TREXIN INSIGHT PAPER

November 5, 2024

MAKING AN IMPACT WITH LARGE LANGUAGE MODELS

A thorough exploration of a recent Client Case Study.

Most of us have been intrigued by the power of Large Language Models (LLMs) and by now, we know it is not a panacea for our problems. However, when applied carefully there are certain business processes where LLMs and associated Generative AI (GenAI) technologies can make a real impact with a clear return on investment (ROI) that is measurable, reliable, and compelling.

BACKGROUND

In my role as Chief Technology and Innovation Officer at Trexin I tend to focus on what is *realistically possible* without overspending, by *not* using a crystal ball. In other words, what can we do now with the available state of a technology, and yet pushing the boundaries just a bit.

A DEEPER LOOK INTO A RECENT CASE STUDY

One of our Fortune 100 Clients spends a couple of million USD each year to have information from mostly handwritten forms converted into digital text through data entry. And while I cannot show you the forms, I can tell you that they are highly complex and dense formats with many value pairs, open questions, tables, multiple pages, and a very unstructured layout in general. There are 100's of different formats and many different languages involved as well.

Over the years our Client had several discussions with their technology partners, but the result was always more a shift in cost and labor, from reducing data entry cost done by relatively inexpensive resources, to a setup by higher priced individuals to manage the myriads of document layouts, and model training. And more importantly, none of the approaches matched their goal which was to process these documents with minimal human involvement.

Before I talk more about our offered solution approach – I will say this – during our brief testing period we found that mainstream LLMs are pretty good at recognizing handwritten text but were also a bit disappointed with the actual quality of the output. While I was not expecting a 100% success rate, some models performed much better than others and it did provide more data points to suggest our approach makes sense in this context.

We tested:

- Cloud-based Optical Character Recognition (OCR) services such as Google Vision, Azure AI Vision, and Amazon Textract, which run directly on cloud platforms without modifications nor training
- LLM services like Google Gemini and OpenAI
- Open-source frameworks like TesseractOCR and EasyOCR

Again, we tested these with a low touch approach in mind, so no training or other adjustments performed on the models and technology. Text was handwritten by a US individual. There was a high variability of form questions and layouts. The forms were complicated such as having multiple pages and not all pages were straight (fax) or photo based. We also used a small sample size.

Summary Table

Technology	Product	Cost Mode	Ranking Based on Output (Poor, So-So, Ok, Good)
OCR Cloud Services			
	Google Vision	Paid Service	Ok
	Azure AI Vision	Paid Service	So-So
	Amazon Textract	Paid Service	Good
Large Language Models			
	Google Gemini	Paid Service	So-So
	OpenAI	Paid Service	Good
Open-Source Frameworks			
	TesseractOCR	Free	So-So
	EasyOCR	Free	Poor

Another element that will support the approach we envisioned is that when we asked OpenAI to analyze a document and return the key-value pairs of all the handwritten answers to the question, it was correct about 99% of the time. Therefore, it would allow to act as a central distributor or handoff coordinator to other LLMs, and manage the JavaScript Object Notation (JSON) outputs to ensure confidence levels maintained high.

On the other hand, OCR Cloud Services do not return a structured key-value pair output. Additional processing is required to organize the output into a proper format, and here you enter the realm of string manipulation, REGEX, etc. It would not deliver on the Client's objective to minimize human involvement. It would also be dreadful to maintain all variations to build value pairs in a consistent and reliable manner.

In our proposed solution approach, we have focused the application of technology in the area surrounded by the dotted rounded rectangle (see Figure 1: Document Processing Model below). Data Capture, Document Classification, and Data Extraction are the areas that traditionally need a lot of training, and adjustments to make document data extraction successful. This style of approach still has merit if your documents do not change all that much and have a limited number of document types.

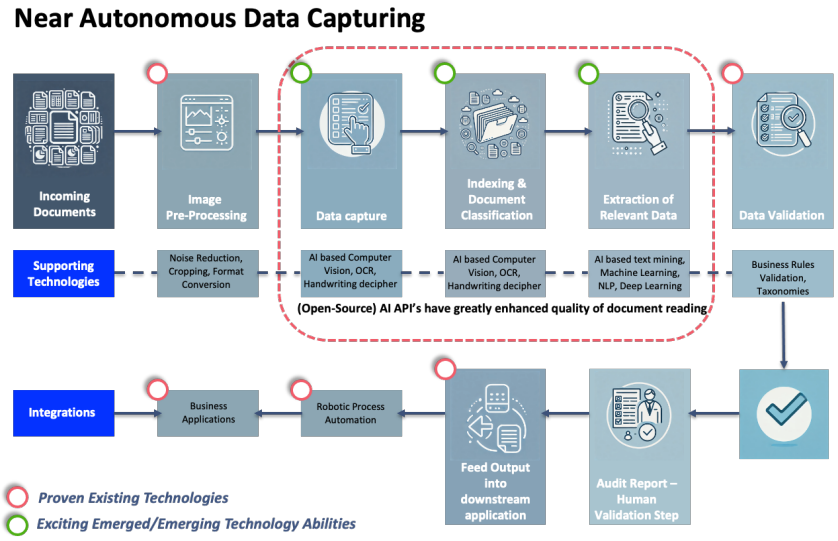

Figure 1: Document Processing Model

Now, what does the solution approach look like?

Let's start with some of the architecture design principles:

- Not based on one LLM model therefore better cost, and quality control that offers optimal control over spend
- Self-learning is key, while this is a big word, it needs to keep track, and report on what LLMs are providing the best value in terms of quality of output to be autonomous
- Minimal human intervention needed
- No training of document formats and key-value pairs
- Autonomous operation, and here we mean that the solution understands what document is in front of it, and can successfully identify its value pair structure, and assign one or more LLMs to extract its data

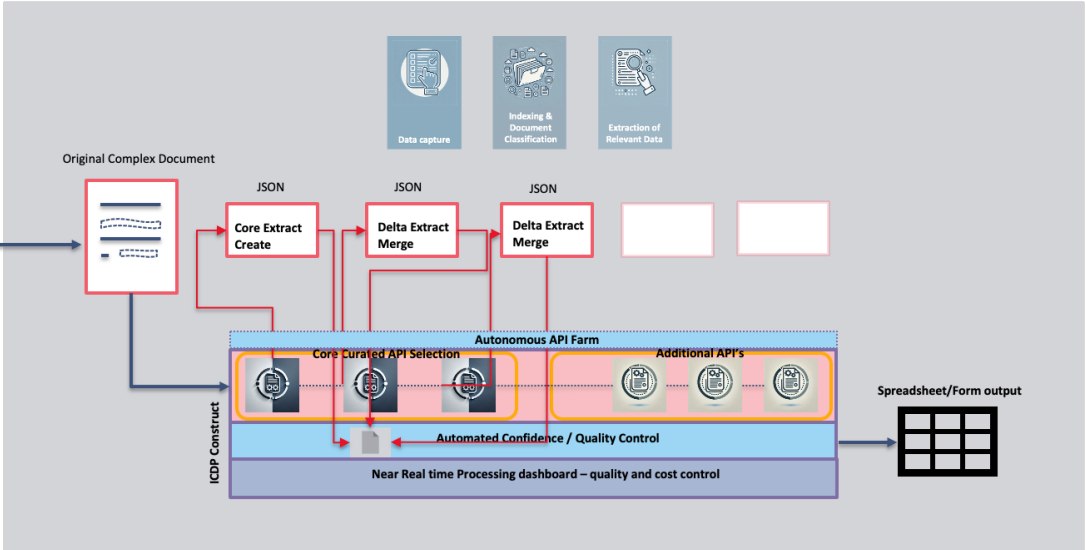

Figure 2: Trexin's Autonomous Extraction Model

The core functions of the solution are:

- Understand a document, and create reliable value pairs
- Autonomous API farm selects one or more LLMs based on prior processing results
- Processing starts and where confidence level is low and/or prior processing results dictate more scrutiny, the document is then handed off to another LLM
- Recording of results, as well as cost of processing will be recorded
- Output provided in Excel, and sent to an RPA solution for further processing

The iterative nature of the solution is key here – LLMs all have strengths and weaknesses, and the true value of this approach is the ability to understand where another pair of eyes (in this case another LLM) is needed to increase output quality and limit human intervention.

Ultimately with this model, there are other areas to experiment with, especially around processing cost (electricity consumption) of the LLMs. You might find you do not need super large LLMs to process these documents at the quality you desire.

For instance, the Allen Institute for Artificial Intelligence (Ai2) has developed "Molmo," a family of smaller multimodal AI models competitive with state-of-the-art models like GPT-4o. Molmo models use around 1,000 times less data by prioritizing data quality. The models, ranging from 1 billion to 72 billion parameters, perform well across academic benchmarks. Molmo is also open source, offering free alternatives to proprietary AI models with fewer commercial restrictions (see Figure 3: Small MultiModal AI Molmo below).

Model Name	Parameters (in billions)	Data Size	Key Feature	Open-Source License	Performance Benchmarks
Molmo 72B	72	700K images, 1.3M captions	Multimodal (text and images)	Apache 2.0	Matches/outperforms GPT-4o, Claude 3.5, Google Gemini
Molmo 7B	7	700K images, 1.3M captions	Multimodal (text and images)	Apache 2.0	Competitive with leading models
GPT-4o	~1,000	Vast (exact unknown)	Multimodal (text and images)	Proprietary	State-of-the-art
LLaMA 3.1	405	Large (exact unknown)	Text-only	Open-weights	Competitive but limited commercial use

Figure 3: Small MultiModal AI Molmo

Power consumption (see Figure 4: LLM Power Consumption Sample below) is an important factor to consider, as it accumulates over time, particularly with larger models. High-performance language models, especially the more extensive ones, have significant power demands with estimates indicating requirements of 200W to 500W for advanced GPUs during inference. For context, cloud providers such as AWS and Google Cloud charge a range of $0.0001 to $0.01 per 1,000 inferences, depending on the model's complexity. This illustrates the cost implications of scaling up inference usage and the need to assess power usage as a critical component of overall deployment costs.

Model Name	Parameters (in billions)	Power Consumption (W)	Cost per Call Estimate ($)
Molmo 72B	72	~300-350W	$0.005
Molmo 7B	7	~200W	$0.0015
GPT-4o	~1,000	~500W+	$0.01
LLaMA 3.1	405	~400W	$0.0075

Figure 4: LLM Power Consumption Sample

CONCLUSION

The proposed solution for near autonomous document data extraction will ensure full control over cost, LLM selection based on the required quality levels, including confidence levels and with minimal human involvement and intervention. The incremental improvement of confidence levels by using multiple LLMs controlled by a central coordinator of sorts will provide the Client an increased control over what models to add or remove from its processing engine. I also believe that this concept will have merit for many other use cases. Of note is that some components of the solution have been tested in a sandbox, yet others are conceptual in nature.

CONTACT US TODAY

Want to learn more about the results of the text recognition scans? Contact a Trexin Advisor today or reach me directly using the email listed below.

This TIP was written by Ton Roelandse. Ton welcomes comments and discussion on this topic and can be reached at ton.roelandse@trexin.com.