others. That is, by using a window size N_W, our prediction model utilizes $(N_W \times 2+1)$ words, including a target point word and N_W words before and after the target point words. In Figure 1, a window size of 4 is employed. These words to be input into a three-layered perceptron also are embedded into vectors.

Next, we describe details about words embedding and the multi-layer perceptron.

[*]Contact author; https://mhirano.jp/
[1]Our code is available on https://s.mhirano.jp/FinSBD2019
[2]https://sites.google.com/nlg.csie.ntu.edu.tw/finnlp/shared-task-finsbd
[3]This Japanese morphological analyzer is called "KyTea."

Proceedings of the First Workshop on Financial Technology and Natural Language Processing (FinNLP@IJCAI 2019), pages 102-107, Macao, China, August 12, 2019.

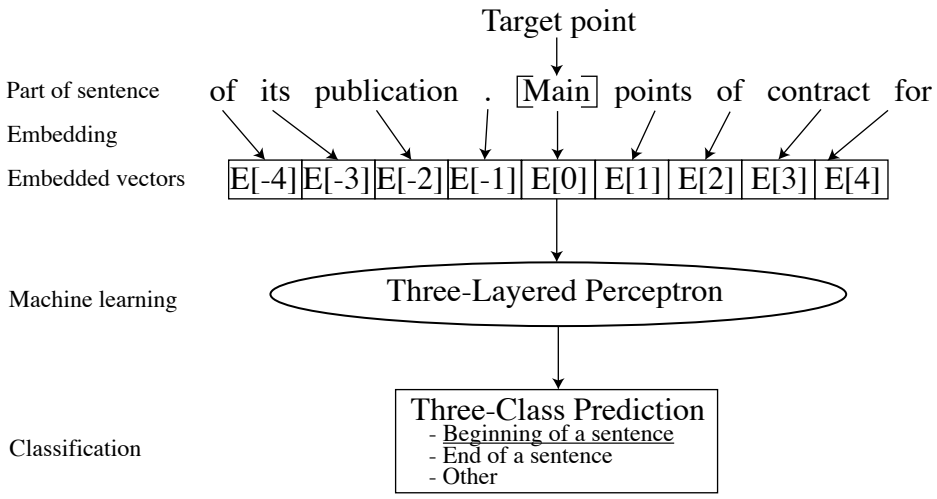

Figure 1: Pointwise prediction outline. This example employs a window size of 4. Words around the target point are gathered and embedded. The embedded vectors are going through a three-layered perceptron and making a prediction for a target point. The prediction has three classes: the beginning of the sentence, the end of the sentence, and other.

2.1 Words Embedding

For words embedding, we use three types of factors. Here are the features for word embedding:

1. Word2vec (`gensim` in `python3`)
2. Part-of-speech(POS) tag (NLTK POS tag)
3. CAPITAL/non-capital patterns for each character in a word
4. Whether including line feed code or not

Word2vec [Mikolov *et al.*, 2013] is a method of translating words into vectors. In this task, we used the training data from this shared task organizers for the word2vec training data, and the word2vec window size is set equal to the window size of our model N_W. Moreover, the word embedding dimension of word2vec N_D is set as a hyperparameter. We employed the vectors through this word2vec as the first N_D factors in the embedded vectors.

The POS tag is fetched from the Natural Language Toolkit (NLTK) [Bird, 2006]. This NLTK's POS tag data is obtained via `nltk` in `python3` and contains 45 types of POS patterns. In case the POS tag classifications in `nltk` fail, we added one additional class to the original 45 classes from NLTK. Hence, this 46-class data are translated into one-hot vectors and used as the subsequent 46 factors in the embedded vectors.

The CAPITAL/non-capital patterns for each character in a word are for recognizing words' appearance patterns, such as "This," "this," "THIS," and others, including numbers and symbols. We categorized each word into the following patterns:

1. Numbers and/or characters (e.g., 2, #, -, .).
2. All characters are alphabet and non-capital (e.g., this, have).
3. All characters are alphabet and CAPITAL (e.g., THIS, HAVE).
4. All characters are alphabet and only the first character is CAPITAL (e.g., This, Have).
5. Others that are not classified above.

These five classifications also are translated into one-hot vectors and used as the next five factors of the embedded vectors.

The last one factor of the embedded vectors is whether to include the line feed code or not. In the data, line feed code is "\n."

The details of the embedded vectors are described previously, and the embedded vectors for each word have a total of $N_D + 46 + 5 + 1 = N_D + 52$ factors. In addition, the embedded vectors of $2N_W + 1$ words are concatenated and input into the subsequent three-layered perceptron; hence, the total input vectors' length become $(N_D + 52) \times (2N_W + 1)$.

2.2 Three-Layered Perceptron

We employed a three-layered perceptron as the classification model. This three-layered perceptron has one input, one output, and two hidden layers. The input layer has $(N_D + 52) \times (2N_W + 1)$ nodes, both the hidden layers have N_H nodes each, and the output layer has three layers. The input and first hidden layers, the first and second hidden layers, and the second hidden and output layers are fully connected. Additional details are presented in Appendix A.

3 Rule-based Prediction Model (Baseline Model)

The rule-based prediction model that we developed is remarkably simple and is constructed as a baseline. This model has two features definition: namely, features definition for beginnings and endings. The features definition of beginnings are as follows:

1. The first character in a word is CAPITAL, whereas the other characters are non-capital.

2. The word is "-" (hyphen).
3. The word is "(" (left bracket) and the subsequent of the next word is ")" (right bracket).
4. The word is "` " (backquote).

The features definition of endings are as follows:

1. The word ending with ".\n" (a period and a line feed code).
2. The word is "." (a period) and the previous and subsequent words are not digits.
3. The word ending with ";" (semi-colon).
4. The word ending with ";\n" (a semi-colon and a line feed code).
5. The word ending with ":\n" (a colon and a line feed code).
6. The word ending with "\n" (a line feed code)[4].

Then, our rule-based model counts these features for each word.

The next process is deciding where the sentences begin and end based on the numbers of features of begins/ends. Basically, the model supposed that the numbers of features of begins/ends are not zero indicate the possibilities of begins/ends. It finds the pairs of the beginning and ending of the sentence that has a minimum length. If the words that have possibilities of the same type of features comes continuously (e.g., the words with possibilities of the beginning of the sentence come again without the appearance of the words having possibilities of the end of the sentence), then we adopt the word that has the highest number of features as the beginning or ending. The algorithm details are presented in Appendix B.

4 Experiments

We were provided with the data by the organizers of this shared task, containing the "training data" and "development data" for two languages, i.e., English and French. We performed the same experiments on each language. In addition, we treated the "training data" as training data and the "development data" as test data. (Actually, the organizers provided also the "test data" to form the leaderboard of this shared task, but we ignored these data in this paper other than final results.) These data contain sentences from the PDF Noisy Text but were split well for each word, symbols, or something, the list of the beginning of the sentences, and the list of the beginning of the sentences [5].

Using these data, we tested our model. In our model, several unfixed hyperparameters are the following:

1. N_W: window size;
2. N_D: the dimension for word2vec; and
3. N_H: the number of nodes on every single hidden layer in the three-layered perceptron.

[4]This feature sometimes overlaps with other features, but when it is, we count it redundantly.

[5]Details are shown in https://sites.google.com/nlg.csie.ntu.edu.tw/finnlp/shared-task-finsbd or [Ait Azzi *et al.*, 2019].

We modified these hyperparameters as in Table 1 and tested the model.

Hyperparameter	Candidates
N_W	5, 10, 200
N_D	10, 50, 100
N_H	300, 600, 1200

Table 1: Hyperparameter candidates for each hyperparameter.

Apart from these parameter sets, we ensembled the results of all models. In the ensembling process, only the words that two-thirds of all models agree with become the beginning/ending of the sentences.

5 Results

N_W	N_D	N_H	BS	ES	Ave.
Ensemble			0.84	0.92	0.88
5	10	1200	0.83	0.92	0.88
10	100	600	0.82	0.91	0.87
5	50	600	0.82	0.91	0.87
10	50	1200	0.81	0.91	0.86
10	100	300	0.81	0.91	0.86
5	10	600	0.81	0.91	0.86
10	50	300	0.81	0.91	0.86
10	10	1200	0.82	0.90	0.86
5	100	1200	0.82	0.90	0.86
5	50	1200	0.82	0.90	0.86
20	50	1200	0.80	0.91	0.86
5	100	600	0.81	0.90	0.86
10	10	600	0.81	0.90	0.86
10	100	1200	0.81	0.90	0.86
20	100	600	0.80	0.91	0.86
5	10	300	0.81	0.90	0.86
20	100	1200	0.80	0.90	0.85
10	10	300	0.81	0.89	0.85
20	10	1200	0.79	0.91	0.85
20	10	600	0.79	0.91	0.85
5	50	300	0.80	0.89	0.85
20	50	600	0.79	0.90	0.85
5	100	300	0.80	0.89	0.85
20	50	300	0.78	0.90	0.84
20	100	300	0.78	0.90	0.84
20	10	300	0.77	0.89	0.83
10	50	600	0.74	0.80	0.77

Table 2: Results of the English language using the pointwise prediction model and its hyperparameter sets. Here, BS/ES indicates f1-score for the prediction of the beginning/ending of the sentences and Ave. denotes the average of BS and ES.

Each result for each parameter sets using pointwise prediction model are shown in Tables 2 and 4. Tables 2 and 3 are results for English and Tables 4 and 5 are results for French. An evaluation tool is provided by the organizers of this shared task. The tool calculates the f1-scores for the prediction of

Ensemble model (En)	Precision	Recall	F1-score
Others	1.00	0.99	0.99
Beginnings	0.81	0.87	0.84
Ends	0.87	0.98	0.92
Micro avg	0.99	0.99	0.99
Macro avg	0.89	0.95	0.92
Weighted avg	0.99	0.99	0.99

Table 3: Detailed results for English using the pointwise prediction ensemble model.

N_W	N_D	N_H	BS	ES	Ave.
Ensemble			0.80	0.88	0.84
10	10	1200	0.79	0.86	0.83
10	50	1200	0.78	0.86	0.82
10	100	1200	0.77	0.87	0.82
10	100	600	0.77	0.86	0.82
20	50	1200	0.75	0.86	0.81
10	10	600	0.76	0.85	0.81
10	50	600	0.77	0.85	0.81
20	100	1200	0.76	0.86	0.81
5	10	1200	0.74	0.85	0.80
20	10	1200	0.73	0.86	0.80
10	50	300	0.73	0.84	0.79
20	10	600	0.73	0.84	0.79
20	50	600	0.74	0.84	0.79
20	100	600	0.74	0.84	0.79
10	10	300	0.72	0.84	0.78
5	10	600	0.72	0.83	0.78
10	100	300	0.72	0.83	0.78
20	100	300	0.71	0.83	0.77
5	100	600	0.70	0.83	0.77
5	100	1200	0.71	0.83	0.77
5	50	1200	0.72	0.82	0.77
5	50	600	0.72	0.82	0.77
20	50	300	0.69	0.82	0.76
5	10	300	0.70	0.82	0.76
5	50	300	0.69	0.80	0.75
5	100	300	0.67	0.80	0.74
20	10	300	0.66	0.81	0.74

Table 4: Results of the French language using the pointwise prediction model and its each hyperparameter sets. Here, BS/ES indicates f1-score for the prediction of the beginning/ending of the sentences and Ave. denotes the average of BS and ES.

Ensemble model (Fr)	Precision	Recall	F1-score
Others	0.99	0.99	0.99
Beginnings	0.74	0.86	0.80
Ends	0.82	0.94	0.88
Micro avg	0.98	0.98	0.98
Macro avg	0.85	0.93	0.89
Weighted avg	0.98	0.98	0.98

Table 5: Detailed results for French using the pointwise prediction ensemble model.

	BS	ES	Ave.
Pointwise Prediction (English)	0.84	0.92	0.88
Baseline: Rule-based (English)	0.73	0.81	0.76
Pointwise Prediction (French)	0.80	0.88	0.84
Baseline: Rule-based (French)	0.67	0.69	0.68

Table 6: Comparing the results of pointwise prediction with ensembling and the results of the baseline model, i.e., rule-based model.

	Test	Final result
Pointwise Prediction (English)	0.88	0.84
Baseline: Rule-based (English)	0.76	0.63
Pointwise Prediction (French)	0.84	0.86
Baseline: Rule-based (French)	0.68	0.68

Table 7: Test and final results of all models and languages. The final result is the result of this shared task and was on the leaderboard.

the beginning/ending of the sentences and others. F1-score denotes the harmonic average of precision and recall.

Table 6 shows the comparison between the results of the pointwise prediction model with ensembling and the results of the baseline model, i.e., rule-based model. In both English and French, our pointwise prediction model outperformed our baseline model in any index.

Table 7 shows the test results from Table 6 and the final results from this shared task's leaderboard. Predictions for the French language in the test and final results have a slight difference, whereas those for the English language have significant gaps. Specifically, the rule-based model in English performs worse in the final results.

6 Discussion

First, based on Table 6, the proposed pointwise prediction model outperformed the rule-based prediction model. Evidently, the reason for the accurate predictions is that our pointwise prediction model employs a type of feature learning. Rules for the rule-based model were implemented by us and the rules are not sufficient. By using the feature learning, these processes, such as implementing rules, are not necessary and render the model more adaptive for wide cases. We employed only a simple three-layered perceptron, but this model worked well, as observed in the final results' leaderboard.

Second, the result of pointwise predictions using various parameters suggests interesting insights. Models with $(N_W, N_D, N_H) = (10, 100, 600), (5, 10, 1200)$, and $(5, 50, 600)$ show the highest results in both English and French. Table 8 shows the top results in various parameters. As observed in this table, $N_W = 20$ does not appear. Thus, the window sizes are limited when the results are high. This result shows that the beginning/ending of sentences can be recognized only by watching approximately 5–10 words around. In addition, the size of the hidden layer of the three-layered perceptron N_H of the top results tends to be high.

N_W	N_D	N_H	En	Fr	Ave.
10	100	600	0.87	0.83	0.85
5	10	1200	0.88	0.82	0.85
5	50	600	0.87	0.82	0.85
10	50	300	0.86	0.82	0.84
10	50	1200	0.86	0.81	0.84
10	100	1200	0.86	0.81	0.84
5	50	1200	0.86	0.81	0.84

Table 8: Top of the results in various parameters. Here, Ave. denotes the average of the results of both English and French.

However, when $N_W = 10, N_D = 100$, the input length for the three-layered perceptron must be 3192. Only when $N_W = 5, N_D = 10$, it must be 682. These results suggest that additional hidden layer sizes are possibly necessary

Third, the final result predictions in English were noticeably worse than those of the test results, whereas the French data were not. We assume that the English test data for the final results include some out-of-sample data. Both the pointwise prediction and rule-based models perform poorly; the characteristics of the data might be the cause of these lousy results.

As future works, we have to test additional parameters for N_W, N_D, N_H. Moreover, we must change the fixed parameters in Appendix A. Not only the parameters but also the feature learning models, apart from the three-layered perceptron, should be evaluated for their accuracy.

7 Conclusion

In this paper, we presented the application approach of pointwise prediction to sentence boundary detection in the PDF Noisy Text in the financial domain for the FinSBD 2019 shared task. Our point prediction model achieved 0.88 and 0.84 averaged f1-score for the beginning/ending of sentences in English and French. In the final results, this model obtained 0.84 in English and 0.86 in French. Evidently, the proposed pointwise prediction model outperformed the rule-based prediction model in any index. In our model, we employed some sets of parameters and ensembled models with these parameter sets. The result shows that the ensemble models outperformed any model without ensembling. However, other parameter sets that are also accurate for this task are possible. Moreover, we fixed some parameters. As future works, these parameters should also be modified.

Acknowledgments

Funding from Daiwa Securities Group is gratefully acknowledged.

A Details of the Three-Layered Perceptron

Table 9 shows the parameters of the employed three-layered perceptron. These parameters are totally fixed and not optimized. Therefore, we will enhance these parameters in the future. As observed in table 9, the maximum training epoch is 10,000. However, training was forced to be stopped using Algorithm 1.

	Parameter
Activation function	relu
Loss function	Cross entropy loss
Optimize function	SGD
Dropout	True
Dropuout rate	0.2
Learning rate	0.01
Momentun	0.9
Weight decay	5×10^{-4}
Training data split	90% for training, 10% for evaluating.
Batch size	1,000
Max training epochs	10,000

Table 9: Parameters for three-layered perceptron

Algorithm 1 The algorithm for stop training

Input: Training data
Output: The best performed model

$step \leftarrow 0$.
$max\_acc \leftarrow 0$, $max\_step \leftarrow 0$, $acc\_list \leftarrow []$.
loop
  Train(90% of training data).
  $acc \leftarrow$ Evaluate(10% of training data).
  $acc\_list[step] \leftarrow acc$.
  if $acc \geq max\_acc$ **then**
    $max\_acc \leftarrow acc$.
    $max\_step \leftarrow step$.
    SaveModel().
  end if
  if $step \mod 10 = 0$ **and** $step \geq 100$ **then**
    Calculate all 100 steps moving average of acc_list.
    $M \leftarrow$(Step with the best moving average).
    if $Step > \max(M \times 1.1, M + 200)$ **then**
      Break.
    end if
  end if
  $step \leftarrow step + 1$.
end loop
return max_acc, max_step

B Detailed Algorithm of the Rule-based Prediction Model

The algorithm is shown in Algorithm 2.

Algorithm 2 The algorithm for rule-based prediction

Input: The list of the numbers of features of the beginning/end of the sentences fob, foe
Output: The list of the beginning bos and the list of the end of the sentences eos

$i \leftarrow 0$.
$bos \leftarrow []$, $eos \leftarrow []$, $status \leftarrow False$.
$last\_start \leftarrow 0$, $last\_end \leftarrow 0$.
loop
  if $fob[i] = 0 \wedge foe[i] = 0$ **then**
    Go to 34.
  else if $fob[i] = 0 \wedge foe[i] \neq 0$ **then**
    if $status$ **then**
      Append $last\_start$ to bos.
      $status \leftarrow False, last\_end \leftarrow i$.
    else if $foe[last\_end] < foe[i]$ **then**
      $last\_end \leftarrow i$.
    end if
  else if $fob[i] \neq 0 \wedge foe[i] = 0$ **then**
    if $status$ **then**
      if $fos[last\_start] < fos[i]$ **then**
        $last\_start \leftarrow i$.
      end if
    else
      Append $last\_end$ to eos.
      $status \leftarrow True, last\_start \leftarrow i$.
    end if
  else
    if $status$ **then**
      Append $last\_start$ to bos.
      Append i to eos.
      $last\_start \leftarrow i$.
    else
      Append i to bos.
      Append $last\_end$ to eos.
      $last\_end \leftarrow i$.
    end if
  end if
  $i \leftarrow i + 1$.
end loop
return bos, eos

References

[Ait Azzi *et al.*, 2019] Abderrahim Ait Azzi, Houda Bouamor, and Sira Ferradans. The FinSBD-2019 Shared Task: Sentence boundary detection in PDF Noisy text in the Financial Domain. In *The First Workshop on Financial Technology and Natural Language Processing (FinNLP 2019)*, Macao, China, 2019.

[Bird, 2006] Steven Bird. NLTK: The Natural Language Toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[Mikolov *et al.*, 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of Twenty-seventh Conference on Neural Information Processing Systems (NeurIPS 2013)*, pages 3111–3119, Stateline, Nevada, USA, 2013.

[Neubig and Mori, 2010] Graham Neubig and Shinsuke Mori. Word-based Partial Annotation for Efficient Corpus Construction. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*, pages 2723–2727, Valletta, Malta, 2010. European Language Resources Association.

[Neubig *et al.*, 2011] Graham Neubig, Yosuke Nakata, and Shinsuke Mori. Pointwise Prediction for Robust , Adaptable Japanese Morphological Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, pages 529–533, Portland, Oregon, USA, 2011.