Continuous User Feedback Learning for Data Capture from Business Documents

Marcel Hanke[1], Klemens Muthmann[1], Daniel Schuster[1], Alexander Schill[1], Kamil Aliyev[2], Michael Berger[2]

[1] Computer Networks, Dept. of Computer Science, TU Dresden, Dresden, Germany
`marcel.hanke@mailbox.tu-dresden.de`
{`klemens.muthmann, daniel.schuster, alexander.schill`}`@tu-dresden.de`
[2] DocuWare AG, Germering, Germany
{`kamil.aliyev, michael.berger`}`@docuware.com`

Abstract. Automatically processing production documents requires document type detection as well as data capture to find appropriate index data from a post-OCR representation of the document. While current learning-based methods perform quite well due to many similar documents created with the same template, their machine learning models require intense training and are hard to update frequently. We provide a method for continuously incorporating user feedback in a layout-based extraction process taking care of both immediate learning as well as limiting the size of the model. The method is evaluated on a tagged corpus of more than 5,000 business documents. It allows not only continuous re-training of the model thus adapting it to new document templates, but also starting from scratch with an empty model requiring less than 10% of the corpus as training documents to reach an accuracy measure of more than 80%.

Keywords: User Feedback, Information Extraction, Document Classification, Document Archiving, Document Management

1 Introduction

Document type classification, data capture, and detection of document sets are the three classical tasks in production document processing [8]. Existing solutions for doctype classification and data capture already achieve quite good extraction rates to find the right document type (e.g., invoice) as well as typical index data such as sender and receiver, document date, document id, or total amount. But as they are based on machine learning algorithms like Naive Bayes or Support Vector Machines, these methods require intense training with several hundreds or at least dozens of training documents per template to reach their high precision. Updating these learning models with new training documents means a complete learning cycle and is thus hard to achieve in a production environment.

Especially if such automatic document indexing should be used by SOHO[3] users, there is a need to provide an easy to train extraction component, which incorporates feedback immediately as only few training examples are available for each document template. Thus we provide a method which works on instance based document type classification and template detection in combination with layout-based data capture relying on positional OCR[4] results as well as positional user feedback.

This method (including means to limit the size of the model) is the main contribution of this paper and described in Section 4. A thorough evaluation on a real world data set of more than 5,000 business documents has been carried out (Section 5) providing general accuracy measure values as well as showing different aspects like the influence of model size on the extraction quality.

Before this, we start by discussing related work (Section 2) and giving an overview of the solution (Section 3).

2 Related Work

Providing adaptable models for machine learning has already been widely researched. Support Vector Machines (SVM) are one of the preferred methods. Cauwenberghs et al. [2] and Jia et al. [6] demonstrate their classification variability. Incremental and decremental functionality for model entries are shown but without performance evaluation in [2]. Jia processes a derivative method for SVM kernel functions. After a certain time step a new kernel function is generated based on the previous one.

In contrast to Cauwenberghs et al. [2] and Jia et al. [6] we focus on a method that immediately adapts to faulty processed documents and is able to handle index data extraction in addition to document classification.

[7, 11] propose an active learning method focusing on rich feedback by feature adaptation. Their system is based on features marked by human experts to update the knowledge model of the learning algorithm. Our proposed method is only based on result corrections, which a typical user carries out in his usual workflow anyway. In addition active learning as applied by Raghavan et al. is no update strategy for models but an enhancement strategy for model generation. Similar issues arise with the extraction approach of Culotta et al. [3] where user provided constraints extend a conditional random field method for index field extraction. Our approach assumes a template-based document layout which is exploited for improved extractions.

Stumpf et al. [9, 10] evaluated user understanding of classification decisions and methods to adapt the model. Even though they say that users are able to adapt rules and keywords to correct classification results, we require a faster result evaluation. Hence we prefer result correction instead of feature labelling, which is not discussed in [9]. Stumpf exploits the obtained experimental results to implement two methods. The first method uses a constraint-based approach.

[3] Small Office Home Office
[4] Opitcal Character Recognition

Relevant feature weights are adapted by increasing and decreasing user decisions. This method shows no improvement for the evaluated scenarios. The second method uses a variant of co-training as presented by Blum et al [1]. Instead of using two different classifiers they use one generated from user feedback. Similar to active learning, this method is also restricted to the model generation phase and would need further enhancements to fit into the continuous model adaptation. A similar approach with the same confinement using Naive Bayes classifier and user feedback is given by Huang et. al. [5].

3 Overview

The classification and extraction process as shown in Fig. 1 consists of two main phases. At first a model is generated from training data. The training data provides the correct index fields for each document as well as their positions (bounding box) within the document as our method works layout-based. In the second phase, unkown documents are processed. During processing, each document is at first classified to its document type (e.g., invoice, reminder) before the remaining index fields are extracted based on coordinates from the k most similar documents. Processing is carried out by a k-nearest-neighbour (k-NN) algorithm based on the automatic document type classification and index field extraction from our earlier work [4].

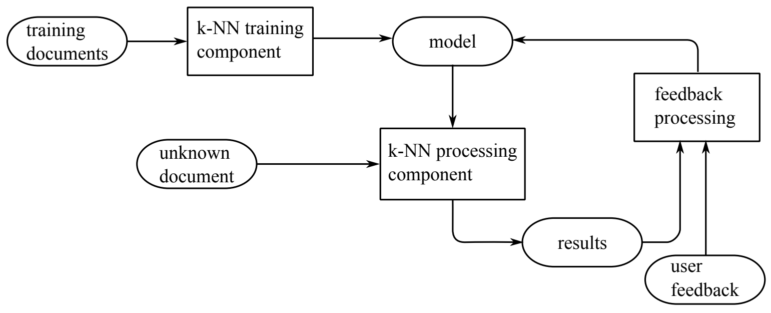
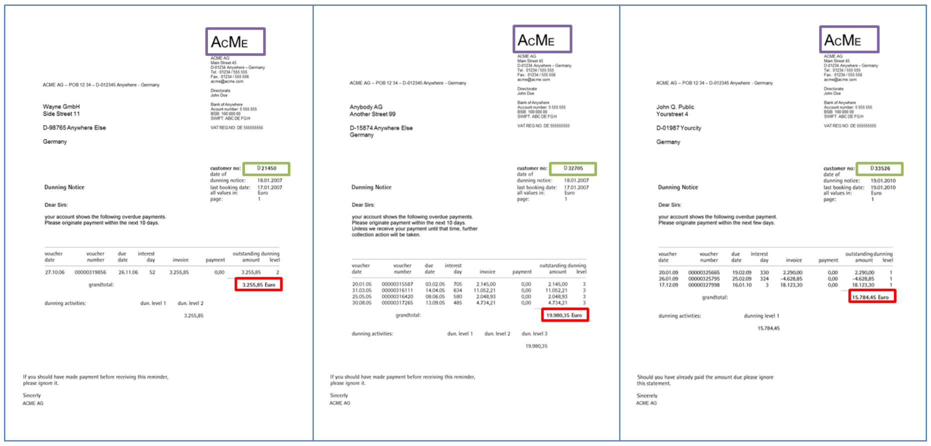


Fig. 1. System overview with main components and data processing.

Feature Extraction: Both algorithms (one for classification and one for extraction) consist of an inverted index of documents to generate a model. As shown in [4] we tried several features and feature combinations. In the end the following very simple features showed the best performance and are thus also used throughout the rest of this work. The classification model stores each document under all words from the document and classifies new documents by the type the k documents with the most similar content have. This is a simple word-based retrieval model as used by most search engines. The extraction adds the position of each word to the index and finds index fields based on the location of index fields in the k documents that share the most words at the same position. For this we sliced the document into boxes and appended the box a word starts in to the word in the index. If a document contains the word "invoice" in the upper left corner for example, it would be stored to the index under "invoice_0_0". The same is true for all other words in the document. That way a query to the index with a new document returns the k documents that have share many words at the same position and thus have a similar layout. Our layout based extraction approach then can apply the coordinates for each index field from the documents in the index to the new documents to extract new values.

Index Data Extraction: Fig. 2 shows an example of several documents using the same template with index fields on similar positions. Thus if the user provides feedback with the correct positions for the first two documents, we are able to find at least the fields sender name and document id as they are at the same position. Surely, this layout-based method does not work for fields with variable positions such as the total amount (as shown in the figure).


AcMe

ACME AG
Main Street 45
D-01234 Anywhere – Germany
Tel.: 01234 / 555 555
Fax.: 01234 / 555 556
acme@acme.com

Directorate
John Doe

Bank of Anywhere
Acount number: 5 555 555
BSB: 100 000 00
SWIFT: ABC DE FGH

VAT REG NO: DE 555555555

ACME AG – POB 12 34 – D-012345 Anywhere - Germany

Wayne GmbH
Side Street 11

D-98765 Anywhere Else

Germany

customer no:	D21450
date of dunning notice:	18.01.2007
last booking date:	17.01.2007
all values in:	Euro
page:	1

Dunning Notice

Dear Sirs:

your account shows the following overdue payments.
Please originate payment within the next 10 days.

voucher date	voucher number	due date	interest day	invoice	payment	outstanding amount	dunning level
27.10.06	00000319856	26.11.06	52	3.255,85	0,00	3.255,85	2
	grandtotal:					**3.255,85 Euro**	

dunning activities:	dun. level 1	dun. level 2
		3.255,85

If you should have made payment before receiving this reminder, please ignore it.

Sincerly
ACME AG

AcMe

ACME AG
Main Street 45
D-01234 Anywhere – Germany
Tel.: 01234 / 555 555
Fax.: 01234 / 555 556
acme@acme.com

Directorate
John Doe

Bank of Anywhere
Account number: 5 555 555
BSB: 100 000 00
SWIFT: ABC DE FGH

VAT REG NO: DE 555555555

ACME AG – POB 12 34 – D-012345 Anywhere - Germany

Anybody AG
Another Street 99

D-15874 Anywhere Else
Germany

customer no:	D32705
date of dunning notice:	18.01.2007
last booking date:	17.01.2007
all values in:	Euro
page:	1

Dunning Notice

Dear Sirs:

your account shows the following overdue payments.
Please originate payment within the next 10 days.
Unless we receive your payment until that time, further collection action will be taken.

voucher date	voucher number	due date	interest day	invoice	payment	outstanding amount	dunning level
20.01.05	00000315587	03.02.05	705	2.145,00	0,00	2.145,00	3
31.03.05	00000316111	14.04.05	634	11.052,21	0,00	11.052,21	3
25.05.05	00000316420	08.06.05	580	2.048,93	0,00	2.048,93	3
30.08.05	00000317265	13.09.05	485	4.734,21	0,00	4.734,21	3
	grandtotal:					**19.980,35 Euro**	

dunning activities:	dun. level 1	dun. level 2	dun. level 3
			19.980,35

If you should have made payment before receiving this reminder, please ignore it.

Sincerly
ACME AG

AcMe

ACME AG
Main Street 45
D-01234 Anywhere – Germany
Tel.: 01234 / 555 555
Fax.: 01234 / 555 556
acme@acme.com

Directorate
John Doe

Bank of Anywhere
Account number: 5 555 555
BSB: 100 000 00
SWIFT: ABC DE FGH

VAT REG NO: DE 555555555

ACME AG – POB 12 34 – D-012345 Anywhere - Germany

John Q. Public
Yourstreet 4

D-01987 Yourcity

Germany

customer no:	D33526
date of dunning notice:	19.01.2010
last booking date:	19.01.2010
all values in:	Euro
page:	1

Dunning Notice

Dear Sirs:

your account shows the following overdue payment.
Please originate payment within the next few days.

voucher date	voucher number	due date	interest day	invoice	payment	outstanding amount	dunning level
20.01.09	00000325665	19.02.09	330	2.290,00	0,00	2.290,00	1
26.01.09	00000325795	25.02.09	324	-4.628,85	0,00	-4.628,85	1
17.12.09	00000327998	16.01.10	3	18.123,30	0,00	18.123,30	1
	grandtotal:					**15.784,45 Euro**	

dunning activities:	dunning level 1
	15.784,45

Should you have already paid the amount due please ignore this statement.

Sincerely
ACME AG

Fig. 2. Business documents with the same template.

Real World Scenario: The algorithms described in this paper are used by a mid-sized german company in their document archiving solution. Users access this solution through a web interface. Over this web interface they are able to archive their documents and run automatic classification and indexing. Each user has an inbox of new documents either scanned or loaded directly from hard disk. Index values are assigned to each document with a certain confidence value between 0% and 100% based on the score of the retrieved similar documents from the index. The user can correct these values, thus providing feedback, in two different ways. He can modify the values in the text box or he can click the correct value for a field on an image of the document. These corrections are then sent to the feedback algorithm described in this paper. For simplicity, we concentrate on feedback that is given by clicking on the document's image in the remainder of this paper.

4 Incorporating User Feedback

Feedback processing consists of three main components shown in Fig. 3(a) and discussed in the remaining section. At first, error recognition is explained with three different types of errors. Based on classification and index field errors models are updated and finally less frequently used entries are removed to shrink models.

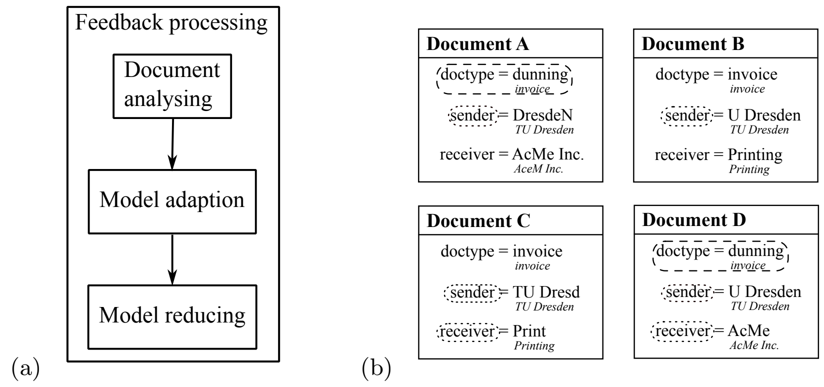


Fig. 3. (a) Feedback processing steps. (b) Faulty processed documents with highlighted grouping. User feedback is marked italic for each extracted value.

4.1 Document Analysis

Feedback processing starts by analysing differences between extracted data and user provided feedback data. User data consist of the corrected document type in case of classification issues and corrected index field with position coordinates. Extraction errors are identified by one of two strategies.

Classification errors are found by checking directly relevant index data and feedback fields. Extraction errors consist of all faulty index fields. Therefore the strategy checks all available extracted and user feedback fields of a document and ignores extraneous fields, e.g., document type or language.

Up to now, three different error types are available. If there was yet no training example available, fields are marked as training error for immediate learning. If classification or extraction delivered wrong results, classification errors and extraction errors are distinguished. Each error is associated with a learning threshold needed for the model adaptation component as described below.

If an error is similar to a previous error, they are grouped together. This way it is possible to consider only groups of errors, thus avoiding overfitting because of outliers. Grouping works as presented in Fig. 3(b). Two classification errors get grouped only if the extracted document type (doctype) is not the same as the user provided one and both expected types are equal. Two extraction errors are grouped if the same index fields are faulty. This happens under the assumption that the model does not consist of a template with proper positions for these index fields.

The example in Fig. 3(b) groups documents A and D because of the same classification error with equal expected document type. Documents B and C are correctly classified. Extraction errors occur in all documents. Documents A and B as well as C and D are grouped because of similar faulty index fields, as we are grouping by fields and their faulty detected template position and not the extracted value.

4.2 Model Adaptation

After document analysis the feedback process performs model adaptation in two phases.

At first the error groups from the previous document analysis step are checked against a threshold t_e associated to each error type e. Training error groups are immediately added to the model as described in the second phase. If the size of classification or extraction error groups is greater or equal to t_e it is marked for consideration in the second step of model adaptation. For example if $t_e = 2$ and analysis found an error group of size 3, the error represented by that group will be marked for the next step.

In the second phase all marked entries are finally incorporated into the extraction or classification model. For our k-NN-based classifier and extractor this simply means adding the documents with its user-provided correct values to the model. In case of classification, the correct document type is added with all words occurring in the document. The extraction model is extended with a new template containing words with their positions.

4.3 Model Reduction

If documents are only added to the model, it will grow indefinitely. Therefore the last feedback processing step removes old documents, that were not used for some time.

The concept of model reduction is based on a voting mechanism. For this purpose it is necessary to monitor which document or documents from the model are the source of information for classification or extraction. So each time a document is used its votes are incremented.

We assume documents having the fewest votes are the least relevant ones. So models are finally reduced by removing such documents. Three parameters influence model reduction. The maximum model size max_m, the minimum model size min_m and the probation period p. As soon as the model size reaches max_m the feedback process removes $max_m - min_m$ documents from the model. If reduction would simply remove documents with the least amount of votes it would mostly remove all documents, added during the model adaptation phase recently. To avoid this, the probation period p protects documents from reduction until p extractions or classifications using that model took place.

5 Evaluational Results

This section shows the variance over extraction and classification quality, while applying user feedback over time. To measure quality we calculated extraction recall and precision per extracted index field. To describe the extraction quality for a document as a single value, we calculated the mean of the accuracy for all fields per document.

The following section first introduces the dataset used for evaluation, before showing detailed results and conclusions.

5.1 Dataset

The dataset consists of 5,627 business documents obtained from the archive of a mid-sized German company. Index fields and document types are labelled for all documents by human annotators. The set consists of seven different document types distributed as shown in Fig. 4. This distribution mirrors the expected distribution of business documents in German companies. Each document type includes several different document styles with different layouts. The labelled index fields are shown in Table 1. These are typical index fields for German business documents. However not all of them occur in every document. So a correct extraction (true positive) is measured if the field exists in the document and the extracted value is correct.

5.2 Experimental Setup

The experiments consist of 20 single runs over the whole document set. Each run starts from an empty classification and extraction model. It processes all documents one by one. Which document to process next is based on a random choice. The user feedback is simulated using the human annotated labels. Based on this feedback information, documents are added to the models as described in Section 4. To calculate the current extraction quality, precision and recall are averaged over the 50 last processed documents. For this purpose the extracted index fields are compared with their expected value. Only in case of exact matching the extracted value is accepted as correct (T). If any other value is extracted it takes count as wrong (F), excluding the empty string which leads to a no result (NR) count. All available index fields of the last 50 processed documents are

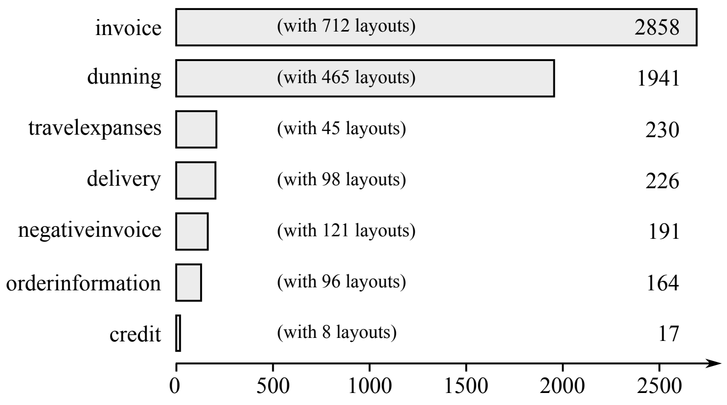


Fig. 4. Document type distribution of the document test set. Each document type contains different layouts listed in the bars.

Table 1. Index fields used for evaluation

Index field
sender
recipient
date
contactperson
tobepaid
docnumber
amount
customerid
contactnumber
email
subject

used to obtained results. This leads to measure accuracy as described at Equ. 1. Results for a single run are shown in the lower plots of Fig. 5.

$$accuracy = \frac{\mathrm{T}}{\mathrm{T} + \mathrm{F} + \mathrm{NR}} \tag{1}$$

The system is evaluated with different configurations for the three parameters introduced in Section 4.3. The configurations are listed in Table 2. All three model parameters are varied for classification models as well as for extraction models. Additionally, each of those configurations is tested for different grouping thresholds t (see Section 4.2). At first t was set to 1 to simulate immediate learning. Then it was set to 3 to research the influence of error grouping on processing performance. The results for each configuration are averaged over 20 runs to remove outliers.

Table 2. Tested model configurations

	max_m	min_m	p
Classification configuration 1	50	40	30
Classification configuration 2	100	80	50
Classification configuration 3	200	180	80
Extraction configuration 1	400	350	70
Extraction configuration 2	1000	800	100
Extraction configuration 3	1500	1300	150

5.3 Representative performance

Figure 5 shows the performance of one single run with model configuration 1 (See Table 2) and threshold $t = 1$. The upper left side shows results for classification whereas the upper right shows the same for extraction. Obviously both start to grow linearly until their size reaches max_m. This happens after roughly 500 documents are processed. At this point model reduction is invoked and reduces the model to size min_m. Afterwards the model grows linearly until it reaches max_m again and so on. This behaviour continues until all documents are processed. For reasons of space the plots in Fig. 5 are truncated after 3,000 processed documents.

The lower graphics show the model adaptation effect. At first average accuracy rises quickly, staying at a constant high performance after initialization. Interestingly there seems to be no correlation between model reduction and extraction or classification performance. We actually expected performance would drop after reduction, rising again until the next reduction occurs. However the quality variation seems to be a result of random ordering of test documents.

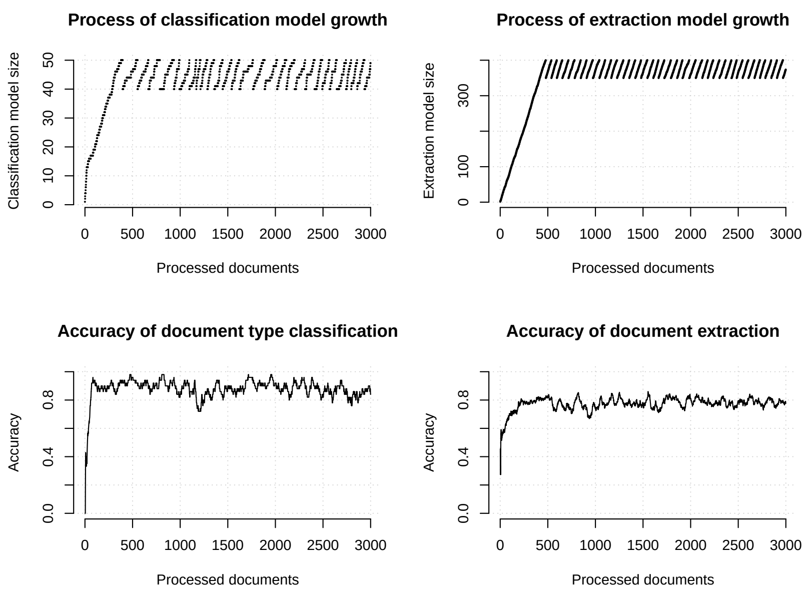


Fig. 5. Evaluation results of one single run with model configuration 1 and immediate learning threshold 1.

5.4 Average performance

The following section explains test results including changes in processing quality as well as runtime performance of our implementation. Table 3 shows the average accuracy for 20 runs and for each of the 3 configurations presented in Table 2. As expected the results indicate that a larger model leads to better accuracy. This is true because a larger model is able to handle more document variations than a smaller one. However, doubling max_m causes only a slight increase in extraction quality. We can therefore conclude that increasing the model's maximum size is only helpful to a certain extend. Table 3 also shows the influence of different error group size thresholds t. It seems that variations of t have no effect on document type classification but increasing the threshold from 1 to 3 causes a small drop in extraction quality. This might be because classification is less error prone then extraction. While we require a document with the same index values at the same positions for extraction, we require only a document containing similar keywords and having the same type for classification.

The graphs in Fig. 5 show a two phase structure. At first the model is initialized with a certain amount of process steps and afterwards shows almost constant behaviour. We investigated this behaviour by counting the amount of processing steps necessary to reach average accuracy of 80%. Table 4 shows the results for the three different configurations presented in Table 2 and the two error group size thresholds. Even though the amount of processing steps for dif-

Table 3. Average accuracy

	Threshold 1		Threshold 3	
	Classification	Extraction	Classification	Extraction
Configuration 1	0.871	0.775	0.870	0.763
Configuration 2	0.90	0.801	0.899	0.781
Configuration 3	0.924	0.811	0.908	0.785

ferent configurations does not vary much, the influence of the threshold becomes more obvious. A threshold of 3 causes classification to need 50% more processing steps, while extraction takes 3-4 times more steps to build up its model. This was expected since for $t = 3$ an error needs to occur at least 3 times until a document containing the required information is included in the model.

Table 4. Model generation performance. A model is considered to be generated if accuracy hits more than 0.8. Amount of processed documents is shown with proportion of test set.

	Threshold 1		Threshold 3	
	Classification	Extraction	Classification	Extraction
Configuration 1	62 (1.1%)	248 (4.4%)	91 (1.6%)	1271 (22.6%)
Configuration 2	56 (0.9%)	319 (5.6%)	124 (2.2%)	1064 (18.9%)
Configuration 3	65 (1.1%)	345 (6.1%)	139 (2.4%)	1225 (21.8%)

Interestingly, there seems to be no benefit in grouping errors and using a threshold larger than 1, although this was expected when designing the algorithms. Model generation takes longer and results do not improve. However a runtime analysis shows a benefit presented in Table 5. Increasing the threshold from 1 to 3 causes our implementation to become 4.9 times faster.

Table 5. Runtime per document of processing and feedback with full set of 5,631 test documents.

	Threshold 1		Threshold 3	
	Processing	Feedback	Processing	Feedback
Configuration 1	38.6 ms	15.1 ms	39.3 ms	5.6 ms
Configuration 2	48.3 ms	30.3 ms	44.5 ms	7.2 ms
Configuration 3	53.2 ms	36.7 ms	43.8 ms	7.6 ms

6 Conclusion

We presented an approach to adapt automatic index extraction for business documents to a business' day-to-day situation as well as changing requirements. We also show how to keep indexing quality constant while documents are changing over time. Furthermore, the evaluation shows that even with a fixed maximum model size our method constantly performes with an accuracy of more than 80% using less than 10% of our dataset of real world business documents. Additionally, our model adaptation approach is designed to use data provided by day-to-day users from accounting or management. Therefore, it avoids configuration by an administrator and is usable by small and mid-sized enterprises.

Current challenges include index fields with variable positions and incomplete user feedback. We are going to address these issues in future work.

References

1. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the eleventh annual conference on Computational learning theory. pp. 92–100. COLT' 98, Madisson, WI, USA (1998)
2. Cauwenberghs, G., Poggio, T.: Incremental and decremental support vector machine learning. In: Advances in Neural Information Processing Systems 13. pp. 409–415. MIT Press (2001)
3. Culotta, A., Kristjansson, T., McCallum, A., Viola, P.: Corrective feedback and persistent learning for information extraction. Artif. Intell. 170, 1101–1122 (2006)
4. Esser, D., Schuster, D., Muthmann, K., Berger, M., Schill, A.: Automatic Indexing of Scanned Documents - a Layout-based Approach. In: Document Recognition and Retrieval XIX (DRR). San Francisco, CA, USA (2012)
5. Huang, Y., Mitchell, T.M.: Text clustering with extended user feedback. In: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 413–420. SIGIR '06, Seattle, WA, USA (2006)
6. Jia, Y., Yan, S., Zhang, C.: Semi-supervised classification on evolutionary data. In: Proceedings of the 21st international jont conference on Artifical intelligence. pp. 1083–1088. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2009)
7. Raghavan, H., Madani, O., Jones, R.: Active learning with feedback on features and instances. J. Mach. Learn. Res. 7, 1655–1686 (December 2006)
8. Saund, E.: Scientific challenges underlying production document processing. In: Document Recognition and Retrieval XVIII. DRR 2011, San Francisco, CA, USA (2011)
9. Stumpf, S., Rajaram, V., Li, L., Burnett, M., Dietterich, T., Sullivan, E., Drummond, R., Herlocker, J.: Toward harnessing user feedback for machine learning. In: Proceedings of the 12th international conference on Intelligent user interfaces. pp. 82–91. IUI '07, Honolulu, HI, USA (2007)
10. Stumpf, S., Rajaram, V., Li, L., Wong, W.K., Burnett, M., Dietterich, T., Sullivan, E., Herlocker, J.: Interacting meaningfully with machine learning systems: Three experiments. Int. J. Hum.-Comput. Stud. 67, 639–662 (August 2009)
11. Wong, W.K., Oberst, I., Das, S., Moore, T., Stumpf, S., McIntosh, K., Burnett, M.: End-user feature labeling: a locally-weighted regression approach. In: Proceedings of the 16th international conference on Intelligent user interfaces. pp. 115–124. IUI '11, Palo Alto, CA, USA (2011)