Information Extraction through AI techniques: The KIDs use case at CONSOB

Domenico Lembo¹, Alessandra Limosani², Francesca Medda², Alessandra Monaco¹, Federico Maria Scafoglieri¹
¹Sapienza Università di Roma
²Commissione Nazionale per le Società e la Borsa
¹{lembo, monaco, scafoglieri}@diag.uniroma1.it, ²{a.limosani,f.medda}@consob.it

Abstract
In this paper we report on the initial activities carried out within a collaboration between Consob and Sapienza University. We focus on Information Extraction from documents describing financial instruments. We discuss how we automate this task, via both rule-based and machine learning-based methods, and provide our first results.

1 Introduction
In Italy, Commissione Nazionale per le Società e la Borsa (Consob) is the supervisory and regulatory authority of the financial market. Among the several functions, Consob has the role of monitoring and supervising any financial instruments that are issued, with the ultimate aim of detecting and enforcing against illicit conduct. In this project, Consob, rather than outsourcing the digital activity to a specific operator, has taken the decision to develop the project through a collaboration between expertise from Academia and its own expertise.

The objective of such an approach is twofold: first to develop a bottom-up approach driven by the actual needs of supervisory activity; and second, to explore and experiment in innovative ways the improvement of supervisory effectiveness. The results to date show not only the digital improvement but also a direct involvement and participation of the Consob teams and thus effective knowledge spill-overs from one Division to another have taken place.

The specific project we will discuss in the following, besides Consob, involves Sapienza University of Rome, and namely a team from the department of Computer, Control, and Management Engineering, with expertise in Artificial Intelligence and Information Management. To support Consob in its supervision activities, and in particular in the task of verifying the correctness and completeness of the information about financial products, within this collaboration a solution for the automatic extraction of structured information from free text documents is being developed. Through such solution, relevant data contained in the documents are not to be manually identified and extracted, but can be collected through an automated process which make them available in a machine processable format. To this aim, essentially two approaches to automatic Information Extraction (IE) have been adopted: rule-based and machine learning (ML)-based. In the rest of the paper we describe them, explain why both are useful to achieve our goals, and report on the first results we obtained in the project, which have been partially reported, limited to the rule-based approach, in [6].

2 The use case
In the EU, the creators of financial products (a.k.a. financial manufacturers) are obliged by law¹ to make information related to so-called PRIIPs (Packaged Retail Investment and Insurance-based Investments Products) publicly available. The NCAs (National Competent Authorities) have supervisory duties on such products, so that they can be safely placed on the respective national markets. The legislation requires information about PRIIPs to be communicated to NCAs through documents called KIDs (Key Information Documents). In the practice, this means that features to be checked are cast into text reports, typically formatted as pdf files (cf. Figure 1), and extracting structured data from them (to bootstrap control activities), is actually in charge to the authority (In Italy, CONSOB). Due to the massive amount of documents to be analyzed (e.g., ~700,000 KIDs received by CONSOB in 2019, more than 1 million in 2020), this process cannot be carried out manually, but still it is only partially automated to date.

3 The approach
The KIDs present key information about financial products through free text or tables. We decided to extract relevant

¹PRIIPs Regulation n. 1286/2014
data contained in the free text part of the documents through a rule-based mechanism using language that is an evolution of regular expressions [2, 4]. This choice lies in the fact that the KIDs follow a quite rigid template imposed by the European authority, a characteristic that makes domain experts able to express quite precise patterns for the identification of wanted data, and AI experts to encode them into extraction rules.

However, rule-based mechanisms do not lend themselves well to the extraction of data contained in tables, which are particularly difficult to identify in pdf documents, depending also on the way in which the table is originally formatted. For this reason the chosen extraction approach is based on learning systems that treat the pdf as if it were an image and aim to identify the set of pixels that are tables, reconstructing the cells from which the information sought will be identified. In the following we will discuss in more detail the two approaches highlighting their complete pipelines and showing the results obtained.

3.1 Rule-based IE

The rules we defined deal with the extraction of 16 information fields from the free text contained in a KID. Among them we mention the product name, the product ID (a.k.a. ISIN), the manufacturer, the investment risk, and so on.

The rule-based extraction, is organized in three macro modules described in the following, each of them dealing with a specific task.

Data Preparation: The Data Preparation Module transforms the PDF into plain text and clean it from errors. This step is very critical, and if the PDF-to-text transformation produces bad quality results, many errors do actually occur in the subsequent phases. For this reason, we performed it using PDFBox3 by Apache, a state-of-the-art and highly customizable library for working with PDF documents. PDFBox presents a large number of options to manage such transformation, including those for dealing with different types of characters, font normalization, white space management, reading modes, etc. We set them in order to minimize the number of errors incoming from the transformation.

Annotation: The annotation task is carried out through CoreNLP, an open-source JAVA toolkit for natural language processing, whose first release dates back to 2006, and which is still maintained and evolving. This toolkit provides core NLP services accessible through APIs that can be combined to generate a pipeline for text annotation. Among these services, we mention those related to tokenization, lemmatization, POS tagging, all used in our pipeline.

Another core component of CoreNLP is TokensRegex [1], which extends traditional regular expressions on strings by working on tokens instead of characters and defining pattern matching via a stage-based application. These extensions of regular expressions allow for writing extraction rules, i.e., rule-based extractors matching on additional token-level features, such as part-of-speech annotations, named entity tags, and custom annotations.

We introduce the syntax of a rule written in TokensRegex through the following example and refer the reader to [1] for a complete treatment. Consider the following TokensRegex extraction rule useful for extracting International Securities Identification Numbers (ISINs) from (the free text portion of) KIDs:

```latex
\begin{verbatim}
$StartISIN = \\n   /ISIN/ /:/ | \\
   /Codice/ /del/ /Prodotto|prodotto/ /:/ | \\
   ... \\
$EndISIN = ( \\
   /$StartISIN\) (?$CodeISIN \{(word:|$code) \& \\
   \{SECTION:"SECTION_PRODUCT"\})+) (?$EndISIN)
\end{verbatim}
```

The ISIN is an alphanumeric sequence of 12 characters identifying a PRIIP. This sequence begins with two characters, identifying the country code of the product, followed by ten numbers. In our TokensRegex rule, the ISIN format is compiled into the regular expression assigned to $code. Then, $StartISIN and $EndISIN represent the set of sequences of tokens (separated from each other through the symbol \( / \)) preceding or following the ISIN code, respectively. The JSON-like structure at the end of the rule is its main part. In a nutshell, ruleType specifies how the pattern should be used. In our case, the rule works on tokens specified by "tokens" value. pattern contains the pattern to be matched over the text, built over the tokens using groups like the ones in POSIX (but also with names such as $CodeISIN). In our example, our pattern specifies how to find the token such that there is a matching with the regex $code among all the tokens in the product section (i.e. annotated with SECTION_PRODUCT), and such that it is preceded by the tokens in $StartISIN and followed by the ones in $EndISIN. Finally action describes what should happen when the pattern is matched, precisely, what annotation to apply. In our example, we annotate the token identified by the group $CodeISIN with the annotation ISIN.

The result of the application of this rule over the text in Figure 1 is the annotation of CH0524993752 with ISIN.

Exporter: The purpose of the Exporter module is to prepare the output in the desired form. Actually, this module is able to transform the text annotations into two possible formats (i) CSV (Excel) file or (ii) database. In particular, option (ii) allows to arrange data in various formats, and paves the way for the construction of knowledge graphs on top of the extracted data, e.g., via an OBDM approach [3].

3.2 Machine Learning-based IE

We used ML techniques to properly deal with tabular data, and in particular to extract data from the following tables:

- Performance scenarios: It shows the potential refund and the average yield per year in all the possible scenarios: stress, unfavorable, moderate, favorable, each at fixed time periods (initial, intermediate, recommended detention).
• Costs evolution: It includes data about total costs and reduction in yield (RIY) in percentage at the same time periods considered for the performance scenarios.

• Costs composition: It describes the RIY for different costs categories and subcategories: input (una tantum costs), output (una tantum costs), wallet transactions (recurrent costs), others (recurrent costs), performance fees (accessory charges), overperformance fees (accessory charges).

We modeled the problem as a Computer Vision task, converting each page of the KID into images. In such a setting, the Extraction involves a Table Detection task, aiming at detecting bordered or borderless/semi-bordered tables, a Table Structure Recognition task, able to identify the cells belonging to the detected table, and a Text Extraction task, to get the textual information inside each cell.

We propose an automatic, learning-based approach, exploiting the power of Deep Learning and Artificial Neural Networks (ANN) for image processing. In particular, for subtasks 1-2 we use CascadeTabNet [5], a recent implementation of a ANN model that benefits from convolutional layers to learn image features at different level of abstractions, detecting table areas and related cells. The model, that has proved to achieve the state-of-the-art for the considered task, combines two existing CNNs for object detection and image segmentation with a 2-stage transfer learning and image augmentation techniques. In details, HRNet is used as a feature extractor, while Cascade R-CNN predicts 3 classes of objects: (i) bordered table, (ii) borderless table and (iii) cell. In order to adapt the object detection capability of the network to the desired domain (tables inside images), the combined network, initially pre-trained on ImageNet dataset, is retrained on images containing tabular data; the transfer learning is applied firstly on a large, general dataset, allowing the network to learn table masks, and then on a smaller dataset. Depending on the specific dataset used for the transfer learning, different models are developed; we use the Modern Table Structure Recognition model, trained on ICDAR-19.

For subtask 3 instead, we use Tesseract, a popular Optical Character Recognition (OCR) engine, originally developed at HP and later by Google. The original Tesseract engine (‘legacy engine’) was based on multiple steps such as page segmentation, outline and blob detection, textlines breaking, 2-pass word recognition with a trained classifier. From Tesseract version 4, an improved engine is available, fully based on LSTM neural networks, and many new languages are supported.

A complete view of the pipeline of the proposed approach is shown in Figure 2 and its module are:

• A Page Identification Module uses a set of strings for which we should find a match in one of the pages. For each table type (i.e. performance scenarios, costs evolution, costs composition) we select a different set of strings that identify the precise section containing the table we aim to extract.

• The Table Detection and Table Structure Recognition Module uses the pre-trained CascadeTabNet model to identify the masks (i.e. bounding boxes) of borderless/bordered tables and cells, with a certain level of confidence. The masks are returned if and only if their level of confidence is higher than the input threshold, that we experimentally tuned to 0.6. Cells are then assigned to the correspondent table depending on their positions.

• The Table Identification and Row Extraction Module applies the OCR to each cell bounding box, configuring Tesseract with the LSTM engine, the english and italian languages, and assuming a single uniform block of text for page segmentation. Again, a string match identifies the table of interest, since many tables may be present in a single page. If the table is founded, the algorithm returns a set of rows. To assign each cell to its row we use the top coordinate of the cell mask but, since very often the bounding boxes of cells belonging to the same row are not precisely aligned (i.e. one top value is slightly different than the other), we defined an alignment factor such that two cells belong to the same row if their top value differs from at most the alignment factor.

• Rows are arranged into fields in the Row Processing and Cleaning Module, assigning the numerical cells to their field type. A final Cleaning step, fully regex-based, deals with missing punctuations, extra space removal, OCR error correction and, mostly important, standardizes the numerical formats: many countries, indeed, use a comma to separate the integer from the decimal part, while others use a point. Currency units of measure are also removed from the costs, extracting just the numerical information.

The effectiveness of the results often depends on the table template: there are few tables that the algorithm fails to recognize at all, or for which some cells are not detected (mostly
the table headers). Table/cell detection is indeed a challenging task, due to different reasons. Firstly, the heterogeneity of table templates adopted by the PRIIP authors makes the task harder for the neural network; templates are slightly different from the ones in ICDAR-19, because they are modern, often semi-bordered or borderless, semi-coloured, with multi-line cells. The same table type may present different number of rows or columns, just because the PRIIP type requires it, or because of a different arrangement of the information in the table (for instance, a single cell contains both a cost and a RIY). Moreover, we noticed issues in identifying multi-line cells: few times true multi-line cells are detected as two distinct cells or, on the contrary, distinct cells that appear one under the other, probably very close, are detected as a single multi-line cell. We addressed this last case in the Rows Cleaning Module: using our knowledge of the table structures, knowing which field types appear one under the other, we split the wrongly multi-line cells and assign the data to the proper field. Regarding cell masks, we noticed also that, in few cases, the detected bounding box was quite inaccurate, cropping the text and consequently causing troubles in OCR. In order to avoid it, we consider an enlargement factor such that we run the OCR on a slightly larger area. The correctness of the results indeed, depends both on the CascadeTabNet and OCR performances. Our first results showed few inaccuracies in recognizing some characters; as an example, many ‘7’ were recognized as ‘/’, or some digits were recognized as alphabetic characters. Changing the OCR engine type from LSTM to legacy gave no improvement at all. Running experiments on the documents that caused the trouble, we discovered that they mostly share the same table type (performance scenarios), the same author and, therefore, the same template, that consisted of a coloured and saturated background with a white text. Since the latest versions of Tesseract has some troubles in detecting white text on a darker background, we addressed this issue by improving Image Pre-Processing. In particular, for each cell, we checked if it has white text on a coloured background and if so we inverted the image. An important improvement has been obtained moving in this direction, removing more than half of the OCR errors in the troubling documents; the remaining errors are all of the same type (‘7’ as ‘/’). Grayscaling and adding more contrast to the image allowed to further reduce the errors.

4 Results

Table 1 provides some information regarding the two datasets on which we tested our approach.

| Dataset     | Total KIDs | Size   | Manufacturers |
|-------------|------------|--------|---------------|
| DATASET-1   | 1240       | 250MB  | 36            |
| DATASET-2   | 7736       | 3GB    | 52            |

Table 1: Dataset Info

Dataset-1 was created specifically as a test scenario. It is representative of document heterogeneity and includes KIDs from the years 2018-2020. Dataset-2 was selected randomly from a sample of KIDs from the first half of 2021.

Table 2 reports an average of precision and recall over all information extracted through the rule-based approach.

| Dataset     | Precision | Recall | F-Measure |
|-------------|-----------|--------|-----------|
| DATASET-1   | 98%       | 96%    | 97%       |
| DATASET-2   | 98%       | 94%    | 96%       |

Table 2: Results rule-based approach

Table 3 shows the number of tables extracted through our ML-based approach for each table type.

| Performance Scenario | Dataset     | Extracted | Missing |
|----------------------|-------------|-----------|---------|
|                      | DATASET-1   | 1218      | 22      |
|                      | DATASET-2   | 6435      | 1301    |
| Cost Evolution       | Extracted   | 752       | 488     |
|                      | Missing     | 3925      | 3811    |
| Cost Composition     | Extracted   | 1153      | 87      |
|                      | Missing     | 6886      | 52      |

Table 3: Results of the Extractor for the different table types.

We obtained the worst recall for the Costs Evolution table; this is probably due to the fact that the extraction of this table mostly relies on just two header cells in the Table Identification step, therefore, if the model fails for any reason in detecting just those two cell masks, the table is not extracted even though the numerical cell masks are often correctly detected. This issue is less critical for the other table types, since more cells can be used for Table Identification.

5 Conclusion

We are currently working to complete IE from KIDs, by attacking some of their portions that have not been so far involved in the extraction process. The next project steps concern with post-extraction data analysis, aimed to identify anomalous documents, correct errors and missing mandatory data, discover interesting correlations among data.

References

[1] A. X. Chang and C. D. Manning. Tokensregex: Defining cascaded regular expressions over tokens. Stanford University Computer Science Technical Reports. CSTR, 2:2014, 2014.
[2] J. Doleschal, B. Kimelfeld, and W. Martens. Database principles and challenges in text analysis. ACM SIGMOD Record, 50(2):6–17, 2021.
[3] D. Lembo, Y. Li, L. Popa, K. Qian, and F. Scafoglieri. Ontology mediated information extraction with MASTRO SYSTEM-T. In ISWC (Demos/Industry), pages 256–261, 2020.
[4] D. Lembo and F. M. Scafoglieri. Ontology-based document spanning systems for information extraction. Int. J. of Semantic Computing, 14(01):3–26, 2020.
[5] D. Prasad, A. Gadpal, K. Kapadni, M. Visave, and K. Sulthanpure. Cascadetabnet: An approach for end to end table detection and structure recognition from image-based documents, 2020.
[6] F. M. Scafoglieri, D. Lembo, A. Limosani, F. Medda, and M. Lenziner. Boosting information extraction through semantic technologies: The kids use case at consob. In ISWC (Demos/Industry), 2021.