A Position-Based Method for the Extraction of Financial Information in PDF Documents

Benoit Potvin
Université du Québec à Montréal
Department of computer science
Montréal, Canada, H3C 3P8
benoit.potvin@metix.ca

Roger Villemaire
Université du Québec à Montréal
Department of computer science
Montréal, Canada, H3C 3P8
villemaire.roger@uqam.ca

Ngoc-Tan Le
Université du Québec à Montréal
Department of computer science
Montréal, Canada, H3C 3P8
le.ngoc_tan@courrier.uqam.ca

ABSTRACT

Financial documents are omnipresent and necessitate extensive human efforts in order to extract, validate and export their content. Considering the high importance of such data for effective business decisions, the need for accuracy goes beyond any attempt to accelerate the process or save resources. While many methods have been suggested in the literature, the problem to automatically extract reliable financial data remains difficult to solve in practice and even more challenging to implement in a real life context. This difficulty is driven by the specific nature of financial text where relevant information is principally contained in tables of varying formats. Table Extraction (TE) is considered as an essential but difficult step for restructuring data in a handleable format by identifying and decomposing table components. In this paper, we present a novel method for extracting financial information by the means of two simple heuristics. Our approach is based on the idea that the position of information, in unstructured but visually rich documents – as it is the case for the Portable Document Format (PDF) – is an indicator of semantic relatedness. This solution has been developed in partnership with the Caisse de Dépot et Placement du Québec. We present here our method and its evaluation on a corpus of 600 financial documents, where an F-measure of 91% is reached.

CCS Concepts

•Information systems → Enterprise search;

Keywords

Financial information retrieval; industrial system; PDF documents; semantic relatedness

1. INTRODUCTION

While XBRL (eXtensible Business Reporting Language) [13] is a global standard XML-based language for business information exchanges that can be used to import reliable financial information from structured documents, almost all financial documents in the industry are shared in the Portable Document Format (PDF). This format allows users to read a document on different output devices while offering a compelling visual aspect. PDF documents are meant to be read by humans – not machines – and their content is organized in function of an encapsulated layout, i.e. position of content, style and graphical elements. In other words, PDF documents are structured by and for humans and offer a rich visual presentation where information is nested in the layout and where semantic content may be implicit [8]. Consequently, PDF documents are difficult to interpret with usual Information Extraction (IE) methods and usually require a substantial preprocessing task [3]. Our method exploits the idea that a PDF document is built over these visual characteristics and that its meaning is embedded in the visual representation.

In this paper, we present a novel approach to extract numerical financial information contained in PDF documents. Table Extraction (TE) is considered as an essential but difficult step in IE for restructuring data of tables in a handleable format by identifying and decomposing their elements. While most suggested methods use involved set of rules in order to extract several types of tables, we use only two heuristics based on the idea that the position of information, in unstructured but visually rich documents, is an indicator of semantic relatedness – or as the proverb says *birds of a feather flock together*. Our approach is slightly influenced by the Simrank algorithm [14] where the object-to-object relationship is given by their relative distance (interpreted as proximity)[1].

Our two heuristics are respectively based on the density and the proximity of information. The first heuristic defines a line density threshold that plays the role of a filter in order to eliminate dense blocks of text – mainly paragraphs – where the use of natural language processing (NLP) techniques is more appropriate. Then, our method uses each numerical value of the document as a single pivot point to perform a rectilinear search (i.e. horizontal and vertical, see Yildiz [25] and Silva [7] for a similar approach) in order to find headers and build elementary clusters of relevant in-

[1]We do not implement the recursive SimRank equations. By this comparison, we want to recall the intuition that guided us through the development of our algorithm.

ADCS '16 Monash University, Melbourne, Australia
© 2016 ACM. ISBN 978-1-4503-4865-2/16/12. . . $15.00
DOI: http://dx.doi.org/10.1145/3015022.3015024

formation. The second heuristic uses a distance threshold to aggregate chunks of information that are part of a same neighbourhood. We call these neighbours *satellites* because they gravitate around pertinent information but are not horizontally or vertically aligned with the pivot. Thereafter, elements of a same neighbourhood are analyzed by a domain ontology that corresponds to an exhaustive dictionary of relevant expressions and synonyms. When the elements correspond to a financial term of the ontology, an attribute-value relation is defined and imported into the data structure.

The remainder of the paper is organized as follows. Section 2 presents related works in the domain of financial information extraction systems. Section 3 presents the four tasks of our method, i.e. a preprocessing task, a bucketing task, a two-step clustering task and a structuring task. A summary of our method is given in section 4. Finally, our method is evaluated in section 5.

2. RELATED WORKS

IE systems aim to extract relevant structured information from unstructured documents [20]. These systems can be classified in two main approaches: knowledge engineering and automatically trainable systems [22]. Knowledge engineering systems are made by specialists and are used when reliability is critical but are laborious to develop, difficult to maintain and necessitate expertise in the related domain [17]. When training data sets exist and performance is less crucial, automated trainable systems are preferred [4]. In the financial industry, the need for reliable information is so important that most of the methods applied in this field belong to the knowledge engineering approach.

Commonly accepted methods for the performance of IE systems are precision, recall, and weighted harmonic mean of precision and recall (F-measure):

$$\text{Precision} = \frac{\text{number of correctly identified values}}{\text{number of identified values}} \tag{1}$$

$$\text{Recall} = \frac{\text{number of correctly identified values}}{\text{total number of values to identify}} \tag{2}$$

$$F = \frac{2PR}{(P+R)} \tag{3}$$

Intuitively, precision is a measure of the quality of the identification process, while recall is a measure of task fulfillment. For instance, an extraction system that would return each time only one correct value, even if hundreds of values are sought, would have a precision of 100%. Likewise, a system that would return all values of a document, regardless of the specific values to identify, would have a recall of 100%. The F-measure, which is the weighted harmonic mean of precision and recall, is therefore used to give an overall performance evaluation of IE systems.

In PDF financial documents, tables contain important information. TE is the task of detecting and decomposing table information in a document. TE has the reputation of being a difficult task because of the heterogeneous nature of tables, at least when a general method is desired for unstructured documents [12]. Nevertheless, when information is stored in tables, TE is considered as an essential step for restructuring data in a handleable format. Some authors even insist that table detection is the first task of any table information extraction process [2]. TE is problematic because it draws a baseline for all upcoming task errors as it circumscribes the information that will have to be analyzed. In other words, errors are reflected and amplified over the IE process. Consequently, the problem to automatically extract reliable financial data in unstructured documents remains difficult to solve in practice and even more challenging to implement in a real life context.

Among TE methods that have been used on a large scale, one should mention those of SmartXBRL, an industrial IE solution developed by BCL Technologies [2]. SmartXBRL system aims to simplify and automate the creation of compliant XBRL documents from unstructured financial PDF reports. SmartXBRL's TE task isolates embedded tables and identifies components such as title blocks and table entries by the means of heuristics. Tables and segments are detected by defining non-overlapping rectangles of spaces between elements of the document. A table is recognized if a minimal set of these rectangles is found [1]. In other words, tables are circumscribed in function of the dispersion of information. The algorithm then differentiates – the authors do not clearly explain the rationale here – between the column and row headers and link the cell content to the cell tag. Afterward, a concept search from a list of keywords is performed and a template pattern task searches for word complex patterns in order to extract relevant semantic relations between matched objects. SmartXBRL's evaluation for TE is 96% for precision and 95% for recall [2]. SmartXBRL's performances are impressive but involve numerous heuristics and fine-tuning. Those heuristics are needed in order to locate and tag relevant items for extraction, such as Document and Entity Information (DEI) and numerical values in notes [2] (see [23] for an alternative tagging method used for inferring types of data and indexing columns). SmartXBRL is a great example of a complex reliable industrial solution that meets actual customer's expectations but will have to be regularly maintained and adapted over time.

Yildiz et al. [25] propose a heuristic-based TE method for PDF files. Three tasks are performed: a preprocessing task, a table recognition task and a table decomposition task. Their method corresponds to the very classical approach of TE. The preprocessing task aims to extract the content of the PDF file according to the absolute position of information in the original file. PDF documents contain text, images and vector graphic layers. There are several tools to extract their content such as XPDF (*PDFtoText*, *PDFtoHTML*, etc.) [18], Apache PDFBox [19] and PDFlib. Yildiz et al. have chosen PDFtoHTML where Cascading Style Sheets (CSS) markup is used to locate information relatively to their original position in the PDF document. In other words, the PDFtoHTML tool returns text chunks and their absolute coordinates in the PDF file. Thereafter, they bring together the elements (i.e. text chunks) that are on a same line in order to identify table line candidates, i.e. lines with more than one element. The second task for recognizing tables is based on the assumption that *a table must have more than one column.* If two consecutive lines are candidate lines then they are part of a same table. On the contrary, if more than 5 lines separate two candidate lines then these lines cannot belong to the same table. The decomposition of the table is based on the vertical alignment of horizontal elements (see Silva [7] and Hassan [11] for examples of this approach). In other words, if the horizontal boundaries of an element fit into a column vertical boundaries, this element will be added to the correspondin column. The whole process is rectilinear, i.e. uses a vertical and horizontal mapping for the definition of a table structure. The algorithm's evaluation for one hundred (100) various tables is 83% for precision and 81% for recall.

Con et al. [4] use a template-based approach to extract financial information from documents of the Electronic Data Gathering, Analysis, and Retrieval System (EDGAR) database. EDGAR is an initiative of the Securities and Exchange Commission (SEC) which aim to increase the efficiency of the corporate information filling process with the agency [21]. EDGAR filings are semi-structured documents based on templates, i.e. a document structure with defined and ordered sections, but also contain custom financial reports from companies. The most used EDGAR form is the annual report 10-K filing that provides an overview of the company's business and financial condition and includes audited financial statements [10]. Con et al. define a rich set of heuristics in order to obtain a template-based approach for 10-K forms. The approach is articulated over a *Locator* component that identifies, locates and extracts the targeted blocks of information. The authors give a great demonstration that highly reliable financial data can be extracted by the means of a customizable template-based approach. Their precision, recall and F-measure for the *Locator* task on 603 documents is respectively 96.85%, 94.81% and 95.82% [4].

Named-entity recognition (NER) plays an important role in a variety of information management tasks, including information search and retrieval, text categorization and document clustering. NER seeks to locate and classify elements in text into predefined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. The targeted IE system will extract structured data or knowledge from the unstructured financial texts by identifying references to the named entities as well as stated relationships between such entities through coreference resolution and relationship extraction. Wang et al. [24] propose a NER system that recognize stock names and financial named entities using a domain dictionary and incorporating internal features in CRF classifiers. The authors make use of mutual information, boundary entropy and context features to recognize the abbreviation of financial named entities candidates. Their experiments were completed on a Chinese financial dataset and achieved a precision of 91.02% and a recall of 92.77%.

While the best systems in the literature deal with dozens of rules and make use of predefined templates, we present a knowledge engineering heuristic-based system that uses only two heuristics in order to extract the financial values of PDF documents with heterogeneous layouts. Our system was designed to be minimalist in order to evolve as an industrial long-term solution. Moreover, our method does not explicitly identify and/or decompose tables but rather focuses on the identification of attribute-value couples. From this point of view, it should be understood as a general based-method for information extraction in financial documents. The architecture can also integrate other IE techniques such as statistical methods to improve its results. We will present the integration of an optional NER module to our workflow.

3. METHOD

In the following paragraphs, we present the four successive tasks that represent the rationale of our heuristic-based system: a preprocessing task, a bucketing task, a two-step clustering task and a structuring task. We also present an optional task, i.e. a NER module, to show that our method can be easily joined or adapted to other techniques. This method offers a flexible architecture that can be readily extended in order to integrate further techniques and improve its results.

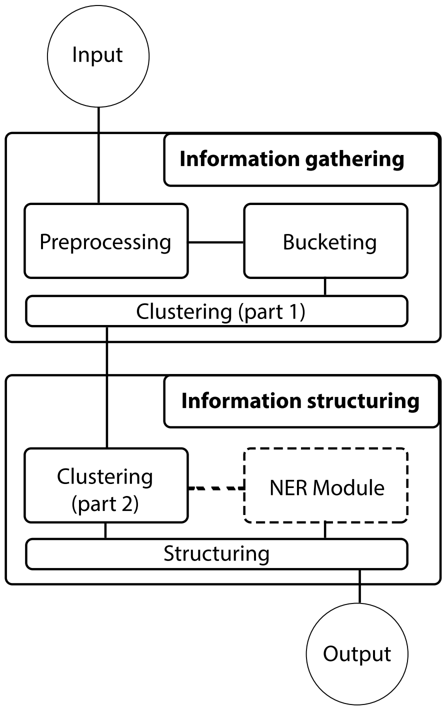

Figure 1: Flowchart of our method

The preprocessing, bucketing and first clustering tasks serve to gather information. Our main concern here is to develop a general gathering process for numerical values that is applicable to all PDF layouts without causing information loss. While these tasks will usually gather superfluous information, the information filtering tasks will eliminate irrelevant information. The second clustering and structuring tasks aim to filter information by the means of an ontology. The ontology plays the role of an exhaustive dictionary of relevant expressions and synonyms. The optional NER module is used to improve our results. The structuring task maps numerical values to corresponding attributes in order to create relevant attribute-value couples that will fit into the defined data model.

More precisely, our method includes the following steps:

1. Preprocessing:
   - Extract the textual content of PDF documents with *PDFtoText*.

2. Bucketing:
   - Split lines in buckets (chunks) of consecutive words.

3. Clustering:
   - (Part 1) Create elementary clusters by the means of a rectilinear search algorithm where a given numerical pivot value is mapped to perpendicular buckets.
   - (Part 2) Using a distance threshold, find, for elementary clusters, related buckets (satellites) and evaluate their relevance by the means of the ontology.

4. Optional NER module:
   - Preprocessing the unstructured text, which includes tokenization, sentence segmentation, part-of-speech tagging and phrase structure parsing.
   - Identification of named entities (ENAMEX), such as names of persons (PER), organizations (ORG), locations (LOC).
   - Identification of numerical entities such as temporal expressions (TIMEX), numbers related to money, percent and units of measurement (NUMEX).
   - Categorization of the recognized NEs – assign predefined categories.
   - Resulting clusters from precedent task (i.e. second part of Clustering) are enhanced with corresponding named-entities in order to facilitate the structuring task.

5. Structuring:
   - With the help of the domain ontology where financial terms, synonyms and relevance indices are defined, map numerical values to corresponding attributes in order to create relevant attribute-value couples that will fit the data model.

3.1 Preprocessing

For our purpose, a preprocessing of the document into free text (.txt) where the position of information complies with the PDF layout is sufficient and *PDFtoText* has been used. Compared to *PDFtoHTLM*, *PDFtoText* does not create extra information (e.g. using CSS or XML) in order to define the original position of information. In a text document where the layout of the original file has been respected, information is organized line per line and correspond essentially to the vertical and horizontal position of information in the original document. Whitespace characters and blank lines in the text document replace blank spaces in the PDF document. From this point of view, a text document is a matrix D_{ij} where the height i corresponds to the number of lines and the width j to the maximum number of characters on one line:

$$D = \begin{bmatrix} T & h & i & s & & i & s & & \\ & & & a & n & & & & \\ & & & & & & & & \\ & & e & x & a & m & p & l & e \end{bmatrix}$$

Therefore, each character has coordinates in the document space. To optimize the process, one could decide to create a matrix for each page.

3.2 Bucketing

By a *bucket*, we mean a group (or bag) of ordered words (i.e. contiguous text strings) located on a same line where two words are not separated by more than one whitespace character. All other characters fail to create a new bucket, e.g. "|", "-" or ";". The creation of buckets (or blocks) respects the assumption that position is an indicator of semantic relatedness where we obtain consistent bags of semantically related words.

This bucketing step aims to minimize computing resources by creating sets of homogeneous information. Every bucket is considered as an unique entity and will be use for further calculations. The set C_D of buckets in matrix D is $\{This\ is, an, example\}$. C_D has three buckets of information because *This* and *is* are separated by only one whitespace character. Note that during the preprocessing task, most tools may add extra spaces between characters or words where it should not be the case. This does not affect our method because bucketing is used as an abstraction method, i.e. a method used to minimize computation. Gibson uses the same technique to minimize the calculation of poker combinations [9]. Bucketing corresponds to a generalization of the position of the information in the document. In other words, groups of words are stored in different buckets. For instance, instead of measuring the distance between multiple words, we will only measure one distance between two buckets. To minimize computation, each bucket can be represented by a unique set of coordinates corresponding to its midpoint.

Bucketing is critical for performance because financial documents can have hundreds of pages and every pieces of information must be evaluated.

3.3 Clustering

The clustering step agglomerate related buckets into *clusters*, i.e. bags of buckets. Clustering proceeds in 2 steps. The first step is to create small clusters of information by the means of a rectilinear search algorithm. The second step aims to add neighbouring information (buckets) – called satellites – to existing clusters by the means of a measure of distance between information. In the first clustering task, we do not require to semantically verify the content of the clusters because we make the assumptions that perpendicular information for a given value in a table is always sound.

However, we will not retain every bucket of information. In fact, we use a heuristic that eliminates buckets that contain dense blocks of text – mainly paragraphs – effectively drawing a line between information where position is semantically important and where it is not. When a bucket is not retained, it does not mean that no relevant information is present but rather than another IE technique, such as natural language processing (NLP), would be more appropriate. This heuristic fulfills our needs to retrieve information in financial documents where tables, headers, footers, side panels, etc. are of great importance. In order to discard a bucket, we compute, for the line on which the bucket is located, its density, defined as the percentage of non whitespace characters. We then ignore all buckets located on lines where the density exceeds a specific threshold.

3.3.1 *Part 1 – Rectilinear search algorithm*

We then search remaining buckets for numerical values, starting at the very left of the first line, processing line by line, from left to right. Numerical values are identified by the means of regular expressions in order to evaluate their nature despite the presence of other characters (e.g. \$, %, US, etc.). This should be understood as a sanitization function.

The rectilinear search algorithm uses each numerical value of a document as a single pivot point to perform a rectilinear (i.e. horizontal and vertical) search for headers and build elementary clusters of relevant information. Consequently, the algorithm will not manage qualitative data such as *low*, *medium*, *high*, etc. However, it will capture page numbers and other numerical values that are not in a table. These values will be ignored if no headers are found (e.g. if surrounding text is too dense) or if no buckets are semantically related (i.e. if the concept is not relevant to the ontology).

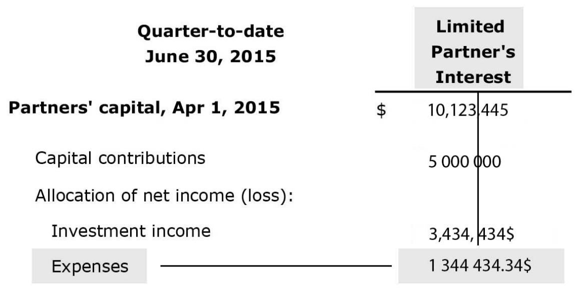

Figure 2: A rectilinear search from a numerical value (highlighted in gray)

The rectilinear search is implemented as follows: according to the first heuristic, grab everything that is in the vertical range of the value – from top to bottom – and everything that is in the horizontal range (i.e. on the same line) from left to right. At this point, clusters are organized in ordered triplets:

num. value | horiz. data | verti. data

For the given example of figure 2, the cluster would be:

1 344 434.34$ | Expenses | Limited Partner's Interest

In financial documents, information is remarkably organized in a perpendicular manner. As shown in [25], most of the information is captured by a rectilinear search (approximately 80%). For tabular information, it is almost trivial that our method will find corresponding headers. However, tables are very complex objects. To manage complex table structures, we identify neighbouring information that cannot be reached by a rectilinear search from numerical values. We call these elements *satellites*.

3.3.2 Part 2 – Identifying satellites

TE techniques aim to recognize tables and their elements. From this point of view, it is very difficult to specify which elements are part or not of a table. Our solution to this problem is to identify additional relevant buckets by considering their position relatively to the buckets that are already in a cluster. We call these neighbouring buckets *satellites* because they gravitate around pertinent information but cannot be reached by the rectilinear search as figure 3 shows.

For instance, in figure 3 the rectilinear search will fail to find either {*Allocation of the net income (loss):*} and {*Quarter-to-date June 30, 2015*}. Nevertheless, these pieces of information are paramount in understanding the table's two last rows. Our algorithm will identify these pieces of information as satellites. For example, {*Allocation of the net income (loss):*} will be associate with {*Investment income*} and {*Expenses*}, and {*Quarter-to-date June 30, 2015*} will be associate with {*Limited Partner's Interest*}.

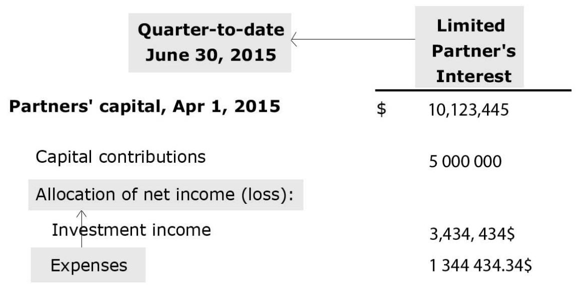

Figure 3: Example of satellite information from identified buckets (highlighted in gray)

The search for satellites relies on a measure of (Euclidian) distance between buckets. More precisely, for all clusters and all buckets of this cluster (i.e. buckets in its horizontal or vertical data), we gather all buckets – that are not associated to a cluster – at a distance less than a specific threshold. Finally, these gathered buckets are evaluated using an ontology and only relevant buckets are kept as the cluster's satellites. The ontology has been developed with financial experts and consists in a list of related words and expressions where relevance between each term is quantified. It contains 2051 expressions and 1989 financial words. For the example of figure 2, the cluster with all satellites would be:

1 344 434.34$ | Expenses | Limited Partner's Interest | Allocation of net income (loss) | Quarter-to-date June 30, 2015

The ontology represents the only evolving component of our general method. All expertise is contain in the ontology. More rules could be add to optimize the process but this will have to be done at the cost of flexibility.

3.4 Optional NER module

In our architecture, the optional NER module aims to extract more contents from the candidate financial documents by identifying and classifying contents into different categories such as names of persons (PER), organizations (ORG), locations (LOC), temporal expressions (DATE), numbers related to MONEY and PERCENT and also FINANCE entities. By the following addition, we want to demonstrate the compatibility of our approach with commonly used information extraction techniques.

We apply the GATE toolkit [5, 6] to automatically annotate our training data. GATE provides a default information extraction system called ANNIE which allows to identify generic concepts such as person names, locations, organisation, dates, etc. We developed new rules and adapted them to our application by creating a JAPE (Java Annotation Patterns Engine) pattern and a company gazetteer. JAPE provides finite state transduction over annotations based on regular expressions.

We have carried out adaptation of the GATE named entity recognition components because most targeted entities were not covered by ANNIE. By using a JAPE pattern, we create *mention* annotations that refer to entity categories. The mention annotations represent classes for the machine learning based training. At this point, a machine learning-based classifier is implemented by using a plugin Batch Learning

PR of GATE with the annotated training data. We choose the PAUM (Perceptron Algorithm with Uneven Margins) algorithm [16] because this algorithm has consistently produced a performance to rival the SVM (Support Vector Machine) [15] with much reduced training times. The classifier has been modelled with 14 classes generated by Batch Learning PR of GATE with the total of 7,200,019 features.

3.5 Structuring

The final structuring task maps the extracted information in the output data model by defining relevant attribute-value couple. This mapping between clusters and the data model is also realized by the means of the domain ontology. Synonyms are defined and priorities between terms correspond to the relevance indices in the ontology. When two terms are indistinguishable in the ontology, the system can give a choice to the user or use other validation techniques. If a financial term can't be found, it is automatically rejected.

4. SUMMARY

Given a financial PDF document D, we extract the textual content while respecting the original layout. This is done with the use of *PDFtoText* tool with the following command line:

```
pdftotext -layout path/to/document/name_of_file.pdf
```

A txt file is generated. From this file, buckets of information are created following the method presented in section 3.2. A measure of density is calculated at this step and is associated to every bucket. Moreover, the midpoint of each bucket is calculated and an identification key is attributed if the bucket contains a numerical value.

The first clustering part uses numerical values as pivots. From each pivot, a rectilinear search is accomplished, according to the method presented in section 3.3.1. For each element, the measure of density must respect the threshold or the element is automatically rejected.

The second clustering part aims to create a neighbourhood around the previous element by aggregating its neighbours (defined by the proximity threshold). The neighbours of a same cluster are semantically compared by the means of a financial ontology. When a relevant financial term is found, the numerical value is imported in the structured output model.

Finally, when an NER module is added, named-entities are incorporated into the clusters in order to facilitate the structuring task.

5. EVALUATION

Our evaluation is based on 600 business documents that have been used in a previous analysis conducted by human financial analysts. At the end of each quarter, fund investors receive a set of financial reports that give a portrait of their investments. There are three types of documents: 1) general financial statement 2) capital account statement and 3) portfolio report. While financial and capital account statements contain mostly tables, portfolio reports are very heterogeneous and integrate graphical elements, tabular information and a large amount of text. For example, a portfolio report will contain textual descriptions of the companies, logos, graphics showing domains of activities, selling points or countries, etc. The three types of document are processed with the same algorithm. The 600 documents that have been use for our evaluation belong to 200 funds (200 funds $\times$ 3 types of documents) from the second quarter (Q2) of year 2015.

Investors receive PDF documents and have no access to the original data warehouse that could have been used by the fund manager. Consequently, for each fund, a financial analyst (human expert) has to manually extract the relevant data. These PDF documents are received quarterly and are directly sent to the financial analysts. Thereby, our financial partner has manually created a database in order to execute statistical analysis and financial predictions. This means that our results are compared with real expert extracted data that have necessitate two validations. It takes approximatively one hour to an analyst to manually manage the data extraction of one fund. For the sake of this evaluation, we consider that the set of data resulting from the work of the expert is fully reliable. The complete corpus corresponds to 786,426 lines and 44,074,525 words. On average, there were 22 numerical values for each page, and 59 pages for one fund. Recall that no table detection task occur and that all lines and characters are processed. That is to say, it is difficult for us to evaluate the number of tables in the corpus. The experiment has been done on a Macbook Pro (2013), Quad core 2.3 GHz Intel I7 with 16 GB DDR3 RAM (1600MHz) and took approximately five hours, compared to the usual 200 hours needed by human experts. This represents thirty seconds per document, which is a considerable time saving compared to the processing by a financial expert.

To measure our performance, we identified two important data to gather and verify on all documents, i.e. the *net asset value* and the *capital account* of a fund. Other values of interest that are extracted by our financial partner and concern the investment in a fund are: ownership, invested capital, cost of investment, total proceeds, fair market value, income, revenu, EBIDTA and net debt. There is no visual distinction between the selected data to extract and other values of interest in our corpus of documents. The *net asset value* and *capital account* of a fund have been selected because they are the two most important data to identify for our financial partner. These values may be located in a header, paragraph, table or footer of any of the three types of document. Moreover, the same information can be found in multiple documents and belongs to different dates of analysis. A script to compare the set of manually extracted data and the set of automatically extracted data has been written in PHP.

The line density threshold value has been defined by *ad hoc* over experiments and has been set to 70% to optimize the results on our corpus. Consequently, all information contained on lines of a density greater than 70% have not been considered. Likewise, the proximity threshold has been fix to an arbitrary value of six ($d = 6$), which means that two elements of a same neighbour can not be separated by more than six distance units, where vertical and horizontal units are respectively represented by the number of lines and the number of characters.

Our results, excluding the NER optional module, for the task of identifying the current *net asset value* and *capital account* of a fund are the following: Precision is 93%, Recall is 90% and F-Measure is 91%. Our result, including the NER module, are the following: Precision is 95%, Recall is

93% and F-Measure is 94%.

Our general method, without any adjustment rules such as those offered by the NER module, approaches those of the handcraft rule-based SmartXBRL system and template-base system, where both scores are around 95%. This is quite impressive as our system only used two heuristics.

These results satisfy the standards of our financial partner. Recall that the industrial goal of our system is to save human resources by automating the extraction process. Consequently, precision is more important than recall because missing data can always be found, in a second step, by human analysts. On the contrary, erroneous data imply a validation which, in the present context, is undesirable.

Finally, we remark that our system allows the flexibility to be combined with further approaches, such as those implemented in the NER module, allowing to reduce the performance gap with complex handcrafted rule-based systems such as SmartXBRL.

6. CONCLUSION

Our method is based on the idea that position of information, in unstructured but visually rich documents – as it is the case for the Portable Document Format (PDF) – is an indicator of semantic relatedness. This assumption led us to make use of two heuristics: 1) a line density threshold that eliminates dense blocks of information – such as paragraphs – and 2) a distance threshold for identifying information that are most inclined to be semantically related.

Our solution has been implemented in a industrial context where it has demonstrated its effectiveness. Our method shows that a small number of semantic relatedness heuristics can yield efficient extractions techniques for the financial field. Furthermore it also offers an architecture in which further methods can be readily integrated.

7. ACKNOWLEDGMENTS

The authors gratefully acknowledge the financial support of Mitacs and the Natural Sciences and Engineering Research Council of Canada (NSERC). We also thank Metix inc. and our financial partner, La Caisse de DÃl'pÃt't et de Placement du QuÃl'bec, to have authorized this publication of an industrial product.

8. REFERENCES

[1] H. Alam, A. Kumar, C. Lee, and Y. Tarnikova. A pattern recognition approach to automated XBRL extraction. In *2012 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)*, 2012.

[2] H. Alam, A. Kumar, C. Lee, and Y. Tarnikova. Automated financial data extraction - an AI approach. In *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2013.

[3] H. Chao and J. Fan. Layout and content extraction for PDF documents. In *Document Analysis Systems VI*, pages 213–224. Springer, 2004.

[4] Y. Cong, A. Kogan, and M. A. Vasarhelyi. Extraction of structure and content from the EDGAR database: A template-based approach. *Journal of Emerging Technologies in Accounting*, 4(1):69–86, 2007.

[5] H. Cunningham, A. Hanbury, and S. Rüger. Scaling up high-value retrieval to medium-volume data. In *Advances in Multidisciplinary Retrieval*, pages 1–5. Springer, 2010.

[6] H. Cunningham, D. Maynard, and K. Bontcheva. *Text processing with GATE.* Gateway Press CA, 2011.

[7] A. C. e Silva, A. Jorge, and L. Torgo. Automatic selection of table areas in documents for information extraction. In *Portuguese Conference on Artificial Intelligence (EPIA'03). Lecture Notes in Artificial Intelligence - 2902. (c) Springer-Verlag. 11th Portuguese Conference on Artificial Intelligence, EPIA, Beja, Portugal*, pages 460–465. Springer-Verlag, December 4-7, 2003.

[8] A. Gabdulkhakova and T. Hassan. Document understanding of graphical content in natively digital PDF documents. In *Proceedings of the 2012 ACM symposium on Document engineering*, pages 137–140. ACM, 2012.

[9] R. Gibson. *Regret minimization in games and the development of champion multiplayer computer poker-playing agents.* PhD thesis, University of Alberta, 2014.

[10] P. A. Griffin. Got information? investor response to form 10-K and form 10-Q EDGAR filings. *Review of Accounting Studies*, 8(4):433–460, 2003.

[11] T. Hassan. Object-level document analysis of PDF files. In *Proceedings of the 9th ACM symposium on Document engineering*, pages 47–55. ACM, 2009.

[12] M. F. Hurst. *The interpretation of tables in texts.* PhD thesis, University of Edinburgh, 2000.

[13] X. I. Inc. Extensible business reporting language (xbrl) 2.1. http://www.xbrl.org/Specification/XBRL-2.1/REC-2003-12-31/XBRL-2.1-REC-2003-12-31+corrected-errata-2013-02-20.html, 2013. [Online; accessed 03-16].

[14] G. Jeh and J. Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543. ACM, 2002.

[15] Y. Li, K. Bontcheva, and H. Cunningham. SVM based learning system for information extraction. In *Deterministic and statistical methods in machine learning*, pages 319–339. Springer, 2005.

[16] Y. Li, H. Zaragoza, R. Herbrich, J. Shawe-Taylor, and J. Kandola. The perceptron algorithm with uneven margins. In *ICML*, volume 2, pages 379–386, 2002.

[17] G. Neumann and F. Xu. Intelligent information extraction. *Proc. of 16th European Summer School in Logic, Language and Information, Université Henri Poincaré Nancy, France*, pages 9–20, 2004.

[18] D. Noonburg. Xpdf. *See http://www.foolabs.com/xpdf*, 2002.

[19] J. P. PDFBox. processing library. *Link: http://www. pdfbox. org*, 2014.

[20] J. Piskorski and R. Yangarber. Information extraction: Past, present and future. In *Multi-source, multilingual information extraction and summarization*, pages

23–49. Springer, 2013.

[21] U. Securities and E. Commission. Edgar. https://www.sec.gov/edgar/aboutedgar.htm, 2010. [Online; accessed 03-16].

[22] M. Sheikh and S. Conlon. A rule-based system to extract financial information. *Journal of Computer Information Systems*, 52(4):10–19, 2012.

[23] P. Thomas, R. Omari, and T. Rowlands. Towards searching amongst tables. In *Proceedings of the 20th Australasian Document Computing Symposium*, page 8. ACM, 2015.

[24] S. Wang, R. Xu, B. Liu, L. Gui, and Y. Zhou. Financial named entity recognition based on conditional random fields and information entropy. In *Machine Learning and Cybernetics (ICMLC), 2014 International Conference on*, volume 2, pages 838–843. IEEE, 2014.

[25] B. Yildiz, K. Kaiser, and S. Miksch. PDF2table: A method to extract table information from PDF files. In *IICAI*, pages 1773–1785, 2005.