Extracting actionable data from structured documents with Amazon Textract, AWS Lambda e Amazon S3

7 July 2023 - 10 min. read

Amazon S3 *Amazon Textract* *AWS Lambda*

In the digital age, effectively processing and managing large quantities of documents is a priority for companies in every sector. Many organizations are faced with the task of digitizing large volumes of paper documents or processing data from structured documents, such as invoices or contracts, automatically and with minimum human intervention. In this context, **Optical Character Recognition (OCR)** has proved to be an indispensable tool for automating processes and improving overall efficiency.

However, being able to extract text from a document is only part of what most applications need; it can be considered a primitive function. The goal is more often to **extract specific information**, selecting text based on the structure of the document.

To correctly select valuable information, is crucial to obtain details on the structure of the document itself, such as how the text is grouped, tabulated, or its position within the page.

Finding answers to these questions is exactly where **Amazon Textract** excels.

In addition to providing the ability to extract text from documents, Amazon Textract can identify and return page structure information, opening up a wide range of data processing possibilities. Unlike traditional OCR software, which requires manual configurations and continuous updates to adapt to changes in forms, Amazon Textract

uses machine learning models to process any type of document, ensuring accurate extraction of text, handwriting, tables, and other data without any manual intervention.

Let's move on to the description of a use case.

Extracting information from an invoice

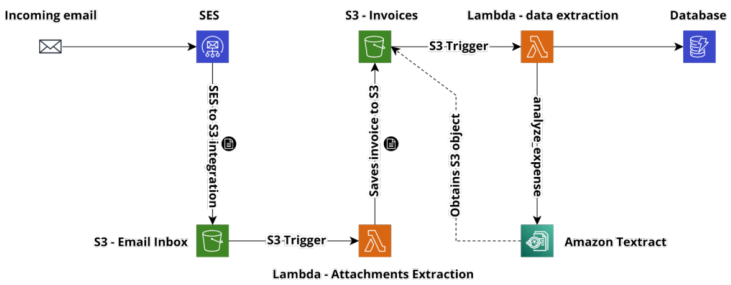
To explore the potential of Amazon Textract we will make use of a (not too much) hypothetical case in which the need is to automatically extract some information from company purchase invoices in order to enter the amounts and the date in a database that is periodically loaded into the financial management software.

Therefore, we need to build an automatic system capable of extracting the amount and date from the invoices it receives. For simplicity, let's assume that all the invoices have the same structure because they come from the supplier's site from which our company obtains its supplies of consumer goods, although Textract can easily analyze heterogeneous invoices.

Invoices are PDF documents meant to be read by a human. They contain headers, the vendor logo image, text, and tables in different places on the page.

The site sends an invoice via email for each order. In our scenario, the address refers to an email group; therefore, we can ensure that the automatic system receives a copy without affecting the processes involving our operators.

In this situation, we could sketch the following high-level solution



By subscribing an ad-hoc email address to the email group we can route a copy of the emails containing invoices to our system.

To receive emails and process them, we can exploit Amazon SES; although it is commonly known for sending emails in the AWS environment, it can also be used for

receiving and consequently integrating with AWS services useful for email processing.

Receiving emails supports various integrations, but in our case, the most practical one is certainly the one with Amazon S3.

Through the integration, Amazon SES saves an object containing the raw data of the mail in MIME format on the designated Amazon S3 bucket. This allows us to track - on the AWS side - a history of the raw body of each message received. This is useful for both archiving and investigating any malfunctions.

The use of Amazon S3 storage space comes with minimal costs, and in the event of very large quantities of messages received, it is possible to optimize the cost further by taking advantage of Lifecycle policies, low-cost storage classes, and all Amazon S3's functions.

At this point, an Amazon S3 trigger comes into operation, starting a Lambda Function that takes care of parsing the body of the email and extracting the attachment. The file can then be saved to a dedicated S3 bucket.

In this article, we will not explore the code necessary to extract the attachments because it is not the core of what we intend to cover. However, libraries exist for most of the popular languages that make parsing and manipulating MIME data easier. For example, this is a NodeJs-based library from which to build the function.

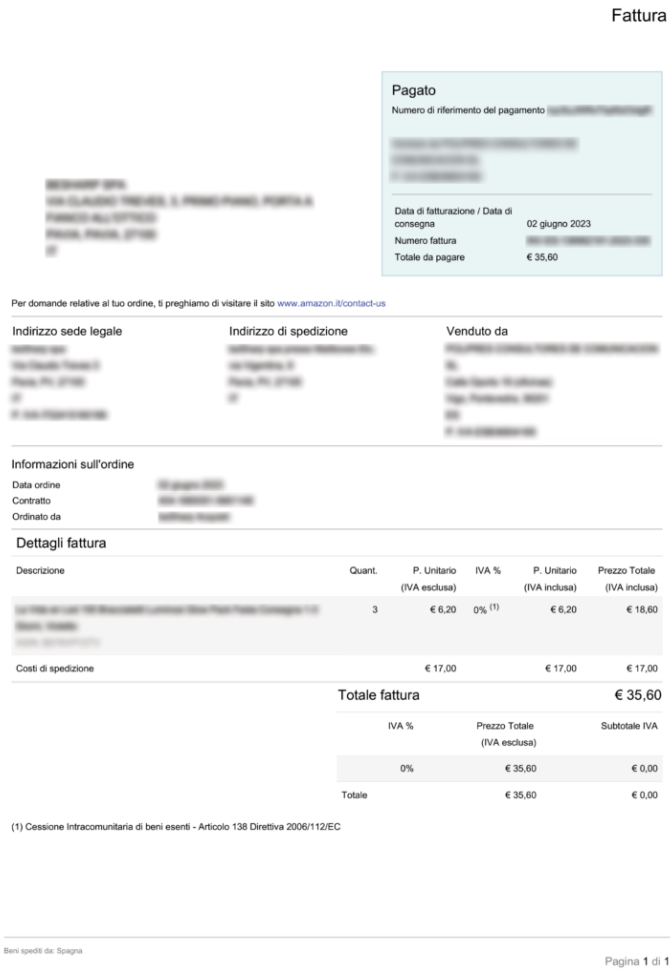
At this point, if the invoice is saved in one of the formats supported by Amazon Textract, it is possible to proceed with extracting the information. Otherwise, it is possible to add a further function or extend the one that manipulates the email to convert to a universal format, such as PDF, for example.

A second Amazon S3 trigger starts a Lambda function used to invoke Amazon Texttract, passing it as input to the object to be analyzed. The lambda will then navigate the result of the analysis to detect the total and issue date of the invoice and save the information involved in the database.

The integration with Amazon Textract is pretty seamless, and the methods exist in all the AWS SDKs for major programming languages, such as boto3 for Python and the AWS SDK for JavaScript.

The reference invoice

In our scenario, we will use invoices generated by Amazon Business, which look something like this:



Fattura

Pagato

Numero di riferimento del pagamento

Data di fatturazione / Data di consegna 02 giugno 2023

Numero fattura

Totale da pagare € 35,60

Per domande relative al tuo ordine, ti preghiamo di visitare il sito www.amazon.it/contact-us

Indirizzo sede legale | Indirizzo di spedizione | Venduto da

Informazioni sull'ordine

Data ordine

Contratto

Ordinato da

Dettagli fattura

Descrizione	Quant.	P. Unitario (IVA esclusa)	IVA %	P. Unitario (IVA inclusa)	Prezzo Totale (IVA inclusa)
	3	€ 6,20	0% (1)	€ 6,20	€ 18,60
Costi di spedizione		€ 17,00		€ 17,00	€ 17,00

Totale fattura € 35,60

IVA %	Prezzo Totale (IVA esclusa)	Subtotale IVA
0%	€ 35,60	€ 0,00
Totale	€ 35,60	€ 0,00

(1) Cessione Intracomunitaria di beni esenti - Articolo 138 Direttiva 2006/112/EC

Beni spediti da: Spagna

Pagina 1 di 1

The document presents the total both outside the table - in the header detail box - and at the bottom of the table.

This is an aspect to consider since we will be able to select the easiest total to identify with Amazon Textract, or build specific logic to identify the total by searching both inside and outside the table or to pick the one with the greatest confidence choosing among two results.

Textract API

The service provides various API calls, both synchronous and asynchronous.

Synchronous versions allow you to send a document for immediate analysis, and the response to the API call is the result of the analysis itself. These API calls have important limitations on the accepted formats and on the fact that the caller must necessarily wait for the end of the analysis, which can take several seconds.

The asynchronous versions, on the other hand, immediately return a JOBID, through which it is possible to request the result of the analysis at a later time. There is also a mechanism to obtain notification of the successful analysis via SNS. See this content for details. In summary, it is possible to specify the SNS topic when starting the asynchronous analysis.

Remember to carefully consider the use of the asynchronous version, especially if the computation is based on Lambda. This allows for better decouple the infrastructural components, building robust retry mechanisms, increasing the overall high reliability of the solution, and reducing costs by eliminating the payment of the Lambda computation time occupied by the synchronous waiting for the analysis result.

At the time of writing this article, the available API calls are as follows:

- analyze-document
- analyze-expense
- analyze-id
- detect-document-text
- get-document-analysis
- get-document-text-detection
- get-expense-analysis
- get-lending-analysis
- get-lending-analysis-summary
- start-document-analysis
- start-document-text-detection
- start-expense-analysis
- start-lending-analysis

For our case, we can take advantage of the function **analyze-expenses**, which uses a specially trained model to recognize and extract financial data from documents such as invoices.

Navigate the Textract response

The Textract analysis response can be extremely verbose and contains a lot of information about the discovered parts of the document.

As you can see from the image above, an average invoice may produce a response of over 34,000 lines.

The response of the function **analyze-expenses** consists of 3 macroblocks:

1. SummaryFields
2. LineItemGroups
3. Blocks

In the **SummaryFields** section, all the information that Textract finds outside a table is collected and labeled.

In the **LineItemGroups** section, on the other hand, all the information present inside the tables is collected and enumerated line by line.

In the **Blocks** section, it is possible to find all the text blocks found by Textract, with information on the bounding boxes and the position within the page. This section is the one that contains the lowest level data and is the section that would have been returned by executing the generic analyze-document function.

For the invoices we are examining, we will use the data extrapolated from SummaryFields, where we can find references to amounts and dates already labeled by type.

This is a fragment of the first element of the SummaryFields vector, the call is related to the analysis of an Amazon Business invoice.


```
{
    "ExpenseIndex": 1,
    "SummaryFields": [
        {
            "Type": {
                "Text": "ADDRESS",
                "Confidence": 59.766048431396484
            },
            "LabelDetection": {
                "Text": "Sold by",
                "Geometry": {
                    "BoundingBox": {
                        "Width": 0.08296574652194977,
                        "Height": 0.008674301207065582,
                        "Left": 0.6447910070419312,
                        "Top": 0.3040856122970581
                    },
                    "Polygon": [
                        {
                            "X": 0.6447910070419312,
                            "AND": 0.30408763885498047
                        } … ,
                    ]
                },
                "Confidence": 95.342376708984375
            },
            "ValueDetection": {
                "Text": "Amazon EU S. r.l. Italian Branch\nViale Mont
e Grappa 3/5\n20124 Milan\nItaly",
```

```
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.21900421380996704,
                    "Height": 0.055659566074609756,
                    "Left": 0.6449917554855347,
                    "Top": 0.3213878870010376
                },
                "Polygon": [
                    {
                        "X": 0.6449917554855347,
                        "AND": 0.32139283418655396
                    } …,                    ]
            },
            "Confidence": 56.650238037109375
        },
        "PageNumber": 1,
        "GroupProperties": [
            {
                "Types": [
                    "RECEIVER"
                ],
                "Id": "42d41651-ba4a-4fc5-97ce-f9b34d1d987d"
            }
        ]
    }
```


Some details on the geometries have been cut from the JSON for brevity since they are irrelevant to the purpose of our discussion.

As you can see, there is a lot of data available in the text fragment. For example, in the Type field, it is possible to know that the captured fragment contains an address.

Below, the LabelDetection object tells us that the label associated with the address is "Sold by". By continuing to scroll, skipping all the information on the position and on the bounding boxes, we can get to the ValueDetection section, which contains the correct value of the address of the seller.

To find the information we need, all we have to do is scroll through the array of SummaryFields objects and look for the string "TOTAL" in the Type field.

Textract will find more totals, at least on the sample document, because the figure is reported multiple times, and there are subtotals. In this specific case, **the safest field to extract is the one identified with the TOTAL type, which also has the greatest confidence.** Further proof can be obtained by analyzing the LabelDetection field, which will contain the text describing the field, in our case, "Invoice total"


```
"Type": {
        "Text": "TOTAL",
        "Confidence": 96.169677734375
    },
    "LabelDetection": {
        "Text": "Invoice total",
        "Geometry": {
            "BoundingBox": {
                "Width": 0.10979866981506348,
                "Height": 0.010328351520001888,
                "Left": 0.49856579303741455,
                "Top": 0.6540638208389282
            },
            "Polygon": [
                {
                    "X": 0.49856579303741455,
                    "AND": 0.6540638208389282
                }...,
            ]
        },
        "Confidence": 95.28824615478516
    },
    "ValueDetection": {
        "Text": "64,99",
        "Geometry": {
            "BoundingBox": {
                "Width": 0.06178450211882591,
```

```
                "Height": 0.011797532439231873,
                "Left": 0.8738263249397278,
                "Top": 0.6542784571647644
            },
            "Polygon": [
                {
                    "X": 0.8738263249397278,
                    "AND": 0.6542784571647644
                },...,
            ]
        },
        "Confidence": 64.54907989501953
    },
    "PageNumber": 1,
    "Currency": {
        "Code": "EUR"
    }
}
```


The value identified in the ValueDetection section is the correct total of 64.99 Euros.

In the same way, we can also find the issue date and, if available, the expiration date.

Some useful types that Amazon Textract can identify are:

- TOTAL
- AMOUNT_DUE
- VENDOR_ADDRESS
- VENDOR_NAME
- RECEIVER_NAME
- SUBTOTAL
- OTHER

Using OTHER and the text identified by Textract as a label, it is possible to extract custom fields.

Certainly, using the Type field to find the relevant values makes the solution capable of interpreting invoices with a heterogeneous structure and takes full advantage of the specialized ML model that Amazon has created. However, if the Textract labeling is not precise enough for specific purposes, it is possible to ignore the type field and proceed by scrolling through the identified Label value pairs, looking for keyword labels to identify the values of interest.

That said, **a single API call is enough to get all the data in the document**.

The most tedious part is to do some tests to understand which information is correctly classified and for which, instead, it is necessary to implement a recognition based on the labels or on a combination of labels and recognized type.

It is obviously necessary to implement **a solid fallback logic** in case not all the information is found or the confidence is below a definable threshold to mark the document for manual analysis.

Conclusions

In this article, we have seen how Amazon Textract can simplify the process of extracting the relevant data from an invoice document without requiring manual configuration or extensive knowledge of the document format.

We have also illustrated one of the main features of the Textract API, which allows you to analyze invoices and financial documents quickly and accurately, returning the data in a structured and easily usable format.

This solution can be applied to different scenarios and it is possible to customize the integrations to handle varied scenarios.

What's the best application for Amazon Textract according to your experience? We're excited to hear your own opinion!

Stay tuned for more articles on the AI and ML world. See you in 14 days on Proud2beCloud!

About Proud2beCloud

Proud2beCloud is a blog by beSharp, an Italian APN Premier Consulting Partner expert in designing, implementing, and managing complex Cloud infrastructures and

advanced services on AWS. Before being writers, we are Cloud Experts working daily with AWS services since 2007. We are hungry readers, innovative builders, and gem-seekers. On Proud2beCloud, we regularly share our best AWS pro tips, configuration insights, in-depth news, tips&tricks, how-tos, and many other resources. Take part in the discussion!

Alessio Gandini

Cloud-native Development Line Manager @ beSharp, DevOps Engineer and AWS expert.Since I was still in the Alfa version, I'm a computer geek, a computer science-addicted, and passionate about electronics all-around.At this moment, I'm hanging out in the IoT world, also exploring the voice user experience, trying to do amazing (Io)Things.Passionate about cinema and great TV series consumer, Sunday videogamer

---