

User-Assisted Tabular Extraction in Japanese Invoice

Vincent Leonardo Yuichiro Shimizu Kensuke Masugata
 Sorami Hisamoto Yoshitaka Uchida Kazuma Takaoka

Works Applications Co., Ltd.

{leonardo_v, shimizu_yu, masugata_k, hisamoto_s, uchida_yo, takaoka_k}@worksap.co.jp

Abstract

We present a user-driven approach to extract tabular information from Japanese business document. Following the previous researches Bart and Sarkar (2010)[1] of the same task, we further extend the approach to cover Japanese business document. The method utilizes a set of fields (on the first row of a table) provided by the user to extract the remaining rows that match the structure of the provided fields set. Firstly, a set of potential field candidates is generated and evaluated against the user-selected fields based on an extended set of spatial features. Another level evaluation is performed to verify the structure similarity based on inter-field spatial relationship. Both evaluation models were trained through Automated Machine Learning (AutoML). On top of that, we added a simple re-evaluation scheme which proven to reduce the false-negatives case. Pairing it up with the evaluations, we achieved an accuracy improvement of 4.56%. We demonstrate that this method is able to handle a wide variety of tabular structure regardless of the language of the documents.

1 Introduction

Invoice processing is a common business operation performed regularly. Although a large number of businesses have adopted the usage of electronic invoices, there are still a handful of companies that use paper invoices. This is especially true in a country such as Japan, where ink stamp is still widely preferred over digital signature as a proof of authenticity. On the other hand, the record keeping and processing have mostly been digitized (as Enterprise Resource Planning system) due to the benefit of storage efficiency and ease of computation. This introduces the needs of a method to digitize all paper records.

With the lack of a reliable extraction system, most companies perform such extraction task manually. A survey [10] shows that manually processing an invoice is estimated to cost around 9 Euros and can result in an average error rate of 5%. A separate survey [19] shows that the adoption of automatic invoice processing may reduce the cost to around 2 Euros per invoice.

In this paper, we propose a user-assisted method for extracting tabular information in business documents. The user first selects a set of fields; each field corresponds to a column of interests to the user and the set of fields represents a single row of entry within the tabular structure. The system will utilize the underlying structural relationship between the fields to search for other combinations of fields that match the structure of user-selected fields. The red boxes in Figure 1 represent the user selections and the green boxes are fields combinations with similar structure inferred by the system.

2 Related Works

While there exists a plenty of researches (reviewed in [2, 3, 5, 9, 20]) on table extraction, most hold strong assumptions on certain visual cues of a table. This includes table lines, white spaces, columns/rows text alignments, heuristic rules, keywords matching and database of pre-defined layout structure; or any combination of them. Furthermore, most of the researches mainly focus on extracting tables from academic papers. Such approaches are hard to be applied to invoice documents where there is a large variety of layouts depending

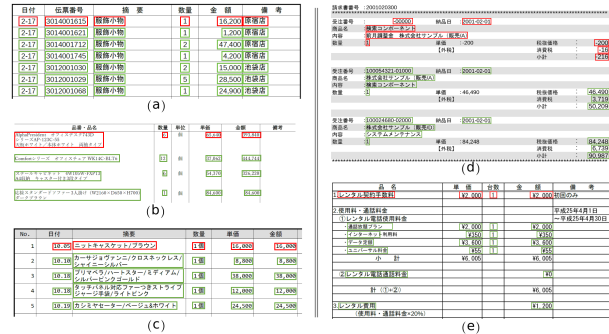


Figure 1: Conventional table structure: (a) common single-row linear structure (b) multi-line linear structure [multi-line input] (c) multi-line linear structure [single-line input]. Complex tabular structure: (d) an extreme case of complex tabular structure (e) optional columns and variations in alignment & text size. [red: input, green: targets]

on the nature of the business. There are also some table extraction methods that rely on the underlying document format such as PDF, XML or HTML (e.g., *PdfExtra*[4], *TEXUS*[14], *Tabby*¹[15], *tabula*², and *camelot*³).

A more recent researches [13, 8] model representation of the entire page contents through Convolutional Neural Network. Both methods are able to handle variation in layouts. They also show improvement over state-of-the-art sequential information extraction framework in tackling table extraction problem. However, due to the nature of deep learning, it requires a large amount of training data (25,200 and 12,000 in [13] and [8] respectively). Unfortunately, there is no open-source dataset on invoice table extraction and creating such a large dataset on our own is very costly.

All these approaches extract the entire table from the document without user interaction and let the downstream task decides which information to present to the user. There are other researches [1, 6] that focus on user-driven extraction which extracts only the relevant results based on user query. These approaches are able to handle layout variations and also extract table based on user-specified requirements; which make the system more generalizable to various documents.

The state-of-the-art method in [6] formalizes user selection as a query graph and extract similar patterns from the document through graph mining. Although the methods show accuracy improvement over [1], they utilize textual features as feature vectors which is language sensitive. The textual features such as string length, number of words and words separation is a weaker representation in Japanese document due to the lack of spacing in Japanese sentence. Some researches [7, 11, 17] on various information extraction tasks touched on the differences in using linguistic features between English and Japanese documents.

The method presented in this paper is an extension of the work presented in [1] where they demonstrate the effectiveness of extracting repeating pattern based on a set of user selected

¹<https://github.com/cellsrg/tabbypdf>

²<https://tabula.technology/>

³<https://camelot-py.readthedocs.io/>

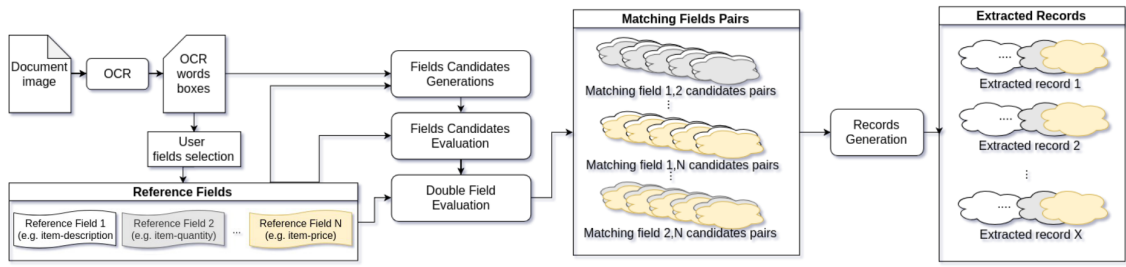


Figure 2: Proposed tabular extraction flow

fields with only the perceptual cues. The authors also show that the approach adapts well to the various form of documents including receipts, medical entries, table of contents and search results. This paper will focus on solving the issue presented in [1] where they are unable to handle missing field(s) which in turn causes the entire record to be missed. The missing fields could be due to OCR errors or the content itself being optional.

3 Methodology

In this paper, we define tabular structure as a list of information organized in a common structure repeated vertically along the page(s). Such structure does not necessarily need to be in the conventional table style, Figure 1(d)(e) show some variations of unconventional layout.

We will use the same terminologies defined in [1]. An entry of the repeating structure is referred as *record* while all the items of interest in the entry are referred as *field(s)*. The *fields* are laid out in a consistent spatial structure across the records (within the document). A *field* can be a single word returned by the OCR system or a sequence of contiguous words in a single line or multiple lines. For example, each box in Figure 3 is a *field/cell* and a group of fields of the same colour is a *record/row*.

In a conventional table structure, the term *record* is interchangeable with *row*, so does *field* with *cell*

請求書番号	:2001020300		
受注番号	:500000	納品日	:2001-02-01
商品名	株式会社サンプル (販売A)		
内容	前月請求金 株式会社サンプル (販売A)		
数量	:1	単価	:200
		税抜価格	:200
		消費税	:16
		小計	:216
受注番号	:100054321-01000	納品日	:2001-02-01
商品名	株式会社サンプル (販売A)		
内容	機室コンポーネント		
数量	:1	単価	:46,490
		税抜価格	:46,490
		消費税	:3,719
		小計	:50,209
受注番号	:100024680-02000	納品日	:2001-02-01
商品名	株式会社サンプル (販売B)		
内容	システムメンテナンス		
数量	:1	単価	:84,248
		税抜価格	:84,248
		消費税	:6,739
		小計	:90,987

Figure 3: Records and fields illustration.

An overview of the processing steps (depicted in Figure 2) is as follows. A scanned invoice is uploaded to the system which will be processed through an OCR system. The OCR result is presented to the user including the bounding box of each word. The user will proceed with selecting the bounding box of each word; forming fields of interest called *reference fields* denoted as $\{R_i\}_{i=1}^n$ where i is the field identifier (e.g. “item-description”, “item-quantity”, etc.). The set of fields will in turn form a *reference record* denoted as $\{G_y\}_{y=1}^m$, where m is the number of remaining records to be found in the document. The red boxes in Figure 3 are the reference fields and the remaining coloured boxes are the target fields.

The system will first generate a set of field candidates $C_i = \{c_{i1}, c_{i2}, \dots, c_{ix}\}$ for each reference field R_i (details in 3.1). We train a binary classifier to evaluate the probability of each candidate in C_i matching the R_i (discussed in 3.2). We train another binary classifier to evaluate the probabil-

Figure 4: Illustrations of field candidates generations of a particular reference field (in red). (a) highlighted area in purple is the horizontal overlaps region. Dashed-rectangles indicate all the base candidates of the reference field. (b) blue lines depict inter-word horizontal-gap and a zoomed-view showing the gaps of contiguous words. (c) highlighted areas in orange are the horizontal-expansion range of each base candidates and the orange dotted-rectangle is a sample of a horizontally-expanded candidate. (d) green boxes are the 5 additional candidates generated through multi-line expansion of the candidate field 「スチールキャビネット 4W105W-FXP13」.

ity of a candidate pair $c_i, c_j \in \{C_i \times C_j\}$ matches the reference pair R_i, R_j . Finally, we will construct a set records G' based on intersecting matching candidate pairs (i.e. $G' = \{c_i, c_j\} \cap \{c_j, c_k\} \cap \dots \cap \{c_{n-1}, c_n\}$).

3.1 Field Candidates Generation

3.1.1 Base Field Candidates (C^b)

We use the basic assumption that a target field will always have horizontal overlaps with the reference field even if it is in a non-linear structure. Therefore, we first select all the tokens (extracted by OCR) that have horizontal overlaps with the reference field as the base candidates (denoted as $\{C_i^b\}_{i=1}^n$). These candidates can be a single word or contiguous words on a single line that falls within the horizontal-overlap area of reference field (e.g. purple area in Figure 4(a)).

3.1.2 Horizontally Expanded Candidates (C^h)

The base candidate selection might not cover the entire phrase within a single line due to the OCR results being in individual words. We need to expand the candidates to the neighbouring words within a single line, the expanded candidates are denoted as $\{C_i^h\}_{i=1}^n$.

We first recursively expand the base candidates horizontally based on the text size and neighbouring information. We refer the horizontal distance of the closest edge of two tokens as *gap* (illustrated with blue lines in Figure 4(b)). The breaking condition is **either** of the following: 1) the horizontal-gap to the next word is more than half of the character height; or 2) the gap to the next word is larger or equal than the gap of the reference field to its neighbour (dark-red line in Figure 4(b)); or 3) the next word has horizontal overlaps the neighbour of the reference field (e.g. last entry in Figure 4(c), the next word after “...D650×H700” is “1”, however “1” has horizontal overlaps with “6”; the next word of reference field).

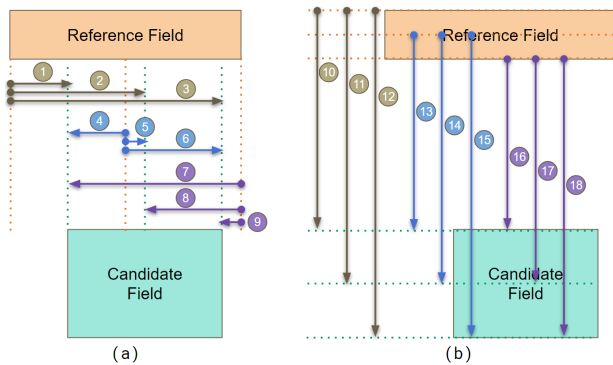


Figure 5: All edges spatial difference. (a) describes horizontal differences between R_i and C_i [$\{l, c, r\}$ in Eq. 1] and (b) describes the vertical differences [$\{t, m, b\}$ in Eq.1]

3.1.3 Multi-line Expanded Candidates (C^m)

There are some fields that can span multiple lines such as item descriptions. For such fields, we also expand the candidates downwards using the C^h . We simply sort C^h based on its y position in ascending order. We then construct the multi-line candidates (denoted as C^m) by merging each subsequent lines together up to (number of line in R) + 3 lines. The limit of 3 additional-lines is decided based on the observation of maximum line variation of a field type in our dataset, it also limits the number of candidates to be evaluated in the next step. Figure 4(d) shows an example of line expansion of a single candidate.

3.2 Evaluating Candidates

Once we obtain a set of field candidates C_i^m that could potentially match reference field R_i , we can evaluate their matching probability to obtain a subset of C_i^m that will be the *target field*. We evaluate the candidate-reference pair based on the features computed from the visual difference of each candidate field against the reference field (referred as *single-field features*, described in section 3.2.1). On top of that, we also added a re-evaluation algorithm to recovers some false negatives prediction (discussed in 3.2.3). The use of additional features and the re-evaluation algorithm are the key points that serve to improve the previous work [1].

Note that we only need to evaluate the candidates in C^m since it also contains the C^b and C^h candidates. For the sake of simplicity, we refer C^m as C from this point onwards.

3.2.1 Single-Field Features

a. Field-to-Field Features

We crafted a set of features based on the coordinates difference between two fields. We included the center- x and middle- y of the boxes which will handle center aligned fields. Instead of just computing basic edge-to-edge differences (distance of left edge of C_i against left edge of R_i and so on), we compute all edge combinations differences. Figure 5 illustrates the edge-to-edge relationship.

* l, c, r refers to left- x , center- x , right- x

* t, m, b refers to top- y , middle- y , bottom- y

* $E_e^{C_i}$ refers to e -edge of C_i ; $e \in \{l, c, r, t, m, b\}$

$$\left| E_x^{C_i} - E_y^{R_i} \right|; \quad (1)$$

9 feats.: $(x, y) \in \{l, c, r\}$ & 9 feats.: $(x, y) \in \{t, m, b\}$

Including the existence of vertical & horizontal overlaps, there will be 20 field-to-field features.

b. Neighbours-to-Neighbours Features

In [1], the authors compute the separation of C_i (against its nearest neighbour) to determine its perceptual-saliency. However, there is a possibility that the R_i itself is not perceptually salient. Instead, we compute the difference in the separation between C_i and its neighbour $N_d^{C_i}$ against the separation between R_i and its neighbour $N_d^{R_i}$; where $d \in \{left, top, right, bottom\}$.

Similar to Field-to-Field features, we compute the separation using all edges combinations. This not only gives us the saliency of a field, but we also obtain information regarding the alignment of a field with respects to its neighbours. This also serves as good features in verifying multi-line candidates (using the separation difference of bottom neighbour). We compute such features using the nearest neighbour in 4 directions which gives us 4×18 features:

$$\left| \left| E_x^{N_d^{C_i}} - E_y^{C_i} \right| - \left| E_x^{N_d^{R_i}} - E_y^{R_i} \right| \right|; \quad (2)$$

9 feats.: $(x, y) \in \{l, c, r\}$ & 9 feats.: $(x, y) \in \{t, m, b\}$;
 $d \in \{left, top, right, bottom\}$

3.2.2 Candidate Evaluation

The features in a. and b. are combined and denoted as $f^s(C_i, R_i)$. We then train a single-field model M^s (details in 3.4) to obtain a probability of C_i matching R_i given $f^s(C_i, R_i)$. Given a threshold θ , we regard

$$\begin{aligned} & \text{Positive prediction } C_i^+; \text{ if } M^s(f^s(C_i, R_i)) \geq \theta \\ & \text{Negative prediction } C_i^-; \text{ otherwise} \end{aligned} \quad (3)$$

3.2.3 Missed-Candidate Re-evaluation

A false negative in single-field evaluation has large impact on the overall result as the double-field evaluation will be performed only on candidates that are evaluated as positive by single-field evaluation (for optimal performance). As there are bound to be some false negatives from M^s using $f^s(C_i, R_i)$, we iteratively re-evaluate the negative-predictions C_i^- with the positive predictions C_i^+ until there is no new C_i^+ or no C_i^- left. This allows us to recover false-negative cases C_i^- that have a low probability of matching R_i but high probability of matching other candidate fields C_i^+ .

3.3 Target Record Predictions

From the previous step in section 3.2, we manage to obtain some set C_i^+ that matches the reference fields R_i . With the recovery of some false-negatives candidates in 3.2.3, the next step is to eliminate false-positive cases. We do so by validating the relationship between a pair of candidate fields C_i^+, C_j^+ to the relation of reference fields pair R_i, R_j . The relationship validation is performed through binary classification (using model M^d) using a feature set $f^d((C_i^+, C_j^+), (R_i, R_j))$ described in 3.3.1. A candidate pair is said to match a reference pair when the probability of M^d is greater than some threshold γ .

The pair-relationship validation will produce multiple candidate pairs that correspond to some reference field pairs. We will combine some intersecting pairs to generate record predictions G' . Two pairs of candidate fields are said to intersect when they share one common field candidate. For example, the pair (c_i^+, c_j^+) intersects (c_j^+, c_k^+) as they share the common field candidate c_j^+ . A predicted-record $g' = \{c_1^+, c_2^+, \dots, c_n^+\}$ is generated by finding all intersecting candidates across the fields.

3.3.1 Double-Field Features

We use the difference between the edges distance of C_i to C_j and R_i to R_j . In [1], only the distance of the same edge (i.e. left-edge of C_i to left-edge of C_j) is computed. As mentioned in b., we can obtain better information based on inter-field alignment from all edges distance (i.e. left-edge of C_i to center- x of C_j , etc.). Therefore, we formulate 18 features representing such edge differences:

$$f^d((C_i, C_j), (R_i, R_j)) = \left| \left| E_x^{C_i} - E_y^{C_j} \right| - \left| E_x^{R_i} - E_y^{R_j} \right| \right|; \quad (4)$$

9 features: $(x, y) \in \{l, c, r\}$ & 9 features: $(x, y) \in \{t, m, b\}$

Table 1: Comparison with previous work (in %) against our method in stages. *Same-edge* refers to using the difference of the same edge (i.e. left-to-left, right-to-right, etc.) only. *All-edges* uses edges combination as depicted in Figure 5. *Single-Field (SF)*: refer to 3.2.1 and *Re-evaluation*: refer to 3.2.3

Features	Method	Prec.	Rec.	F1
Same-edge (as used in [1])	SF Only	86.34	68.98	79.69
	SF+Re-eval	81.14	91.81	86.15
	Full Method	94.81	90.57	92.64
All-edges (Our method)	SF Only	90.95	89.83	90.38
	SF+Re-eval	86.56	94.29	90.26
	Full Method	96.90	93.05	94.94

3.4 Training

All the features described in 3.2 are normalized against the width or height of the page (i.e. all features are in the form of 0-100% of the page size).

The training inputs are constructed from document images with annotated records, each having at least 2 records. Instead of just using the first record against other records, we construct the training pairs from all combinations of records. This allows us to not only learn the relationship of the first record (i.e. user input) against other records but also the inter-record relationship which is useful for re-evaluation as discussed in section 3.2.3.

In [1], authors train a Bayesian model to evaluate a set of candidate fields as a whole by defining a custom log-likelihood. In this paper, we focus on algorithm improvement and features engineering; to provide enough information to learn about the structure of the fields. We leave the model selection to an automated process.

We adopt Automated Machine Learning (AutoML), specifically using the Tree-based Pipeline Optimization Tool (TPOT⁴)[12]. TPOT uses genetic programming (GP) to select the best combination of features, model and parameters. We train two pipelines through TPOT; one as the field-to-field model M^s and another for the field-pair model M^p .

4 Experiments

4.1 Dataset

We use real-world Japanese invoice images obtained from Works Applications Co., Ltd.⁵ The training set consists of 18 invoice images with various layout structure including both linear & non-linear. Each invoice contains 2 to 11 records totalling up to 92 records. Each record consists of 2 to 6 fields. Note that the number of fields across the records in a single document might vary due to some field being optional (i.e. not present in some records). With the training set construction described in 3.4, we constructed 1245 positive and 33368 negative samples from the 18 invoices.

We prepare a separate dataset of 27 invoices for evaluation purposes. All these invoices are from different vendors, hence have a great variation of layout. And none of the layouts exists in the training dataset. There are 2-23 records per invoice; 128 records in total. There are 2 to 6 fields in a single record.

For evaluation purpose, we simply use the first record on each document to represent user input and attempt to detect the other records; simulating the real-world user operation. This gives a split of 27 input records and 101 records to be tested.

4.2 Results

We evaluated our approach in stages from Single-Field evaluation (SF Only), Missed-Candidate Re-evaluation (SF + Re-eval) and the Double-Field evaluation (Full Method).

As observed in Table 1, introducing more edges relation can increase the performance. We also observed that recovering the false negative cases (improving Recall) through candidates re-evaluation will affect the precision (introducing some false positives). However, after running the double-field evaluation

(full method) the precision can be improved while maintaining the recall, hence achieving better overall performance.

From the end-user point of view, we manage to retrieve 96 out of the 101 records (95%). 14 of the 96 retrieved records are partially retrieved record (missing 1-2 fields).

5 Discussions

We have demonstrated the usefulness of using more extensive visual features in improving the accuracy of tabular extraction task. The proposed algorithm also solves the problem in handling missing/optional fields as faced by [1]. Partial records only require the user to fix 1-2 cells while not retrieving the record (problem in [1]) would require the user to manually fix the entire row.

Unfortunately, the candidates re-evaluation is unable to fully recover all candidates in some cases. [6] presented a missing-field recovery method that uses word gaps and area on English invoices. In future, we would like to incorporate a similar recovery method but with less reliance on word gaps as there is no word spacing in the Japanese language.

We would also like to explore the feasibility of incorporating textual features as implemented in [6], but specific to Japanese language context. Few possible ways in learning the Japanese textual features include usage of Sudachi [18] and/or various Japanese word vectors such as nwc2vec [16].

References

- [1] E. Bart, P. Sarkar. Information extraction by finding repeated structure. In *Proc. of the 9th IAPR International Workshop on Document Analysis Systems, DAS '10*, 2010.
- [2] A. C. e Silva, A. M. Jorge, L. Torgo. Design of an end-to-end method to extract information from tables. *IJDAR*, 8, 2006.
- [3] D. W. Embley *et al.* Table-processing paradigms: a research survey. *IJDAR*, 8, 2006.
- [4] M. Fan, D. S. Kim. Table region detection on large-scale PDF files without labeled data. *CoRR*, 2015.
- [5] M. Göbel *et al.* Icdar 2013 table competition. In *2013 12th ICDAR*, 2013.
- [6] T. Kasar, T. K. Bhowmik, A. Belaïd. Table information extraction and structure recognition using query patterns. In *2015 13th (ICDAR)*, 2015.
- [7] A. Katsuta *et al.* Information extraction from english & japanese resume with neural sequence labelling methods. In *In Proc. of the 24th Annual Meeting of the Association for Natural Language Processing*, 2018.
- [8] A. R. Katti *et al.* Chargrid: Towards understanding 2d documents. In *Proc. of the 2018 Conference on EMNLP*, 2018.
- [9] S. Khusro, A. Latif, I. Ullah. On methods and tools of table detection, extraction and annotation in pdf documents. *J. Inf. Sci.*, 2015.
- [10] B. Klein, S. Agne, A. Dengel. Results of a study on invoice-reading systems in germany. In *Document Analysis Systems VI*, 2004.
- [11] T. T. N. Le *et al.* Extracting indices from japanese legal documents. *Artificial Intelligence and Law*, 23, 2015.
- [12] R. S. Olson *et al.* Evaluation of a tree-based pipeline optimization tool for automating data science. In *Proc. of GECCO 2016*, 2016.
- [13] Y. Qian *et al.* GraphIE: A graph-based framework for information extraction. *CoRR*, 2018.
- [14] R. Rastan, H.-Y. Paik, J. Shepherd. Texus: A task-based approach for table extraction and understanding. In *Proc. of the 2015 ACM Symposium on Document Engineering*, 2015.
- [15] A. Shigarov *et al.* TabbyPDF: Web-based system for PDF table extraction. In *Information and Software Technologies*, 2018.
- [16] H. Shinnou *et al.* nwc2vec: Word embedding data constructed from NINJAL web japanese corpus. *Journal of Natural Language Processing*, 2017.
- [17] K. Sudo, S. Sekine, R. Grishman. Automatic pattern acquisition for japanese information extraction. In *Proc. of the 1st International Conference on Human Language Technology Research, HLT '01*, 2001.
- [18] K. Takaoka *et al.* Sudachi: a japanese tokenizer for business. In *Proc. of the 11th LREC*, 2018.
- [19] V. Voutilainen, T. Pentto. Electronic invoice processing as a tool for cost reduction. *Frontiers of e-Business Research*, 2003.
- [20] R. Zanibbi, D. Blostein, R. Cordy. A survey of table recognition: Models, observations, transformations, and inferences. *Document Analysis and Recognition*, 7, 2004.

⁴<https://github.com/EpistasisLab/tpot>

⁵<https://www.worksap.co.jp>