Modelspace
Cooperative Document Information Extraction in Flexible Hierarchies

Daniel Schuster, Daniel Esser, Klemens Muthmann and Alexander Schill
Computer Networks Group, TU Dresden, Dresden, Germany

Keywords: Self-learning Information Extraction, Cooperative Extraction, Document Archiving, Business Documents.

Abstract: Business document indexing for ordered filing of documents is a crucial task for every company. Since this is a tedious error prone work, automatic or at least semi-automatic approaches have a high value. One approach for semi-automated indexing of business documents uses self-learning information extraction methods based on user feedback. While these methods require no management of complex indexing rules, learning by user feedback requires each user to first provide a number of correct extractions before getting appropriate automatic results. To eliminate this cold start problem we propose a cooperative approach to document information extraction involving dynamic hierarchies of extraction services. We provide strategies for making the decision when to contact another information extraction service within the hierarchy, methods to combine results from different sources, as well as aging and split strategies to reduce the size of cooperatively used indexes. An evaluation with a large number of real-world business documents shows the benefits of our approach.

1 INTRODUCTION

Document archiving and management systems are essential to keep pace with the flow of business documents that are exchanged on a daily basis. While they have been used by large organizations for decades, such systems now reach small and medium enterprises (SMEs) and small office/home office (SOHO) users as well. Techniques for information extraction enable semi-automatic document analysis and indexing which eases the archiving process dramatically (Marinai, 2008). In the past there has been heavy use of rule-based techniques, requiring a long and expensive adaptation phase to the organization's domain. Self-learning systems in contrast skip this adaptation phase improving results and adapting to new documents via user feedback.

Such self-learning information extraction (IE) can work either locally (based on knowledge within the organization) or centralized (based on knowledge in the cloud). The local approach is highly scalable and can be personalized to the user's need. It offers high privacy as no documents cross the organization's border for IE purposes. However, extraction efficiency is lower, as local IE systems only use local user feedback to improve the results. Especially in the starting phase, many documents need to be indexed manually which causes user frustration. We call this the cold start problem.

Centralized IE systems offer better performance in the starting phase as they build on knowledge from a world-wide community of users. But documents have to leave the organization for indexing. Furthermore, results cannot be personalized the same way as in local systems. Scalability is an issue, as IE tasks are computationally expensive and need to run on the provider's infrastructure.

In this paper we discuss the idea of combining both approaches in flexible hierarchies of IE services. The IE compound acts local in the long term thus offering good scalability and privacy. IE services are able to cooperate in the starting phase to improve the experience of new IE users, i.e., eliminating the cold start problem.

We already introduced the idea of using a parent IE service for local IE services in earlier work (Schuster et al., 2013b). The results of this work were then field-tested in a big document archiving solution. While we could improve the cold start experience of the pilot users, some problems still need to be solved to be able to use this idea on a larger scale. This paper thus deals with practical issues when deploying information extraction hierarchies on a large scale.

First, we need proper strategies to decide which documents to share within the IE service hierarchy. Documents should only be shared to avoid the cold

Schuster D., Esser D., Muthmann K. and Schill A..
Modelspace - Cooperative Document Information Extraction in Flexible Hierarchies.
DOI: 10.5220/0005376403210329
In *Proceedings of the 17th International Conference on Enterprise Information Systems* (ICEIS-2015), pages 321-329
ISBN: 978-989-758-096-3

start problem. As soon as there is enough information in the local IE service, documents should be handled locally. The second problem concerns the combination of results from multiple IE services. An aggregated result set has to be created. Third, we need to test the overall scalability of the solution to prove that it is able to handle millions of documents.

Our approach offers strategies to solve these three challenges. We tested these strategies in our own IE system and provide evaluation results on large sets of business documents.

The remainder of this paper is organized as follows. In Section 2 we give a general definition of the problem of extracting relevant information from business documents. Section 3 focusses on related works in the area of local and centralized IE and presents open research questions regarding the combination of these topics. In Section 4 we present an overview of our approach called Modelspace. Section 5 goes more in detail and gives a technical view on strategies and algorithms we developed to solve the introduced problems. We show evaluation results in Section 6 and conclude the paper in Section 7.

2 PROBLEM DEFINITION

The information extraction task to be carried out for document archiving can be described as follows:

Given any scanned, photographed or printed document, an OCR process is called to transfer the document in a semi-structured representation, which includes text zones. Each word from these text zones is recognized as a pair $(text, boundingbox)$. The bounding box contains the relative coordinates of the upper left and lower right corner of the rectangular area where the word is printed on the document page. The information extraction system takes a document in this representation and tries to retrieve values for typical archiving index fields like document type, sender, receiver, date or total amount.

The IE process within a self-learning system is mostly done following the two steps of classification and functional role labeling (Saund, 2011). Classification (or clustering in case the class of training documents is not known) tries to identify a group of similar training documents that share the same properties as the document to be extracted. Properties defining a group may be the type, sender, period of creation or document template. Functional role labeling uses these similar training documents to create a knowledge base of where to find the field values. For each of the fields the IE system finally returns a list of candidate tuples of the form $(value, score)$ sorted by score descending.

Whenever the results are not sufficient for the user, he may correct them offering the right result pairs $(text, boundingbox)$ for each of the recognized fields. This feedback set is added to the knowledge base of the IE system and improves future results.

The IE system should not depend on an initial set of training documents nor should it use any language- or document-dependent rules to improve extraction results. The approach should be purely self-learning to be able to adapt to user needs without intervention of a system administrator.

3 RELATED WORK

Solutions for extracting relevant information out of documents are used in many domains. These approaches can be differentiated based on the structuredness of the documents. While extraction methods for unstructured documents are built on the document's text (Nadeau and Sekine, 2007), approaches for semi-structured documents, like business documents (Marinai, 2008) or web pages (Chang et al., 2006), focus mainly on the document's layout and structure.

Within this paper, we are going to deal with another important aspect of extraction systems, namely, the locality of the extraction knowledge. Local and centralized information extraction approaches have been adopted widely by existing scientific and commercial solutions. Depending on the location, extraction knowledge is stored and information extraction is carried out, a system either performs a local or a centralized processing.

Local information extraction systems are used for processing business documents. Especially large and medium-sized organizations rely on such kind of systems as they perform very well under large training sets of business documents and avoid sharing of documents with external services. Examples for such systems are smartFIX (Klein et al., 2004) and Open Text Capture Center (Opentext, 2012).

In centralized systems the task of information extraction is shifted to a centralized component (in the cloud), whose knowledge base is shared among and used by all participants. Examples are commercial services in the area of Named Entity Recognition (e.g., AlchemyAPI (AlchemyAPI, 2013)) or Business Document Processing (e.g., Gini (Gini, 2013)). Scientific works like Edelweiss (Roussel et al., 2001) and KIM (Popov et al., 2004) present scalable servers for information extraction that provide well defined APIs to make use of the centralized knowledge base

to extract relevant information out of transferred documents.

A combination of both approaches, as it is pointed out by this work, is nearly unknown for processing business documents. Schulz et al. (Schulz et al., 2009) take a first step towards a cooperative document information extraction and present an extension for the smartFIX system that enables a transfer of local extraction knowledge between different smartFIX systems. This allows the authors to overcome the limitations of purely local or purely centralized systems and enables the integration of external extraction knowledge into someones own extraction process.

The Intellix system (Schuster et al., 2013b) uses a comparable approach but does not exchange extraction knowledge among organizations as this may reveal sensitive information. Instead, a small fraction of documents is sent to a trusted shared IE service which only delivers extraction results based on shared documents but no information regarding the content of these documents.

While these are the first approaches dealing with a combination of local and centralized information extraction, they only describe a small part of a possible solution. Research problems like finding and requesting external extraction knowledge and aggregating results from different sources are not tackled by the authors.

4 MODELSPACE APPROACH

We call our approach the Modelspace approach as we split the extraction knowledge over different cooperating model spaces. A model space is the data store of an extraction service hosting a set of information extraction models based on the knowledge the service has gained by training or feedback. Figure 1 shows a minimal example hierarchy with three levels and seven model spaces. New documents of a user are first sent to his client model spaces. In case the IE results are not satisfying, the document is escalated to its parent model space which might be a domain model space (like for health care or administration). In the case that the domain model space is of no help again, the document might be sent to a global model space which collects knowledge from the whole indexing community.

This hierarchy model is fully flexible and may be changed during runtime. Each model space can be connected or disconnected to a parent model space any time. Each model space escalates unknown documents to its parent. This decision may be repeated at upper levels of the hierarchy until a model space without a parent is reached. The results are sent back the same way. Thus, model spaces down the hierarchy are able to analyze results from parent model spaces. Furthermore, if final results are corrected by the user, the user feedback is propagated back up to the top-level.

For privacy reasons only bounding box coordinates of extracted values are propagated back to a calling model space. That way we ensure that a child may never learn anything of the documents stored within its parent and thus may never learn about the documents the parent received from other siblings.

While the percentage of escalated documents may be high in the starting phase of a new model space, it should level down in the long term to ensure good scalability. With each new document the model space increase its knowledge base and produces better extraction results, making further escalations dispensable. Thus, upper-level model spaces will rarely be contacted by model spaces which are already running for quite some time. This compares to the concept of hierarchical web caching where we expect the local cache to serve 90+% of the requests.

It can be seen, that the three problems mentioned in the introduction are to be solved here. We need a strategy for the *escalation decision* to escalate only the documents which have a good probability to get their extraction results improved by the parent. The parent results need to be *combined* with local results. This is no trivial task since our experiments show that good results are not achieved by simply overwriting local results with parent results. For some fields the local results might be better so an adaptive strategy is needed. Third, model spaces at upper levels in the hierarchy tend to grow in size with more documents sent by child model spaces. We need a strategy to *control the size* of parent model spaces which should be optimized regarding the extraction efficiency.

5 STRATEGIES AND ALGORITHMS

The following section explains the solutions we propose for the three main challenges: escalation, aggregation, and controlling size.

5.1 Escalation Strategies

To control the decision when to give a document to a parent model space for extraction various features can be used. Within this work we present two different strategies based on the scores of the extracted results given by the IE system (quality-based) and

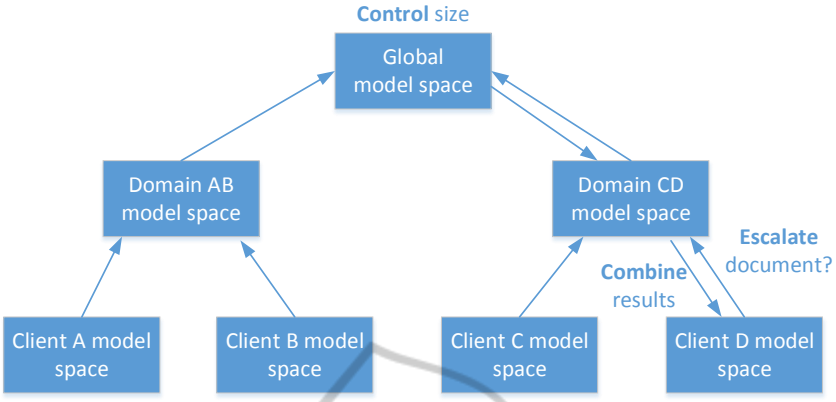

Figure 1: Modelspace hierarchy.

the historical performance of the parent model space (improvement-based) as a basis for a decision.

Most information extraction systems provide a confidence score for each extraction result. The IE scores should in theory provide the best indicator. But often, the scoring does not fully correlate with actual extraction efficiency. Thus scoring efficiency of information extraction components should be tested and adjusted as described in (Schuster et al., 2013a).

The second feature describes the history of results delivered by the parent and follows a simple assumption: The more often a parent model space extracts correct results in the past, the more likely he will provide correct results in the future. This feature somehow creates a profile of the parent and may be used to calculate the likelihood of improvement.

Quality-based Strategy. If the IE scores correlate with the correctness of an extraction result, i.e., a score of 0.0 indicates a wrong extraction whereas a score of 1.0 points to a correct extraction, an escalation threshold t_{esc} can be calculated to separate expectedly correct results from expectedly wrong results like in the following equation:

$$t_{esc} = \begin{cases} \frac{\mu_{inc}+\mu_{cor}}{2}, & \text{if } \mu_{inc} \leq \mu_{cor} \wedge (\sigma_{inc}+\sigma_{cor}) = 0 \\ \frac{\mu_{inc}\cdot\sigma_{cor}+\mu_{cor}\cdot\sigma_{inc}}{\sigma_{inc}+\sigma_{cor}}, & \text{if } \mu_{inc} \leq \mu_{cor} \wedge (\sigma_{inc}+\sigma_{cor}) \neq 0 \\ 1, & \text{if } \mu_{inc} > \mu_{cor} \end{cases}$$

μ_{cor}: mean score of correct extractions
μ_{inc}: mean score of incorrect extractions
σ_{cor}: standard deviation of correct extractions
σ_{inc}: standard deviation of incorrect extractions

The mean score of correct results is weighted with the standard deviation of incorrect results and vice-versa. This causes the threshold t_{esc} to smoothly balance between μ_{inc} and μ_{cor}. The calculation of the equation above has to be done separately for each field thus to enable an adaptive escalation decision.

The effect can be shown on a small example with $\mu_{inc} = 0.3$ and $\mu_{cor} = 0.9$ with standard deviations $\sigma_{inc} = 0.2$ and $\sigma_{cor} = 0.1$.

$$t_{esc} = \frac{\mu_{inc}\cdot\sigma_{cor}+\mu_{cor}\cdot\sigma_{inc}}{\sigma_{inc}+\sigma_{cor}} = \frac{0.3\cdot0.1+0.9\cdot0.2}{0.2+0.1} = 0.7$$

While 0.6 would have been the average of the two mean values, the threshold is slightly higher as the standard deviation of incorrect results is higher than the standard deviation of correct results.

Improvement-based Strategy. The improvement-based strategy records the performance of the parent model space for each field in the past and calculates the possibility that the parent will improve the current local results.

This strategy first determines the number n_{sim} of similar training documents available for the current test document. This is based on the classification step that identifies training documents of the same class (type, sender, time, or template). The concrete implementation of the classification (we use a Lucene-based kNN classifier approach (Schuster et al., 2013b) to find documents with the same hidden document template) is not relevant for understanding the strategy. Knowing the number of similar documents, the following algorithm is carried out:

if $n_{sim} = 0 \rightarrow$ Escalate
if $0 < n_{sim} < t_{sim} \rightarrow$ Compare
if $n_{sim} \geq t_{sim} \rightarrow$ Do not escalate
n_{sim}: number of detected similar training documents
t_{sim}: similarity threshold

Thus documents which do not create a single similarity match within the set of training documents are

escalated to the parent without any further calculation. Documents which create at least one similarity match but less then the threshold t_{sim} have to be considered for escalation. Documents which create t_{sim} or more matches are very likely to be extracted correctly using the local model space only. They are thus not escalated to the parent. The parameter t_{sim} has to be determined by experiments for a concrete implementation. We had good results with a value of $t_{sim} = 4$.

For the Compare operation we take the last extraction of the best matching similar document into consideration. We record local result candidates as well as parent candidates for each extraction process. In case the parent extraction improved results of the best matching similar document in the past, we assume that the parent will also be helpful for the current test document and escalate the test document. Thus a document prevents escalation of similar documents once it was escalated but did not improve results.

Obviously, the improvement-based strategy only considers whole documents and is thus not able to fine-tune weights for each one of the fields.

5.2 Aggregation Strategies

Aggregating results from different model spaces should be done based on field-based weights which should be adapted or improved by user feedback. According to (Schuster et al., 2013a) the process to create a good result aggregation consists of the four steps selection, normalization, weighting and combination. The following strategies allow combination of results from multiple model spaces, however in practice we only aggregate results from a model space and its parent.

In case of combining the result from local and parent model space, the selection step can be skipped as we don't want to ignore one or more sources. Normally the selection is used to eliminate information extraction components which do not improve the results significantly. As an information extraction system develops and grows over time, the number of extraction algorithms rises. While each one of them is proven to be useful for different test setups, some of them might be useless in combination with other algorithms, as they only detect results which have already been found. Moreover, the combination with such a redundant algorithm will typically reduce correctness, as there is always a residual probability of claiming correct results of other algorithms as wrong.

After selection, the scores of the local and parent model space need to be normalized as already described above. This step requires a high correlation between high scores and correct results as well as between low scores and incorrect results. While the first is true for almost all scoring approaches, the latter is often wrong and requires adjustment of the scoring approach.

We use an adaptive approach for the weighting step. Based on user feedback we calculate the precision per field f of each model space m. We do this by counting the correct results (COR), incorrect results (INC) and spurious results (SPU) of past extractions in dependence on the definition of Chinchor and Sundheim (Chinchor and Sundheim, 1993). The field weight is then calculated by dividing the field precision of the model space by the sum of all field precisions of this field. Thus higher precision results in a higher weight.

$$Precision_{m,f} = \frac{COR_{m,f}}{COR_{m,f} + INC_{m,f} + SPU_{m,f}}$$

$$Weight_{m,f} = \frac{Precision_{m,f}}{\sum_{m \in M} Precision_{m,f}}$$

For the combination step we have to group and sum up the scores of different model spaces m by similar values v. Thus, if model space m_1 detected a value v_1 with score 0.8 and the parent model space m_2 detected the same value with score 0.7, the scores will be added up but each weighted with the field-specific weight which is reflected in the following equation.

$$CombScore_{f,v} = \sum_{m \in M} Weight_{m,f} \cdot Score_{m,f,v}$$

But a problem occurs in case of missing values for some fields. If the parent refuses to detect something for a field f, this will produce a score of 0 to be combined with the local score. This may cause the local value to be dismissed because of a total score below the threshold for correct results. To eliminate this problem we use a slightly modified version to calculate a combined score which lifts results of one model space in case of missing results from other model spaces by only combining results from those model spaces ($M_{f,v}$) that delivered a value v for field f. The modified version $LiftedScore$ is calculated by the following equation.

$$LiftedScore_{f,v} = \frac{\sum_{m \in M_{f,v}} Weight_{m,f} \cdot Score_{m,f,v}}{\sum_{m \in M_{f,v}} Weight_{m,f}}$$

Finally, the combined result list for each field f is sorted by descending combined score. If the value on top of the list is above some predefined threshold Δ, it is returned as the extracted result for field f.

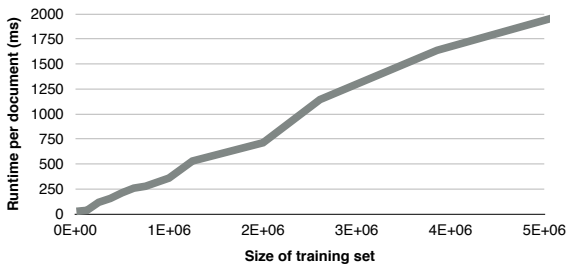

Figure 2: Scalability test.

5.3 Size Control Strategies

The third building block of hierarchical information extraction to be discussed here are control strategies to ensure the scalability of the approach. Clearly, model spaces which are on upper levels of the hierarchy, will grow quickly and have to establish some size control strategies.

Granularity Tests. To quantify the critical size of model spaces we did extensive scalability tests. We used our Lucene-based kNN classifier for detection of template documents. We further developed a document randomization strategy to create a large number of documents from a set of 12,000 business documents. By appending numbers to document words we were able to create a document set of 5 million different generated documents which behave like business documents and show the same distribution of document templates than the smaller set.

We compared the runtime of the template detection per document with rising size of the training set using the Lucene file system index on a quad core system (Intel Core i7 with 2.3 GHz) with 16 GB RAM. As Figure 2 shows the response time grows steadily and exceeds 1,000 ms at about 2.5 million documents in the index. For our test maximum of 5 million documents, the response time doubled to about 2,000 ms.

This obviously reflects the computational complexity of the kNN algorithm which is $O(kn)$. While the total numbers will differ, any self-learning IE system for business documents will have to do a similarity search on training documents. Lucene-based kNN is just an example implementation but other systems will typically show comparable results.

Reduction Strategies. The goal of reduction strategies is to limit the number of training documents in a model space to some predefined threshold n_{max} which is known to be a borderline for good performance of the system (e.g., 2.5 million documents in our case as not to exceed 1 s response time).

The first idea to control the size of parent model spaces is to eliminate documents that did not contribute much to extraction results. This is comparable to cache replacement strategies. However, a simple least recently used (LRU) strategy will not succeed, as we are especially interested in improving the behavior in the starting phase. We need a strategy that maximizes the number of classes that are recognized by a parent model space with sufficient accuracy.

Thus we need to group documents by class and we need statistics on the utility of each document in the training set. The utility is based on the following criteria: (1) The age of the document, (2) how often did the document help to extract correct / incorrect results (based on user feedback) and (3) does the feedback for this document match the feedback of other documents with the same class?

The last criteria is especially helpful to eliminate documents which got some wrong feedback accidentally. Each document of a class gets a rank according to the three measures above. The ranks are then combined using a rank aggregation strategy (e.g., simple Borda ranking).

When we extract a new test document, we get a set of similar documents SIM, where the number of documents is $n_{sim} = |SIM|$. Whenever $n_{sim} \geq t_{sim}$ (see section 5.1), one or more documents with low utility value may be safely removed from the model space.

If the steps above are not sufficient to reach the desired maximum number of training documents n_{max} in the model space, it may still be beneficial to remove documents, as long as the number of similar documents is in the range $1 < n_{sim} < t_{sim}$. This will already slightly decrease the extraction accuracy but at an acceptable level while the number of classes is not reduced.

Splitting Strategies. If we need to further reduce the size of a model space, the next option is to split a model space in two or more sub model spaces. Why should this be beneficial? Documents are often domain-specific (health care, building sector) and are thus only important for a subset of users. If we are able to identify clusters of users with different information needs, it should be possible to split a parent model space as well as the user base without loosing extraction accuracy.

Figure 3 shows the idealized idea of domain-specific splitting. Within the model space there are two independent clusters which do not show any overlap of users. Thus, it is possible to safely split the model space and its user base without loosing extraction accuracy.

We developed a clustering strategy that records the

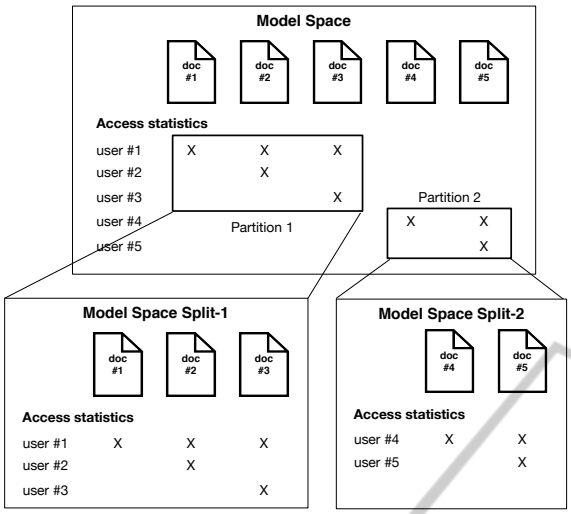

Figure 3: Splitting strategy.

user IDs of accessing users for each training document. It then tries to find k clusters (typically with $k = 2$) with low or no overlap of the user base. This can be done with traditional clustering techniques like k-Means.

6 EVALUATION

To evaluate the strategies discussed in Section 5 we implemented them for our hierarchical information extraction system.

We used a dataset of 12,000 multilingual real-world business documents for the evaluation as mentioned above. We captured each document with a stationary scanner and tagged commonly used fields in document archiving to create a proper gold standard. Beside a minimal set of fields to enable structured archiving (document type, recipient, sender, date), we added further popular fields like amount, contact, customer identifier, document number, subject, and date to be paid. In total we identified 105,000 extractable values, we will further on use for evaluating our approach. We split the dataset in 4,000 documents each for training, validation and test.

The evaluation is always done iteratively, i.e., we start with an empty model space and then input each of the 4,000 documents from the test set as a test document. In case the IE system does not deliver proper results, we deliver the values from the gold standard as emulated user feedback. In case of a distributed setup, the 4,000 training documents are added to the parent model space before testing.

For evaluation, we focus on the common information retrieval and extraction measures precision, recall and F_1 according to Sebastiani (Sebastiani, 2002) and Chinchor and Sundheim (Chinchor and Sundheim, 1993).

Table 1: Evaluation of escalation.

Strategy	F1	% Escalated
Always	0.0631	100.0 %
Never	0.9369	0.0 %
Quality-based	0.1615	71.9 %
Improvement-based	0.9449	10.3 %
Ideal	1.0000	6.3 %

Due to privacy concerns our dataset of 12,000 business documents is not yet publicly available. To reach this goal and to allow other researchers to compare with our solution, we currently work on methods to anonymise our documents by replacing critical information but not breaking the structure and intention of the documents.

6.1 Escalation and Aggregation

The results of the evaluation of the escalation strategies are shown in Table 1. In our test set 68.1% of the documents contained at least one field which had a missing or incorrect value. But only 6.3% of all documents could actually be improved by the parent which was pre-trained with the training set. Thus the ideal strategy should only escalate these 6.3% of the test set.

The quality-based strategy was quite good in matching the 68.1% of documents with erroneous values. It selected 71.8% of documents for escalation. But as only 6.3% did improve at the parent, this strategy gets a quite bad F_1 measure of 0.1615. The improvement strategy was able to check the parent if it has been helpful for documents of this template in the past. This much better matches the ideal escalation strategy. The improvement-based strategy reaches an F_1 measure of 0.9449 and escalates only 10.3% of the documents. This is much more resource-efficient than the quality-based strategy.

The evaluation of the aggregation involved two evaluation setups. In the first round we used the validation set to determine the weights for the different fields for the local model space as well as the parent model space. We got weights in the span 0.46 to 0.55 thus showing an almost even weighting for both local and parent model space. The second round of evaluation tried to find an optimal value for the threshold Δ to distinguish correct from incorrect results based on our validation set. According to our aggregation strategy LiftingScore, a value of $\Delta = 0.15$ reaches the highest F_1 measure of $F_1 = 0.8893$.

The specification of our strategies and parameters

are of course heavily based on the system's concrete implementation.

6.2 Reduction and Splitting

For evaluating our reducing and splitting strategies, we compared the extraction effectivity of large model spaces with the extraction effectivity of reduced and splitted model spaces. Therefore we trained a single model space using the business documents in our training set and evaluated it against our test documents. Afterwards we split the model space with the help of our size control strategies into new model spaces and evaluated them against our test documents partitioned by user ID.

On average, we could identify a decrease of F_1 measure by 1.85 percentage points, while time for processing a single document speeds up by 4.26 percentage points. This decrease is much smaller, the more documents are stored in a model space. With 6,000 training documents we could measure a drop of 0.86 percentage points. The reason is that with large model spaces the probability of having similar documents that can be equally split increases, which results in splitted model spaces that can extract documents of a special type.

Concrete results highly depend on the extraction algorithms and their behavior with small training sets. Algorithms that reach high extraction rates with a low number of training documents perform much better with small model spaces than algorithms that need large sets of examples to build up their knowledge base. A detailed analysis can be found in an earlier work (Esser et al., 2014).

As our document set of 12,000 documents is quite low for testing splitting strategies, we are planing to run large scale testing with business users to put our results on a broader basis.

6.3 Overall

Our last evaluation shows the overall utility of the distributed approach. Figure 4 is a presentation of the results. We used a test setup with 20 local model spaces and one common parent model space where about 25% of document templates were shared, i.e., used by at least two users in their respective model spaces.

To detect the improvement in the starting phase, we used the $F_1@k$ measure. This measure examines the F_1 measure grouped by the number of similar template documents found in the local model space. It shows that the hierarchical approach does not add any value if 4 or more template documents are already available in the local model space. Thus we set the parameter $t_{sim} = 4$ as described above.

The true value of the hierarchical approach can be seen in the case of $k = 0$. This case occurs quite often in the starting phase of the system as the user inserts new documents he never had used before. Only 5% of the fields are extracted correctly in the local scenario, thus leaving the user frustrated with the system. The distributed approach lifts this number to 49% which creates a much better user experience. Thus the user is motivated to work with the system and improve its performance by giving feedback.

7 CONCLUSION

We presented hierarchical information extraction as an approach to eliminate the cold start problem in semi-automatic archiving of business documents. Three main challenges are inherent when transforming a local or centralized IE system to a hybrid hierarchical IE system: proper escalation strategies, strategies for result aggregation, and strategies to control the size of parent model spaces.

Our approach proposes self-regulating strategies based on user feedback for all three challenges thus reaching the goal of a self-learning and self-managing IE system. Systems of this type are able to adapt to any application domain as well as to any size of organization, be it SOHO, SMEs or large organizations.

Nevertheless, during first tests with business users we noticed challenges we are going to solve in the future. Our current approach focussed on a hierarchical approach for information extraction, which easily allows to control the document flow and ensures the privacy of documents, but reduces flexibility as it forces each model space to only use one single parent model space at a time. Breaking up this restriction, i.e. by transforming the tree-based hierarchy to a bidirectional graph would lead to a higher flexibility, but needs additional mechanism, i.e. to avoid escalation cycles or to anonymize business documents before escalation.

Another goal, we are currently dealing with, is the publication of our document set of 12,000 multilingual real-world business documents. Nearly all related works use small sets of private documents, not able to share them due to privacy issues. As far as we know, there does not exist any public dataset of business documents for information extraction that is comparable to ours in size and degree of annotation.

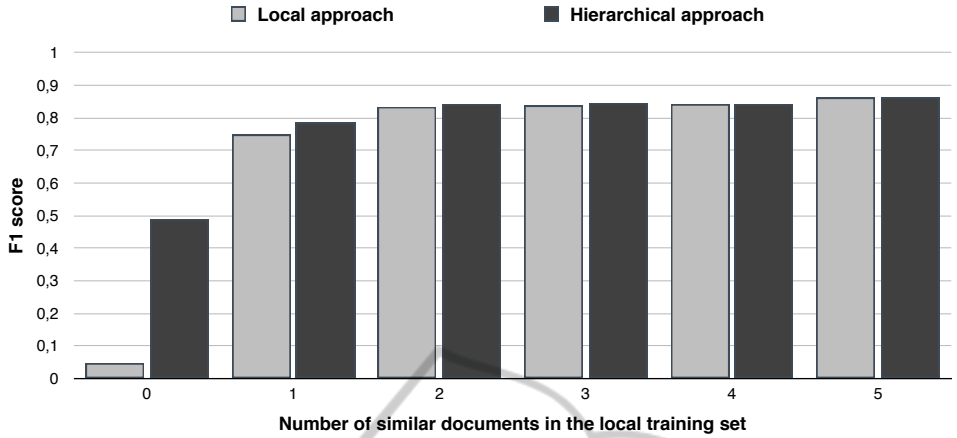

Figure 4: Overall evaluation.

ACKNOWLEDGEMENTS

This research was funded by the German Federal Ministry of Education and Research (BMBF) within the research program "KMU Innovativ" (fund number 01/S12017). We thank our project partners from DocuWare for insightful discussions and providing us with the document corpus used for evaluation.

REFERENCES

AlchemyAPI (2013). http://www.alchemyapi.com/. [Online; accessed 20-August-2014].

Chang, C. H., Kayed, M., Girgis, M. R., and Shaalan, K. F. (2006). A survey of web information extraction systems. *Knowledge and Data Engineering, IEEE Transactions on*, 18(10):1411–1428.

Chinchor, N. and Sundheim, B. (1993). Muc-5 evaluation metrics. In *Proceedings of the 5th conference on Message understanding*, MUC5 '93, pages 69–78.

Esser, D., Schuster, D., Muthmann, K., and Schill, A. (2014). Few-exemplar information extraction for business documents. In *16th International Conference on Enterprise Information Extraction (ICEIS 2014)*.

Gini (2013). https://www.gini.net/en/. [Online; accessed 20-August-2014].

Klein, B., Dengel, A., and Fordan, A. (2004). smartfix: An adaptive system for document analysis and understanding. *Reading and Learning*, pages 166–186.

Marinai, S. (2008). Introduction to document analysis and recognition. In *Machine learning in document analysis and recognition*, pages 1–20. Springer.

Nadeau, D. and Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Opentext (2012). Opentext capture center. http://www.opentext.com/2/global/products/products-capture-and-imaging/products-opentext-capture-center.htm.

Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D., and Goranov, M. (2004). Kim - semantic annotation platform. *Journal of Natural Language Engineering*, 10(3-4):375–392.

Roussel, N., Hitz, O., and Ingold, R. (2001). Web-based cooperative document understanding. *2013 12th International Conference on Document Analysis and Recognition*, 0:0368.

Saund, E. (2011). Scientific challenges underlying production document processing. In *Document Recognition and Retrieval XVIII (DRR)*.

Schulz, F., Ebbecke, M., Gillmann, M., Adrian, B., Agne, S., and Dengel, A. (2009). Seizing the treasure: Transferring knowledge in invoice analysis. In *10th International Conference on Document Analysis and Recognition, 2009.*, pages 848–852.

Schuster, D., Hanke, M., Muthmann, K., and Esser, D. (2013a). Rule-based vs. training-based extraction of index terms from business documents - how to combine the results. In *Document Recognition and Retrieval XX (DRR)*, San Francisco, CA, USA.

Schuster, D., Muthmann, K., Esser, D., Schill, A., Berger, M., Weidling, C., Aliyev, K., and Hofmeier, A. (2013b). Intellix - end-user trained information extraction for document archiving. In *Document Analysis and Recognition (ICDAR)*, Washington, DC, USA.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47.