Infomation Extraction from English & Japanese Résumé with Neural Sequence Labelling Methods

Akihiro Katsuta[◇*] Hutama Adhi Hanjaya[♣] Somnath Asati[♣]
Sorami Hisamoto[♡] Kazuma Takaoka[♡] Yoshitaka Uchida[♡] Yuji Matsumoto[♠]

[◇]*Nagaoka University of Technology*
[♣]*Works Applications Singapore* [♡]*Works Applications*
[♠]*Nara Institute of Science and Technology*

katsuta@jnlp.org
{hanjaya_h, asati_s, hisamoto_s, takaoka_k, uchida_yo}@worksap.co.jp
matsu@is.naist.jp

Abstract

We extract information from résumés by formalizing it as a sequence labelling problem. We employ neural network methods to conduct sequence labelling.

We prepare our own annotated résumé datasets for both English and Japanese. There are token/phrase level labels and sentence/paragraph level labels.

Preliminary experiment shows it is necessary to consider the structure of document, instead of processing each line independently, in order to have better results.

1 Introduction

We aim to automatically extract information such as skills or career history from résumé documents. If we can create a database storing applicants' profiles, based on that information we can make the recruitment process much more efficient and effective.

There exists previous work that formalizes information extraction from a résumé as Named Entity Recognition (NER) problem. NER is a task to extract entities such as places or proper nouns from text[12], and is often solved using sequence labelling approaches. In recent years it is popular to use neural network methods such as Long-Short Term Memory (LSTM) or Convolutional Neural Network (CNN) for sequence labelling problems, and it has been shown to give good results on standard NER benchmarks[14]. In this research we formalized our goal, information extraction from résumés, as a sequence labelling NER task, and use neural network methods to solve it.

We consider both English and Japanese résumés.

There are two different types of résumés in Japanese, namely Rirekisyo (履歴書) and Syokumukeirekisyo (職務経歴書). In the former type, you write a fixed set of personal information, and it often has predefined layouts. Meanwhile the purpose of latter type is to write your self introduction or job description, and the content and layout are different for each applicant. Therefore we might need different approaches for each type, and information extraction from latter type is more complicated. In our experiments we focus on the latter type only for now. For English résumés, you can freely decide the content of the document, therefore it is more similar to the latter type.

We expect to receive résumés in PDF format. To solve it as a sequence labelling task, we pre-process the PDF to extract raw text from the file. We also expect the files are digitally generated, therefore handwritten or scanned files which need Optical Character Recognition (OCR) are out of our scope for now.

There are no publicly available datasets for résumé NER. Therefore we first create or collect résumés and annotate them, then use that datasets to conduct experiments.

2 Related Work

As we described in previous section, NER is often formalized as a sequence labelling task. It is common to use Conditional Random Fields (CRF)[7] as a model to solve the problem.

In previous researches such as the ones by Yu et

*This work was conducted while the first author stayed at Works Applications for internship.

al.[18] and Chen et al.[1], they take multiple steps to extract information from résumés, namely preprocessing, block detection & classification, and sequence labelling.

There exists a previous work on information extraction from Japanese rirekisyo (履歴書)[17]. They simply used TF-IDF to extract characteristic keywords as skills.

Collobert et al.[3] started to work on NER using unified neural network. Following this structure, Chiu and Nichols[2] combined CNN and LSTM, and Huang et al.[6] integrated CRF to solve NER. After these works, it is now common to use character level embedding, pre-trained word embedding, bidirectional LSTM (BiLSTM) layer, and CRF on the top as you can see in BiLSTM-LSTM-CRF[8] or BiLSTM-CNNs-CRF[9]. The difference between these two models are to use LSTM or CNN for character level embedding. It is also shown in previous work[14] that the differences between these models' results are not necessary statistically significant, and you should be careful when you compare the results as they are non-deterministic approaches.

Misawa et al.[11] looked into NER on Japanese data using these neural network methods. They stated that sub-word information with CNN is not effective for Japanese data, and also there could be boundary problem if the word and entity have different boundaries. Given the situation they proposed a character-based NER model.

3 Datasets

For our experiments, we prepared datasets for English résumé and Japanese Syokumukeirekisyo (職務経歴書) in PDF format. The overview of each dataset is shown in table 1.

Table 1: Résumé dataset statistics

	English	Japanese
Number of documents	543	142
Token length average	675	1097
Token length S.D.	339.4	436.0

To conduct sequence labelling, we extracted raw text from résumés in PDF format. To extract raw text from PDF, we used a tool called `PDFBox®`[1]. It extracts text per visual line as we can see on the document, and not per sentence. We fix this manually for now, and we would like to do this process automatically in the future.

We used `BRAT`[16] for manual annotation.

We defined 64 labels we want to extract from Japanese résumés. There are token/phrase level labels such as *skill*, *certificate*, or *company name*, and also sentence/parapraph level labels such as *self introduction* or *job description*. For English résumés, we defined 15 labels; all of them are token/phrase level labels.

For Japanese data, the top 3 labels with longest length are *job description* with 40.73% of tokens in document, *profile* with 22.31%, and *career summary* 8.03%. We expect to use these labels for sentence/paragprah level information, and we can see that the large fraction of the tokens in documents are of this type.

4 Experiments

We used a pre-existing tool called `NeuroNER`[4] to conduct NER using neural network methods. NeuroNER employs BiLSTM-LSTM-CRF, that means character level embedding is done via LSTM layer. You may turn off the character level embedding to make it BiLSTM-CRF. NeuroNER accepts BRAT format input/ouput, this made our annotation and experiment process smoother.

Previous work[11] stated character level information is not useful for Japanese data. To check the effectiveness of character level embedding, we tried the version with and without this layer. We also tried the experiments with and without pre-trained word embeddings, resulting in the following four settings.

- None: Without character level embedding, without pre-trained word embedding
- Char: With character level embedding, without pre-trained word embedding
- PreTrainedWord: Without character level embedding, with pre-trained word embedding
- Char+PreTrainedWord: With character level embedding, with pre-trained word embedding

For English we used `GloVe`[13] embedding pretrained with Wikipedia 2014 dump and English Gigaword 5th Edition[2]. For Japanese pre-trained word embedding, we used `nwjc2vec`[5] which used NWJC[3] corpus and word2vec[10] model to learn. This embedding is based on UniDic[4] morphological unit, therefore we conducted morphological analysis on our Japanese data using UniDic resources.

[1] https://pdfbox.apache.org/

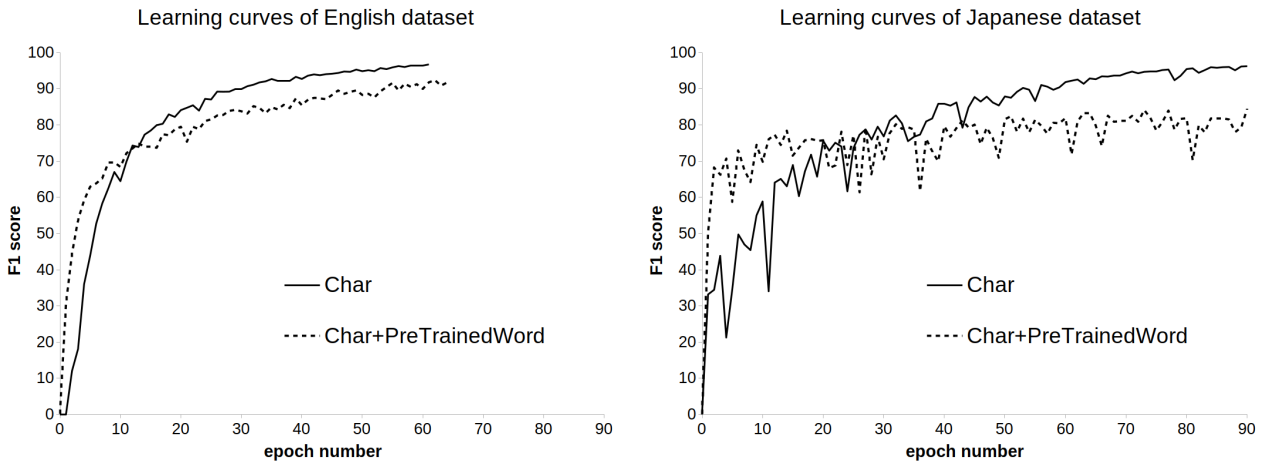

Fig. 1: Learning curves in traning phase; English models on left and Japanese on right. You can see the curve for Japanese model using pre-trained word embedding is not converging well compared to others.

5 Results & Discussion

We show the accuracy of résumé NER experiments for the English dataset in table 2. The results are calculated as per-label micro average.

The purpose of the preliminary experiment is to see the rough tendency, and we did not do cross validation or other procedures necessary for more proper analysis. Therefore the small differences in numbers are not meaningful.

Table 2: NER results on English dataset

	precision	recall	F1
None	71.34	62.33	66.53
Char	75.42	67.63	71.31
PreTrainedWord	73.99	72.50	73.24
Char+PreTrainedWord	76.40	75.74	76.07

In general we can see that both character embedding and pre-trained word embedding are helpful to improve the accuracy as we expected.

Effectiveness of character embedding makes sense in English as first character caplitalization or prefix/sufix can be good source of information to tell what kind of token it is.

Pre-trained word embedding is expected to be useful as we can make use of information learned from large scale raw text. When we trained the model with smaller labelled dataset, the model already gave a result with decent accuracy. This can be attributed to the significant influence of pre-trained word embedding.

Next we show the results for the Japanese dataset in table 3.

[2]http://nlp.stanford.edu/data/glove.6B.zip
[3]http://pj.ninjal.ac.jp/corpus_center/nwjc/
[4]http://unidic.ninjal.ac.jp/

Table 3: NER results on Japanese dataset

	precision	recall	F1
None	68.79	69.94	66.00
Char	74.70	77.17	74.81
PreTrainedWord	71.56	70.08	66.67
Char+PreTrainedWord	63.03	70.72	64.38

We can see a different trend compared to the English result. When we compare each result, Char (with character embedding, without pre-trained word embedding) setting has better accuracy compared to others. This is surprising, as intuitively using pre-trained word embedding learned from large scale dataset will give more information to the network.

Looking at the learning curve of the Japanese models, it seems that the learning of ones with pre-trained word embedding did not converge very well, whereas the ones without pre-trained word embedding, learning curve were smoother. For English, the learning curves were smooth for all settings. In figure 1 we show the learning curves of English and Japanese models in training phase.

A common case we observed in the results is that the model labels only a few tokens, even when we want it to label entire sentence or paragraph. We had more that cases when using pre-trained word embedding.

Another common case we noticed is the lines that are ambiguous without context information. We feed each line in document independently to the model. However you often cannot determine the correct label unless you know in which part of the document it appears. For example, the same sentence can be interpreted as either part of *self introduction* or *carrer summary*, depending on which section it appears witin the document.

To summarize, we can see that there may be two reasons that make this problem difficult.

- Sentence/Paragraph Level Labels
  It is not efficient to do token level sequence labelling for such cases. It would be more appropriate to first detect the sentence/paragraph block, and then classify each block into a label.

- Document Structure
  The same tokens or phrases can be labeled differently depending on where in the document they appear. The model does not have enough information to correctly predict if the input is an individual sentence.

6 Future Work

Given the reasons in the last section, to improve the accuracy we first consider how to get the document structure.

Currently we are working on section header detection and classification, and we are getting good results. Given this header information, we can split the document into different blocks. From the header we know what kind of block this is, and depending on the type we may be able to give a single label to the entire block, or do more fine NER within the block. We can give this block information to the NER model, and it can make use of this information to better predict the label.

Our dataset, PDF files, contains position and other visual information such as font type and size. We can incorporate these clues for header detection and NER, as they give richer information beyond raw text.

For some labels it is easier if we use dictionaries (e.g., company or school) or patterns (e.g., date or email address). We may do so in parallel to the neural network NER, or we may give these clues from dictionaries or patterns as the input to the network.

Sato et al.[15] proposed segment-level neural CRF, which considers segment-level sequence labelling. With this approach it is easier to incorporate entity level features, as entities are often longer than a token, therefore it may be suitable for our problem.

Lastly, we can pre-train or further fine-tune the word embedding using related dataset such as job posting. If we have large scale text of such data to learn from, the resulting embedding may be more suitable for our problem.

References

[1] J. Chen, L. Gao, Z. Tang. Information extraction from resume documents in pdf format. In *Document Recognition and Retrieval*, 2016.

[2] J. P. Chiu, E. Nichols. Named entity recognition with bidirectional lstm-cnns. *TACL*, 4, 2016.

[3] R. Collobert *et al.* Natural language processing (almost) from scratch. *JMLR*, 12, 2011.

[4] F. Dernoncourt, J. Y. Lee, P. Szolovits. Neuroner: an easy-to-use program for named-entity recognition based on neural networks. *CoRR*, abs/1705.05487, 2017.

[5] S. Hiroyuki *et al.* nwjc2vec: Word embedding data constructed from ninjal wed japanese corpus (in japanese) / nwjc2vec: 国語研日本語ウェブコーパスから構築した単語の分散表現データ. *Journal of NLP*, 24, 2017.

[6] Z. Huang, W. Xu, K. Yu. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991, 2015.

[7] J. Lafferty, A. McCallum, F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289, 2001.

[8] G. Lample *et al.* Neural architectures for named entity recognition. In *NAACL-HLT*, pages 260–270, 2016.

[9] X. Ma, E. Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL*, pages 1064–1074, 2016.

[10] T. Mikolov *et al.* Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119. 2013.

[11] S. Misawa *et al.* Character-based bidirectional lstm-crf with words and characters for japanese named entity recognition. *EMNLP*, 2017.

[12] D. Nadeau, S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30, 2007.

[13] J. Pennington, R. Socher, C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014.

[14] N. Reimers, I. Gurevych. Reporting score distributions makes a difference: Performance study of lstm-networks for sequence tagging. In *EMNLP*, pages 338–348, 2017.

[15] M. Sato *et al.* Segment-level neural conditional random fields for named entity recognition. In *IJCNLP*, pages 97–102, 2017.

[16] P. Stenetorp *et al.* brat: a web-based tool for NLP-assisted text annotation. In *EACL*, pages 102–107, 2012.

[17] M. Takahiro *et al.* Skill discovery and matching based on feature words extraction from senior people's resumes (in japanese) / 高齢者の履歴書からの特徴語抽出によるスキルの発見とマッチング. *IEICE Technical Report. NLC*, 112, 2012.

[18] K. Yu, G. Guan, M. Zhou. Resume information extraction with cascaded hybrid model. In *ACL*, pages 499–506, 2005.