Recurrent Neural Network Approach for Table Field Extraction in Business Documents

Clément Sage, Alex Aussem, Haytham Elghazel, Véronique Eglin, Jérémy Espinas

Recurrent Neural Network Approach for Table Field Extraction in Business Documents

Clément Sage*†, Alexandre Aussem*, Haytham Elghazel*, Véronique Eglin* and Jérémy Espinas†
*Univ Lyon, CNRS, LIRIS UMR 5205, F-69100, VILLEURBANNE, France
{clement.sage, alexandre.aussem, haytham.elghazel, veronique.eglin}@liris.cnrs.fr
†Esker, F-69100, VILLEURBANNE, France
{clement.sage, jeremy.espinas}@esker.fr

***Abstract*—Efficiently extracting information from documents issued by their partners is crucial for companies that face huge daily document flows. Particularly, tables contain most valuable information of business documents. However, their contents are challenging to automatically parse as tables from industrial contexts may have complex and ambiguous physical structure. Bypassing their structure recognition, we propose a generic method for end-to-end table field extraction that starts with the sequence of document tokens segmented by an OCR engine and directly tags each token with one of the possible field types. Similar to the state-of-the-art methods for non-tabular field extraction, our approach resorts to a token level recurrent neural network combining spatial and textual features. We empirically assess the effectiveness of recurrent connections for our task by comparing our method with a baseline feedforward network having local context knowledge added to its inputs. We train and evaluate both approaches on a dataset of 28,570 purchase orders to retrieve the ID numbers and quantities of the ordered products. Our method outperforms the baseline with micro F1 score on unknown document layouts of 0.821 compared to 0.764.**

***Index Terms*—Table Field Extraction, Information Extraction, Document Analysis, Named Entity Recognition, NER, Recurrent Neural Networks, RNN, Business Documents, Purchase Orders**

I. Introduction

Business documents, whose main exchanged classes are invoices and purchase orders, contain valuable information that companies want to retrieve for further processing such as integration in their Enterprise Resource Planning (ERP) system and structured archiving. Even if Electronic Data Interchange (EDI) of documents exists and is progressively spreading, a large part of daily issued business documents is still printed on paper or generated in digital format such as PDF, thus requiring an information extraction step. If performed manually, this additional task is time-consuming for employees in charge of document processing, especially in companies dealing every day with potentially thousands of documents coming from hundreds of issuers. Yet, automating information extraction from business documents is challenging due to the semi-structured nature of these documents [1], i.e. the fact that an instance of a specified document class mandatorily contains a finite and predefined set of information to retrieve (e.g. issuer's name and address, total amount) but the positioning and textual representation of the information are unconstrained. Indeed, every document issuer is free to generate business documents with a specific layout (i.e. arrangement of elements) and change it when desired.

In this work, we focus on the extraction of field instances in item tables found in invoices and purchase orders - see Fig. 1 for an illustration of the task - but our approach is generic enough to adapt to other tabular fields with minimal efforts. Compared to non-tabular fields, these fields are more difficult to deal with since they are subject to higher position and textual content variations even between documents with the same layout. This is due to the variable number of field instances to extract and the unstructured nature of certain table data such as item descriptions.

One could think that this problem might be solved by detecting the appropriate tables, then retrieving their physical structure and finally classifying their rows and columns in order to extract the table fields. However, as pointed out by [2], the physical structure of tables from industrial contexts may be ambiguous in case of overlapping columns and may not provide enough knowledge for further field extraction, e.g. relevant information may be scattered throughout multi-line rows and inter-stratified with irrelevant data such as in the leftmost column of the item table in Fig. 1. Therefore we cannot rely on it to extract the logical structure of tables in business documents. Bypassing physical structure recognition, we propose an end-to-end method for table field extraction taking as input the document tokens retrieved by an Optical Character Recognition (OCR) system and directly tagging each of them with their field type. Inspired by the method described in [3] for non-tabular field extraction in invoices, our approach resorts to a token level recurrent neural network based solution that combines textual and spatial features. Starting with OCR results, our approach has the advantage of working both with scanned and born-digital documents.

This paper is organized as follows. In section II, we review related work. Section III presents our approach in detail. The dataset used for experiments is depicted in section IV while the baseline method and experiments are described in section V. Results are discussed in section VI. Finally, section VII concludes and gives some perspectives for future work.

II. Related Work

Most recent information extraction systems build on knowledge coming from a database of documents already processed

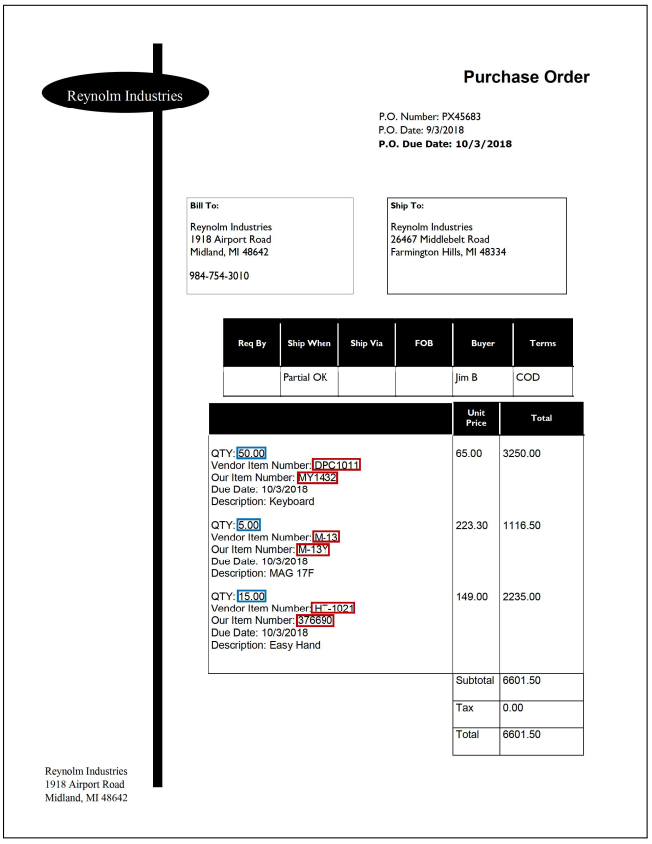

Fig. 1. Example of a purchase order from which we want to extract the ID numbers and quantities of the ordered products. Their corresponding token level labels are respectively highlighted by red and blue rectangles. Note that distinct table field types may be displayed in the same physical column.

by the system. We can cite the works of Intellix [4] and of Rusinol, Benkhelfallah and Poulain d'Andecy [5]. These incremental approaches are based on the document layouts, also called templates. When extracting information of an incoming document, the system first detects if its template has already been processed. To do so, [4] uses k-nearest neighbor (kNN) model with low resolution word position features while [5] considers this step outside of the scope of work. Then, rather than applying a fixed spatial mask which cannot extract floating fields, these approaches rely on local contexts of the targeted fields by comparing spatial relationships between pairs of words in the incoming document either with the matching template documents [4] or a template model, e.g. star graphs [5]. After verification, modification and completion of the extracted fields by the end user, the processed document is added to the database or incrementally refines the template model.

After having processed a few documents per layout, these systems reach good performances in extracting the main non-tabular fields with micro F1 scores of about 0.9. However, as shown by the authors of Intellix in an later paper [6], they naturally suffer from cold start performances when dealing with documents from an unknown or insufficiently known layout. Indeed, they reported F1 scores of 0.22 and 0.78 for respectively zero and one training document with the same template, which is problematic in industrial contexts with continuously new business partnerships. Even if this issue could be mitigated by sharing the template document databases (e.g. across companies in [7]) or by approaches enhanced by *a priori* models (e.g. hybrid approach proposed by [8]), incremental approaches rapidly become intractable when facing thousands or more distinct layouts. Moreover, while the end user of these systems can define sets of fields to retrieve which are template specific, we assume in this work that we extract a single fixed set of fields for a given document class. Finally, none of these approaches deal with tabular fields which are far more difficult to extract even with extraction knowledge for the considered template.

Template-free approaches for field extraction in business documents have also been proposed in the literature. Belaïd and Belaïd [9] resort to a bottom up morphological tagging approach for detecting invoice items and extracting their different fields. They achieve excellent field recognition rates on invoices but their approach is limited to regular table structures and not easily adaptable to other languages than French, the one for which the system was designed for. Zhu, Bethea and Krishna [10] specifically address the extraction of transaction information in image receipts by using a linear discriminative Conditional Random Field (CRF) on the list of homogeneous regions segmented by an OCR engine. The system relies on engineered features: rich layout features (font, bold, text alignment...) and language content features (capitalization, digit frequency, presence of special characters or patterns...). Tests carried out on two datasets show that extraction performances are significantly lower for field types with important content variability such as vendor names (average F1 score of 0.69), than more structured field types like dates, credit card or phone numbers with respective average F1 scores of 0.89, 0.83 and 0.80. Later, smartFix conceived by Deckert, Seidler, Ebbecke and Gillmann [2] tackles the table content understanding problem in semi-structured documents by an expectation-driven approach. To that end, a global quality measure over the set of possible column configurations that involves local and global expectations is optimized by exhaustive search after application of heuristics eliminating a large part of clearly irrelevant configurations. Evaluated on orders, invoices and medical documents, the system reaches satisfying cell extraction rates.

However, all of these template-free approaches rely on significant domain specific knowledge and a number of hand-crafted features or rules, showing troubles when tackling highly unstructured fields in documents with strong layout variability. In our work, a model able to implicitly learn complex patterns is required to reach satisfying extraction performances at minimal human configuration cost.

Our information extraction task is also closely related to the Named Entity Recognition (NER) problem from the Natural Language Processing (NLP) community. This latter task usually refers to the extraction of instances of generic classes, such as persons or organizations, from natural text organized

in sentences. For this task, neural architectures currently constitute the state-of-the art methods to implicitly learn the grammatical structure of sentences in order to classify the individual words with their appropriate type. See [11] for an up-to-date survey about methods employed in NER, highlighting the effectiveness of feature-inferring Recurrent Neural Network (RNN) systems over earlier approaches such as knowledge-based and feature-engineered supervised systems. So, Palm, Winther and Lawsthey [3] propose an RNN iterating over tokens segmented by an OCR engine for extraction of non-tabular fields from invoices. Adapting RNN for information extraction in business documents is not straightforward as this document type implies several specificities. Compared to documents used in NER tasks which are organized in sentences, business documents do not have a natural reading order as both vertical and horizontal axes encode information of different nature and the spacings and alignments of their tokens carry important semantic knowledge. Considering this, in addition to textual based features, they design numerous spatial features for each token describing their normalized positions in the page, alignments and proximity with neighbors. With some post-processing applied on the token level RNN predictions, they report excellent extraction results on a large dataset of invoices. They experimentally show that the use of an RNN is justified in information extraction tasks by leading to greater performances over logistic regression with spatially limited context and feedforward neural networks, especially for templates unseen during model training.

Leveraging these findings, we adopt a recurrent neural network approach for solving the less tackled task of tabular field extraction within business documents.

III. Approach

We now describe our approach for extracting table field instances of an incoming document. It involves 4 main steps which are depicted in Fig. 2.

A. Text Extractor

First of all, the document text is retrieved, resulting in a list of the document tokens with their textual value, page index and horizontal and vertical coordinates of their bounding boxes. If the document does not contain embedded text, an external commercial OCR engine is used. Hence, OCR optimizations are out of our scope of work.

B. Vocabulary Indexer

Then, we assign a vocabulary index for each token of the document based on its textual value. Instead of determining the index directly from its raw text returned by the OCR, we first transform the text of the token. Indeed, text in business documents follows a specific grammar characterized by a limited number of structuring keywords - possibly abbreviated - which are common across documents and a huge amount of rare or unique tokens specific to an issuer or even a document instance, e.g. occurrences of document number, dates and amounts. Therefore, to avoid having a lot of tokens with the out-of-vocabulary index for documents not part of the training set and thus losing essential knowledge from textual content, we design a small set of text categories specific to business documents. If the raw text of a token matches the regular expression corresponding to a category, the index of this category is assigned to the token's vocabulary index. As it may match multiple categories, we keep only the index of the first matched category. The ordered list of the rather explicit categories is the following: *DigitSequence*, *ContainsDigitAndAlpha*, *ContainsDigitAndDash*, *ContainsDigitAndSlash*, *ContainsDigitCommaAndPeriod*, *ContainsDigitAndComma*, *ContainsDigitAndPeriod*, *PunctuationSequence*, *URL*, *EmailAddress*. The categories involving digits are rather generic, letting the further classifier the task to distinguish what these categories may refer to, e.g. integers, floats, dates, phone numbers, depending on the document language and culture. Besides, these categories have also the advantage of making the model more robust to OCR failures, as recognition errors on some characters of a token are likely to have little impact on the category matching.

Tokens which haven't matched any of the previous categories are standardized as follows. Raw textual values of these tokens are converted to lower case and, if any, punctuation and whitespace characters are removed at the beginning and the end of the text. For example, this results in "Total:" and "TOTAL" tokens having the same standardized textual representation. If the standardized textual value of a token is in the vocabulary, its corresponding index is assigned to the token's vocabulary index. Otherwise, we assign the index corresponding to the out-of-vocabulary element.

The vocabulary is determined by enumerating the unique standardized textual values from all the tokens of the training documents. This vocabulary is then restricted by removing its values that rarely appear in the training set since they are not helpful for later processing of unknown layouts or documents and make the extraction model more likely to overfit. To this end, we list the vocabulary values by decreasing frequency of occurrence in the training set and we retain only the most frequent values that, put together, match a minimum percentage of the total number of occurrences.

C. Feature Calculator

For each token in the document, we constitute a rich feature vector combining textual and spatial components. The textual part includes the dense continuous embedding vector associated to its previously determined vocabulary index, case features (percentage of its characters in upper case and binary factor indicating if it has a title form or not) and the number of characters composing the raw text of the token. Textual embeddings are learned jointly with the rest of the extraction model. Some authors [12], [13] reported NER performance gains by using embeddings pre-trained on a large unlabeled corpus. However, in our case, pre-training them by adapting Skip-gram model proposed by Mikolov, Sutskever, Chen, Corrado and Dean [14] did not result in an increase of the field extraction performances. This may be explained by our

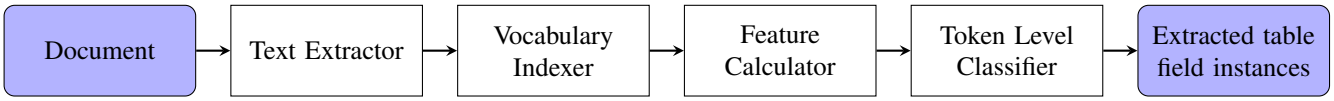

Fig. 2. Schematic view of our approach for table field extraction.

vocabulary indexer delivering a relatively small vocabulary size, thus facilitating the learning of the embeddings. So, we randomly initialize embedding vectors and learn them from scratch.

The spatial part encompasses the coordinates of the top-left and bottom-right corners of the token's bounding box, normalized by the width of the page containing the token. We do not create more spatial features for describing alignments and relative distances with its neighboring tokens. We instead rely on the memorization abilities of the RNN to infer them from the individual spatial positions.

D. Token Level Classifier

Tokens of the document are organized as a unidimensional sequence by reading them in Z-order, i.e. starting with the top left token of the document and finishing with the bottom right token. In case of a multi-page document, the sequences of each page are concatenated based on the order of appearance of the pages.

The token sequence of the document is then fed to a network composed of several stacked bidirectional LSTM (BLSTM) [15] layers followed by a dense output layer having $k + 1$ softmax units, k being the number of field types to retrieve. One of these units is dedicated to tokens carrying information that we do not want to extract. Class prediction for each token directly corresponds to the highest softmax unit value without performing any post-processing operations such as checking that its data type is consistent with the predicted field type or corrections based on external databases containing *a priori* knowledge.

IV. Dataset

We train our extraction model and evaluate its performances on a dataset of real world business documents.[1] The dataset consists of 28,570 purchase orders originating from 2,818 issuers. We assume that each issuer generates a distinct document template. Each template have between 3 and 31 different document instances. Although the dataset is monolingual with English as language of each document, the dataset is multi-cultural with European and American documents, thus impacting the format of key document elements such as dates and amounts.

From these documents, we want to extract 2 types of fields contained in the table of the ordered products: ID number and quantity. These two fields are sufficient for performing the complete extraction of the ordered products since all the other product information such as description and unit price can be inferred from them with an external product database of the document recipient. The token level labels have been determined by establishing correspondences between the list of tokens segmented by the OCR and the field values validated by the end users of a commercial document automation software. For limiting discrepancies between the validated data and the documents, we have selected documents whose field values all necessarily match the textual value and position of a single different OCR token. 0.78% and 0.72% of the dataset's tokens are respectively labeled as ID number and quantity corresponding to an average of 3.7 and 3.5 instances per document.

[1]Unfortunately, we are not allowed to release the dataset due to privacy restrictions.

V. Experiments

To gauge the usefulness of recurrent connections for our task, we compare our model with a baseline method composed of a feedforward neural network having the same number of layers and learnable parameters. This neural architecture leading to independent predictions across tokens of the same document, we introduce some context knowledge from neighboring tokens for fair comparison. To do so, we modify the input feature vector of each token by concatenating its original feature representation with the feature vectors of its closest tokens in the left, right, top and bottom directions. One token is kept in each direction, resulting in a feature vector size five times larger than for the RNN model. Hyper-parameters common to both models have identical values.

We conduct experiments to assess the ability of these two extraction models to generalize to document templates not seen during training phase. To this end, similar to experiments designed by [3], we split the dataset according to two ways: *DocumentLevelSplitting* and *IssuerLevelSplitting*. The former way naively randomly separates the document instances in the dataset to constitute the training, validation and test sets. The latter data splitting is at issuer level: all the documents from the issuers in a set of the *IssuerLevelSplitting* experiment then constitute the documents of that set. This data separation ensures that there are no documents that share templates between these three sets. In order to accurately estimate model performances, for each data splitting way, we randomly partition the dataset into 5 folds and evaluate the model on each fold while using the 4 remaining folds for its training. Each ensemble of 4 folds is further independently divided in training and validation sets so that they respectively represent 70 % and 10 % of the whole dataset. We evaluate both models on the same data partitions produced by the two dataset splitting ways.

The model hyper-parameters are chosen based on the micro F1 score on the validation sets. The following hyper-parameter values are found to be satisfying for all the above experiments. We use 64 dimensional textual embeddings and document

mini batches of size 32. The sequence loss is minimized by the Adam optimizer [16] with default recommended settings except for the learning rate which is equal to 0.001 during the first 5 epochs and then exponentially decreases with a rate of $1/1.15$. We train each model instance until the micro F1 score has not improved on the validation set for 3 epochs, conducing to about 18 and 12 training epochs respectively for *DocumentLevelSplitting* and *IssuerLevelSplitting* experiments. The RNN model is composed of 2 bidirectional LSTM layers of 1,300 cells each with outputs of each direction being concatenated while the baseline FNN has two dense layers with 6,947 and 1,300 rectified linear units. All network weights are randomly initialized following a uniform distribution of amplitude 0.05. To deal with exploding gradients, we apply the gradient norm clipping strategy [17] with a clipping threshold equal to 5.

The percentage threshold for restricting the vocabulary to the most frequent transformed textual values of tokens across the training set is equal to 95 %. This threshold is chosen by examining the plot of the proportion of occurrences of these textual values that belong to the restricted vocabulary as a function of the vocabulary size, textual values in the vocabulary being sorted by descending frequency of occurence in the training set, as shown in Fig. 3 for the *DocumentLevelSplitting* experiment. With this filtering, we reduce the vocabulary size

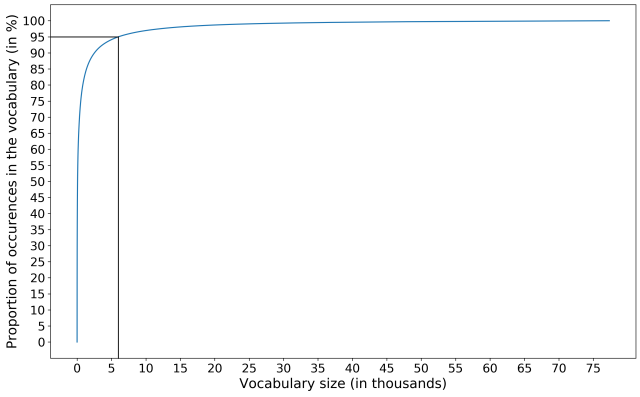

Fig. 3. Proportion of occurrences of the transformed textual values of training tokens that belong to the vocabulary as a function of the vocabulary size for the *DocumentLevelSplitting* experiment.

by a factor greater than 10, resulting in a vocabulary of only 6,010 transformed textual values.

Training and evaluation is performed on a single NVIDIA TITAN X GPU. This explains that, for computational reasons, we do not train the models on documents with more than 1800 tokens, which amounts to about 5 % of the training set being put aside for both data splitting ways. However, we evaluate the models on all documents in validation and test sets.

VI. Results

For each experiment, according to [18], we compute the recall, precision and F1 score for our 2 targeted fields as well as the corresponding micro averaged metrics obtained by each model instance on its test fold. We average each metric across the 5 folds of the experiment and we report the averaged results for the baseline and RNN approaches in Table I and Table II, respectively for *DocumentLevelSplitting* and *IssuerLevelSplitting* experiments. Bold font highlights the best performing model for a given evaluation metric and field.

TABLE I
Extraction Performances For DocumentLevelSplitting Experiment.

	F1 score		Precision		Recall	
Field	Baseline	RNN	Baseline	RNN	Baseline	RNN
ID number	0.853	**0.906**	0.863	**0.907**	0.844	**0.905**
Quantity	0.926	**0.964**	0.902	**0.955**	0.952	**0.974**
Micro avg.	0.889	**0.934**	0.882	**0.930**	0.896	**0.938**

TABLE II
Extraction Performances For IssuerLevelSplitting Experiment.

	F1 score		Precision		Recall	
Field	Baseline	RNN	Baseline	RNN	Baseline	RNN
ID number	0.685	**0.752**	0.689	**0.769**	0.687	**0.738**
Quantity	0.848	**0.894**	0.842	**0.902**	0.859	**0.888**
Micro avg.	0.764	**0.821**	0.763	**0.834**	0.769	**0.810**

For both data splitting ways and both fields, the RNN method substantially surpasses the baseline in terms of precision, recall and thus F1 score. Given the differences between the two methods, the increased performances of the RNN proves that the context knowledge which is modeled in its hidden states is more effective than the local context knowledge introduced in the feature vectors of the baseline for further extraction of table fields.

As expected, extraction performances are lower for the *IssuerLevelSplitting* experiment than for *DocumentLevelSplitting* with respective micro F1 scores of 0.821 and 0.934 for the RNN model. However, the difference of F1 score values is small compared to performances gaps that are usually observed for template-based incremental methods when retrieving information for unknown versus known templates [6]. That further advocates in favor of template-free approaches for extraction of table field instances.

We notice that there is a significant difference of performance between the two field types, especially for the *IssuerLevelSplitting* experiment with the RNN model having its ID number F1 score which is 0.142 lower than for the quantity. One reasonable explanation is the higher level of noise in the token level labels for the ID number field. Indeed, ID number instances may appear twice for a single item, reflecting the product references on the issuer and the recipient side. Labels being generated by document automation software users from many distinct companies, one ID number instance or another or both may be marked as ground truth by a particular user. This choice depends on the further integration of extracted data in their Enterprise Resource Planning (ERP) system. Moreover, for some documents, ID number instances

might not have a dedicated physical column, often appearing within the description column (e.g. Fig. 1) sometimes without keywords clearly introducing them, which makes their correct extraction without additional business context ambiguous. Confusing labels also exist for the quantity field with multiple instances for a single ordered product, e.g. number of boxes and number of total units within these boxes, but it is less frequently observed among our dataset.

The visualization of RNN model predictions for a representative subset of the validation documents gives us insights about the main difficulties encountered. Firstly, some designed text categories appear to be too broad since tokens that belong to categories that also often include instances of our targeted fields are more prone to misclassification errors. For example, item due dates are sometimes predicted by the RNN as ID number instances as they may share the *ContainsDigitAndDash* or *ContainsDigitAndSlash* categories. Secondly, RNN occasionally produces predictions that are not structured enough at document level. Indeed, its predictions within a single physical column may be inconsistent across the different rows. Besides, the numbers of ID number and quantity predictions may deviate considerably across a single document although these two fields appear at a comparable frequency, i.e. about the number of ordered products in the document.

VII. Conclusion

In this paper, we proposed a generic template-free approach for extracting table field instances in business documents whose layouts may not have been seen during training. We evaluated its effectiveness on a large dataset of real world purchase orders in order to retrieve the ID numbers and quantities of all their ordered products. Our recurrent neural network based method outperformed the feedforward network baseline enhanced with local context, with micro F1 scores of 0.821 and 0.764 on unknown document templates. Therefore, we showed that RNNs combining textual and spatial features of OCR tokens which constitute the state-of-the-art for extracting non-tabular fields are relevant for retrieving the more challenging table fields. We presented an application to a precise business use case but our approach can be easily adapted to extract tabular fields for other document types as our method relies on little domain specific textual preprocessing.

In the near future, we plan to provide a more granular text parsing by replacing the vocabulary index step by a character level module for generating the textual representations of the tokens. Besides expecting improved extraction results, this would have the advantage to remove all the operations related to the business domain from our approach. We also want to explore models for getting more structured predictions such as resorting to Conditional Random Fields (CRF) on top of the RNN. This technique was proved to be helpful for capturing correlations within sequences of labels for NER tasks [12]. More importantly, we intend to tackle end-to-end recognition of structured tabular entities and evaluate our proposed methods on item tables of business documents for extracting products with their respective ID number and quantity. Finally, we will assess the behavior of our extraction models when confronted with a multilingual dataset.

Acknowledgment

These researches were supported by Esker. We thank them for providing the dataset on which experiments were performed and for insightful discussions about this work.

References

[1] M. Cristani, A. Bertolaso, S. Scannapieco, and C. Tomazzoli, "Future paradigms of automated processing of business documents," *International Journal of Information Management*, vol. 40, pp. 67–75, 2018.

[2] F. Deckert, B. Seidler, M. Ebbecke, and M. Gillmann, "Table content understanding in smartfix," in *2011 International Conference on Document Analysis and Recognition*. IEEE, 2011, pp. 488–492.

[3] R. B. Palm, O. Winther, and F. Laws, "Cloudscan-a configuration-free invoice analysis system using recurrent neural networks," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2017, pp. 406–413.

[4] D. Schuster, K. Muthmann, D. Esser, A. Schill, M. Berger, C. Weidling, K. Aliyev, and A. Hofmeier, "Intellix–end-user trained information extraction for document archiving," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 2013, pp. 101–105.

[5] M. Rusinol, T. Benkhelfallah, and V. Poulain dAndecy, "Field extraction from administrative documents by incremental structural templates," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 2013, pp. 1100–1104.

[6] D. Esser, D. Schuster, K. Muthmann, and A. Schill, "Few-exemplar information extraction for business documents." in *ICEIS (1)*, 2014, pp. 293–298.

[7] D. Schuster, D. Esser, K. Muthmann, and A. Schill, "Modelspace-cooperative document information extraction in flexible hierarchies." in *ICEIS (1)*, 2015, pp. 321–329.

[8] V. P. d'Andecy, E. Hartmann, and M. Rusiñol, "Field extraction by hybrid incremental and a-priori structural templates," in *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*. IEEE, 2018, pp. 251–256.

[9] Y. Belaïd and A. Belaïd, "Morphological tagging approach in document analysis of invoices," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 1. IEEE, 2004, pp. 469–472.

[10] G. Zhu, T. J. Bethea, and V. Krishna, "Extracting relevant named entities for automated expense reimbursement," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007, pp. 1004–1012.

[11] V. Yadav and S. Bethard, "A survey on recent advances in named entity recognition from deep learning models," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 2145–2158.

[12] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," *arXiv preprint arXiv:1603.01354*, 2016.

[13] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.

[14] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[15] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[17] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International Conference on Machine Learning*, 2013, pp. 1310–1318.

[18] Z. C. Lipton, C. Elkan, and B. Naryanaswamy, "Optimal thresholding of classifiers to maximize f1 measure," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 225–239.