Privacy-Preserving AI for Document Understanding with Controlled Unclassified Information

Scott M. Sawyer
Paperless Parts, Inc., Boston, MA, USA, scott@paperlessparts.com

***Abstract*—In manufacturing, manual exchange and review of engineering drawings constrains the supply chain. AI techniques have the potential to streamline these processes. This paper presents two important use cases and proposes solutions powered by generative AI models, including text and multi-modal Large Language Models. The solutions preserve the privacy of documents used for training and inference and are thus suitable for controlled data. Performance is measured and discussed in the context of the applications, including technical and other practical considerations. Results indicate generative AI is comparable or better than previous solutions.**

I. Introduction

The global manufacturing supply chain is driven largely by person-to-person communication in the context of technical data describing parts to be manufactured. Parts are commonly described by 3D Computer Aided Design (CAD) models and 2D engineering drawings. 3D models capture the geometry of the part and, when applicable, how components of an assembly fit together. 2D drawings, as well as other specification documents, provide important product manufacturing information. Drawings commonly include metadata, notes, special instructions, information about materials and finishes, weldments, hardware, quality and other requirements, and callouts for dimensions and tolerances, all of which are required to fully specify a part to be manufactured. While there are some standards and best practices used widely, drawings vary considerably based on the industry, region, and part designer. Drawings are exchanged in file formats like PDF, DXF, and other image formats. These formats ensure the documents are rendered and printed consistently across devices and technologies. However, they do not generally structure the information to make it easily machine readable. Today these drawings must be reviewed by human experts.

Supply chain digitization has been underway for years, with many categories of software solutions being adopted by the industry, such as Computer Aided Design, Manufacturing and Engineering (CAD/CAM/CAE); Part Lifecycle Management (PLM); Enterprise Resource Planning (ERP); Manufacturing Execution Systems (MES); Quality Management Systems (QMS); and more. Technical part data are created, processed, or used by many of these systems. A hypothetical digital thread of part information facilitates communication between these systems. In practice, the digital thread tends to consist of the same 3D models and 2D drawings that specify part designs, along with artifacts created through the supply chain by these software tools. There is no definitive standard for this technical information. Rather, there are dozens of proprietary and open data formats for various use cases. Furthermore, commercial communication, like solicitations, quotations, and orders, is driven primarily by email and personal interaction.

An additional challenge, especially in the United States where an estimated 10% of all manufacturing serves the defense industry [1], is evolving cybersecurity regulations aimed at protecting the sensitive technical data marked as Controlled Unclassified Information (CUI). CUI is a broad class of information that includes technical data describing defense articles. Cybersecurity Maturity Model Certification (CMMC) is rolling out imminently to the Defense Industrial Base (DIB), requiring thousands of manufacturers to implement the controls described by NIST SP 800-171 [2] and obtain a third-party assessment and certification. CMMC imposes additional requirements on the use of Cloud Service Providers (CSPs) that store, transmit, or process covered data. Most notably, CSPs must be authorized through the Federal Risk and Authorization Management Program (FedRAMP) at the Moderate baseline or have implemented the equivalent security requirements [3]. Based on NIST SP 800-53 [4], this is a stringent cybersecurity baseline, typically implemented by cloud service offerings sold broadly across the federal government. FedRAMP authorized or equivalent cloud services are not common in industry-specific manufacturing software.

Given the variety of types and formats of data, along with the current need for human review in industries facing labor shortages, Artificial Intelligence (AI) is a potential solution for further digitization and automation in the supply chain. Generative AI is particularly exciting for its potential to enable new modes of human-computer interaction that complement existing business practices, like requesting quotations and reviewing technical drawings. State-of-the-art multi-modal Large Language Models (LLMs) are now readily available via cloud-based APIs. However, cloud-based AI services present cybersecurity compliance challenges for manufacturers handling CUI. AI operating on CUI must either run on on-premises infrastructure subject to CMMC requirements, or run in a cloud service meeting FedRAMP Moderate security requirements. Furthermore, the AI must not train on or learn from CUI in such a way that CUI could be disclosed to an unauthorized recipient. This problem space and regulatory landscape presents a unique opportunity for domain-specific AI models operating in a FedRAMP environment.

Generative AI models, such as chatbots, are trained on vast corpora of data and are designed to generate novel outputs from given prompts. While measures are taken to

prevent direct plagiarism, the generated content is inherently influenced by the training data in complex and unpredictable ways. This poses a significant risk when using sensitive data, such as CUI, for training, as there is no foolproof way to ensure that sensitive training data will not be included in responses to other users.

In contrast, using LLMs as an enabling technology for *detection* algorithms—such as Named Entity Recognition (NER) or token classification—largely mitigates this risk. In detection tasks, the model's output in response to a query document is strictly a subset of the content within that document. This inherent characteristic of detection problems provides a natural safeguard against data leakage, ensuring that sensitive information from the training data is not exposed through the model's responses.

In this paper, two use cases for natural language detection problems are presented, and solutions involving LLMs are proposed. Key parameters affecting performance are identified and explored. The effectiveness of generative AI as a solution to the problems is analyzed and compared with a previously developed solution.

II. Related Work

AI for document understanding is an active area of research with significant advancements in recent years. One prominent development is Microsoft's LayoutLM, a series of multi-modal large language models (LLMs) designed to understand the structure and layout of document pages to better comprehend their content. Three open-source versions of LayoutLM have been released [5] [6] [7], each offering improvements and new capabilities for tasks such as document question answering (Q&A), token classification, and document classification.

An alternative approach is presented by Donut [8], which does not rely on the availability of accurate document text, for instance, from Optical Character Recognition (OCR). This method is particularly useful for business documents like invoices and receipts, but it has not been tailored to address the unique challenges posed by manufacturing-specific documents, such as engineering drawings.

In the context of engineering drawings, Villena Toro, et al. [9] have developed a system for understanding these documents for quality control applications, with a specific focus on Geometric Dimensioning and Tolerancing (GD&T). Their approach involves training an OCR model on synthetic data to detect GD&T symbols embedded within the text, followed by the application of complex algorithms to interpret the semantics of the GD&T callouts. A number of proprietary QMS products, like HighQA [10], 1factory [11], and Werk24 [12], offer some information extraction capabilities from technical drawings, but these solutions have not disclosed their techniques or performance metrics enabling independent evaluation.

For AI model development and deployment, Paperless Parts has adopted Amazon SageMaker as a comprehensive platform for machine learning (ML) and ML operations [13]. SageMaker is available in the Amazon Web Services (AWS) FedRAMP-authorized GovCloud region, and the service is included in the authorization, ensuring that SageMaker can be used within a FedRAMP environment, provided that the appropriate architecture and controls are implemented [14]. This capability is particularly relevant for Paperless Parts, enabling the development and hosting of AI solutions where training and inference data, including CUI, remain within the secure FedRAMP boundary.

Another related research area, Retrieval-Augmented Generation (RAG), is a technique that enhances the performance of generative models by retrieving relevant documents from a large corpus and using this information to generate more accurate and contextually appropriate responses [15]. While RAG has been adopted in many applications by leveraging extensive public datasets, it is not a suitable fit for our problem. This is primarily because there is no large public corpus of technical documents that would benefit all users of our system. While individual users might have smaller, relevant corpora, our system does not currently have access to these documents. Consequently, integrating RAG would conflict with the need to ensure that sensitive technical data remains secure and private.

III. Use Case 1: Document Metadata Detection

A. Problem Statement

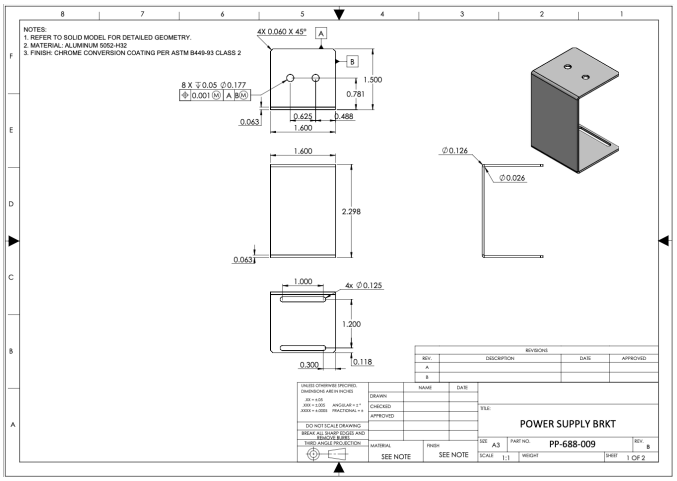


Fig. 1: While engineering drawings vary significantly across the industry, they typically contain basic metadata defining the part, often in a title block located on an edge or corner of each page.

Engineering drawings contain critical specifications that manufacturers must review and with which parts must comply. Failure to understand and adhere to the full specification can result in parts being made out of specification, which can have negative business and mission impact. Today, drawings are primarily reviewed manually because their format is inherently difficult for software to understand and because drawings vary significantly across the enormous manufacturing industry. Fig. 1 shows an example of a typical drawing. Paperless Parts has a dataset of real engineering drawings with manually

labeled metadata. This dataset is not publicly available, but it provides a rich source of real-world samples. This use case focuses on extracting certain pieces of metadata commonly found in drawings: part number, drawing number, revision, and description. This metadata is important everywhere in the supply chain, as it defines a drawing's role in the digital thread. We will define the targeted metadata elements as follows, along with their frequency of occurrence in a sample of 291 drawings held out for testing:

- *drawing number*: unique identifier for the drawing itself, which may contain specifications for any number of parts (occurs in 68% of sampled drawings)
- *part number*: unique identifier for the part depicted, when the drawing is associated with a single part (occurs in 35% of sampled drawings)
- *revision*: the revision of the part or drawing, typically a number or single letter (occurs in 86% of sampled drawings)
- *description*: also known as title, this is a short, natural language name for the part or group of parts described in the drawing (occurs in 97% of sampled drawings)

The dataset has human labels that can be used as ground truth. These labels are provided by users when drawings are associated with quote items within the Paperless Parts application. They are associated with the drawing file but not any particular text string contained within that document. Therefore, training with this dataset requires an unsupervised means of assigning the labels to a portion of the document text. In the case of drawing number, part number, and revision, a data labeling algorithm looks for the target string in close proximity to an appropriate label (e.g., "DWG NO." for drawing number, or "Rev" for revision). Descriptions are not always labeled (e.g., as "description" or "title"), so the algorithm simply looks for the presence of the target string, accounting for the fact that it may span multiple lines.

In this framing, document metadata detection is a token classification problem. A model or algorithm seeks to classify each token of the page's text as a part number, drawing number, revision, description, or none of those. If multiple tokens are classified as part number or drawing number, the token with the highest confidence is selected as the detected value. If multiple tokens are labeled as revisions, the tokens are alphanumerically sorted and maximum revision is selected as the detection. This handles a common case where a table may describe multiple sequentially identified revisions (e.g., 1, 2, 3, or A, B, C). If multiple tokens are classified as description, those tokens are concatenated in the order they occur in the document text. This handles the common case of long and multi-line descriptions.

Detections are used in the Paperless Parts application to assist with setting up parts for quotation. The user interface has form fields for the part metadata, and values are suggested for these fields when detected in an associated drawing. Accurately populating this metadata is valuable for multiple reasons: it creates clarity in quotations, it makes information about past jobs easy to find and use, and it enables integration with other software tools via Application Programming Interface (API).

The performance of metadata detection is measured by precision, P, and recall, R. Precision is the percentage of detections that are correct given the algorithm has a detection. Recall is the percentage of documents from which a target was detected, given that the document contains the target entity. Compared to the traditional framing of a detection problem, precision is inversely related to probability of false alarm and recall is related to probability of detection. Detection algorithms generally have a tunable threshold or other parameter that can be varied to trade off precision and recall. Formally, precision and recall are defined as:

$$P = p_t/(p_t + p_f) \tag{1}$$

$$R = p_t/(p_t + n_f) \tag{2}$$

where p_t is the count of true positives, p_f is the count of false positives, and n_f is the count of false negatives.

Paperless Parts has learned that users are highly sensitive to incorrect detections and internally requires a detection technique to achieve 75% or better precision on a benchmark set of drawings in order for that technique to be used in the application to assist users. Where applicable, solutions are tuned to maximize recall while maintaining very high precision.

All experiments and measurements in this paper are based on a dataset of roughly 100,000 engineering drawings. A holdout verification dataset of 291 drawings are used for performance evaluation.

B. Prior Solution

TABLE I: Prior Solution Performance

Part Number	Precision Recall	90.2% 63.2%
Drawing Number	Precision Recall	86.8% 79.0%
Revision	Precision Recall	90.6% 76.6%
Description	Precision Recall	62.1% 69.5%
Description (Allowing Partials)	Precision Recall	64.8% 70.3%

Paperless Parts has previously developed and deployed a metadata detection algorithm. The solution embodies significant domain knowledge. It uses multiple ML classifiers but does not use transformers or LLMs. The algorithm extracts engineered features from tokens, and the classifiers are then used to predict the probability of the tokens belonging to the various target classes. An inference algorithm then applies additional domain knowledge to determine the best detections and filter out detections with inadequate confidence. This solution required significant engineering effort, but achieved extremely low-cost inference and acceptable detection performance. Table I details the performance of the currently deployed solution

as measured on the holdout dataset. When partial matches are allowed, a detected description is considered a true positive if it is a subset of the true description. Paperless Parts is not currently using this solution for description extraction because precision is below the required 75%.

C. Proposed Approach

The proposed solution seeks to use a multi-modal LLM instead of a complex algorithm for document metadata detection. This trades engineering effort for training and inference cost. In other words, much less domain knowledge needs to be embedded into the algorithm as code, at the expense of longer training times and higher inference resource requirements.

LayoutLMv3 Base (133 million parameters) is selected as the pretrained model [7]. The training and testing dataset is formed using the drawing and label data described previously. Each point represents a page of an engineering drawing, and consists of the page image rendered at 300 dpi, the list of words (i.e., tokens) on the page, the bounding box of each word, and the true class (i.e., label) of the word. Data points are then tokenized with the LayoutLM tokenizer. For words split into multiple tokens, the corresponding bounding boxes and labels are duplicated and aligned. LayoutLM v3 has a 512 token limit, while engineering drawings may in general be longer. For this training exercise, the dataset consists only of documents with 512 tokens or fewer per page. To support larger documents, techniques could be applied to split pages into separate data points or to truncate documents. The LayoutLM Base model is then fine tuned for a token classification task, using a learning rate of 2×10^{-5} and three epochs. The training dataset size is varied as an independent variable.

The set of classes are labeled "PN" (part number), "DN" (drawing number), "Rev" (revision), "Desc" (description), and "O" (for any other token "outside" the target classes). When used for inference, the token classification model calculates the probability $P(c_i = C_j \mid x_i)$, where x_i is the i-th token, c_i denotes the class of x_i, and $C = [\mathrm{O}, \mathrm{PN}, \mathrm{DN}, \mathrm{Rev}, \mathrm{Desc}]$ represents the set of possible classes. In a straightforward implementation, each token would be classified according to its most likely class, which in testing resulted in higher precision at the expense of recall. Instead, a threshold θ is applied to the target classes such that the token is classified as the most likely non-"O" class provided its probability exceeds the threshold, as shown here:

$$c_i = \begin{cases} C_{\arg\max_j P(c_i = C_j \mid x_i)} & \text{if } \max_{j>0} P(c_i = C_j \mid x_i) \leq \theta \\ C_{\arg\max_{j>0} P(c_i = C_j \mid x_i)} & \text{if } \max_{j>0} P(c_i = C_j \mid x_i) > \theta \end{cases} \tag{3}$$

Performance can then be calculated for several values of θ, which can be selected to tune the model for the desired trade-off of precision and recall.

D. Results

Training is performed in SageMaker on an ml.g4dn.4xlarge compute instance, which has 16 CPU cores, 64GB of memory, and one 16GB NVIDIA T4 GPU. The HuggingFace PyTorch implementation of LayoutLMv3 is used [16]. Models are trained for a variety of dataset sizes. For each model, 90% of the dataset is used for training and 10% is used for validation between epochs. The holdout dataset (described earlier) is used to compute final performance metrics. Fig. 2 shows the precision and recall for for each target entity as a function of training dataset size for a fixed threshold, θ. Fig. 3 shows performance as a function of threshold, θ, for a fixed dataset size of 60,000 pages. Table II shows the training time required for the dataset sizes.

TABLE II: Training Time by Dataset Size

Dataset Size (1000s)	5	10	20	40	60
Training Time (hours)	7.2	13.3	27.6	53.2	66.2

The performance of the LLM-based models are approaching but do not match the current-state performance for part number, drawing number, and revision. For description, the recall when allowing partial matches now exceeds the minimum requirement of 75%, which was not achieved by the prior solution. This model is not being used in the application, but there are several paths forward for improving performance. There are larger and newer multi-modal LLMs available, and training can be performed with a larger dataset. Some models (such as [5]) support unsupervised pre-training using masked language modeling; this technique may enable better performance on the downstream token classification task. In the future, larger GPUs can be used to support larger base models, but at present, the selection of GPUs available in AWS GovCloud is limited.

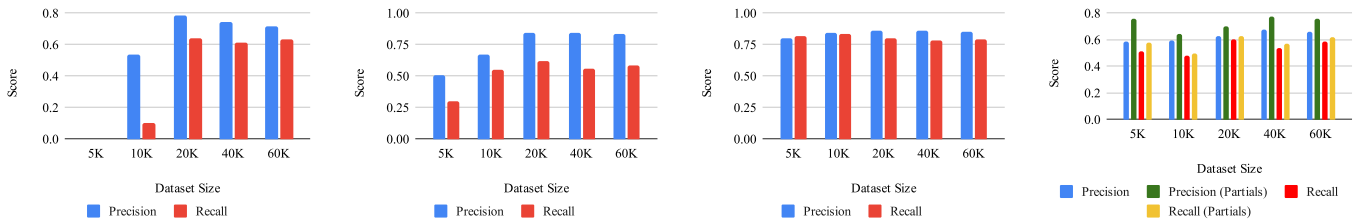
IV. Use Case 2: Email Message Entity Recognition

A. Problem Statement

Most quotes start as request-for-quotations (RFQs) sent to a manufacturer via email. RFQ emails contain natural language bodies (and sometimes include conversation threads) which may describe the line items to be quoted in an unstructured or semistructured format, such as sentences, bulleted lists, and tables. They may also convey technical data as file attachments or via file sharing links. Parsing the highly varied content and plethora of files into well-defined line items with part numbers, requested quantities, and associated files is tedious and time-consuming, even for an experienced estimator. This makes quote setup one of the most common sources of quoting delay, for organizations where delayed responses to RFQs result in lost opportunities.

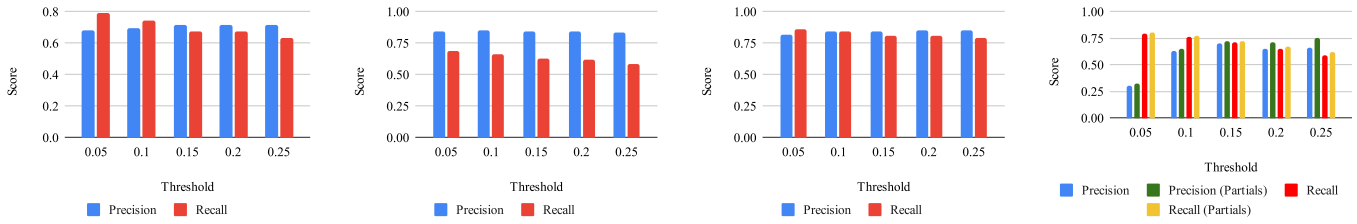
This use case focuses on identifying quote items from email bodies. Part numbers must be extracted, while quantities are extracted if they are present and can be linked to a part number. Fig. 4 shows a typical examples of an RFQ email.

Paperless Parts offers a spreadsheet-like user interface for creating items on a quote. If quote items can be reliably extracted from email messages, then the extractions can be used to assist the user by pre-populating the interface, which can save the user considerable time. Part numbers have no



(a) Part Number (b) Drawing Number (c) Revision (d) Description

Fig. 2: Precision and recall are measured as a percentage versus training dataset size, for a fixed detection threshold of 0.25. Performance increases with training dataset size, with diminishing returns beyond 20,000 pages.



(a) Part Number (b) Drawing Number (c) Revision (d) Description

Fig. 3: Precision and recall (shown for a fixed dataset size of 60,000 pages) can be traded off based on application requirements by varying detection threshold.

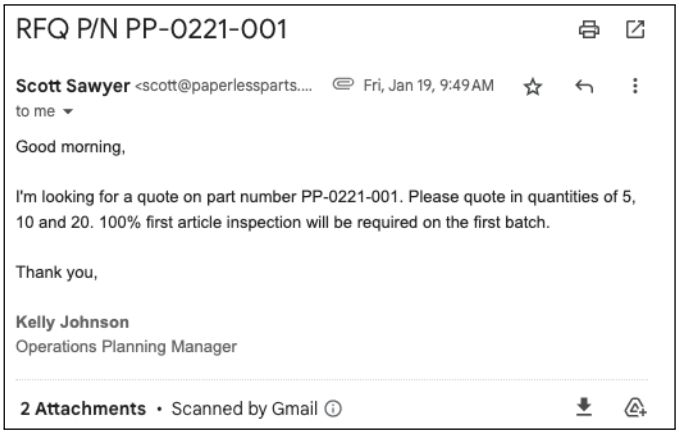


Fig. 4: RFQ email messages may contain part numbers and quantities in unstructured natural language. In other cases, the messages may be semi-structured, where tables with arbitrary format and column headers list part numbers, quantities, as well as other quote item metadata.

consistent format across designers and industries. Therefore, LLMs may be a good fit for this use case, particularly when the message contains completely unstructured text. State-of-the-art hyperscale LLMs are likely very effective at this task; however, they are not suitable for use with CUI and cannot be tested with real data. Similar to the first use case, users have an expectation of accurate suggestions. If the solution involves a generative model, precautions must be taken to protect against incorrect answers and hallucination, and the approach must preclude data leakage between users. Due to business and scalability requirements, the solution is further constrained to run on a single 16GB GPU (ml.g4dn.xlarge instance) or to run inference on CPU using an instance of roughly equivalent cost (e.g., ml.t5.4xlarge). Furthermore, to support the existing user experience, inference must complete in less than 10 seconds per email message.

B. *Proposed Approach*

In the proposed solution, RFQ emails are serialized to text, including the body of the message, subject line, and filenames of attachments. The email is further pre-processed to remove irrelevant text, tokenized, and it is truncated to the maximum supported 2048 tokens. An auto-prompting algorithm queries the LLM to identify requested part numbers, and iteratively queries for requested quantities associated with those part numbers. The generated output is cleaned and filtered to ensure all identified quote items meet basic criteria and that extracted strings exist in the original message.

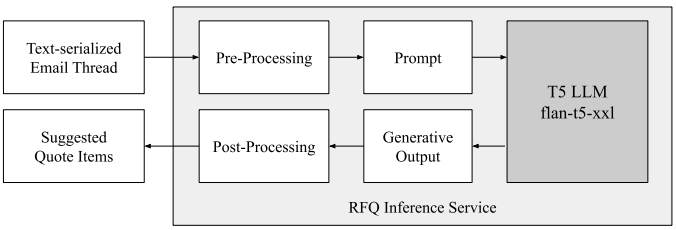


Fig. 5: As shown in this block diagram, RFQ email messages are processed by a LLM-powered service.

TABLE III: LLM Extraction Performance

Model Name	Instance Type	Parameters	Quantization	Precision	Recall	Avg. Seconds per Email
flan-t5-xxl	ml.t5.4xlarge	11B	32-bit	31.95%	45.56%	43.05
flan-t5-xxl	**ml.g4dn.xlarge**	**11B**	**4-bit**	**21.00%**	**37.28%**	**4.25**
flan-t5-xl	ml.m5.2xlarge	3B	32-bit	88.24%	17.75%	16.30
flan-t5-xl	ml.g4dn.xlarge	3B	8-bit	85.71%	17.75%	4.78

This architecture is privacy preserving by design. T5 is not fine-tuned based on the proprietary dataset. Furthermore, the post-processing of the LLM's generated output prevents hallucination. Early experiments showed that T5 performs better for question answering with shorter, simpler questions.

When the RFQ message contains a table, this table structure can be used to improve extraction quality, especially when the message lists a large number of quote items. The pre-processing step detects tables and applies heuristics to determine if the table is likely describing quote items. If so, the LLM is prompted to identify which columns correspond to metadata columns (part number, quantities, revision, description). The post processor then uses HTML parsing to extract the items from those tables using the LLM- generated column mapping.

C. Results

Similar to the first use case, recall and precision are important metrics for evaluating performance of this extraction solution. The extraction performance is measured by examining the combinations of part numbers and quantities. For instance, if a quote item includes part number 123 with requested quantities of 1, 5, and 10, this is represented as three distinct tuples: (123, 1), (123, 5), and (123, 10). To obtain ground truth data, a sample of 100 real emails is manually labeled to identify the actual tuples present. The model's output is then compared against this truth data. The model will be measured prior to the full post-processing step to better isolate the performance of the LLM. Post processing will then remove most false positives by filtering out part numbers that did not occur in the original message. Therefore, the objective is to maximize recall while meeting the constraints placed on runtime and EC2 instance size.

The solution was implemented with multiple models for comparison. Models are constrained to fit on reasonably sized EC2 instances as determined by business and technical requirements. This limits the model to the ml.g4dn.xlarge instance for GPU inference (which has a 16GB GPU) and the ml.t5.4xlarge for CPU inference. Performance and inference time are then compared for different sized models and different precisions. Parameter quantization is used to reduce the memory footprint of models when using GPU inference. Table III shows the results of four variations of models. A GPU solution is required to meet latency requirements, while the larger parameter count at lower precision outperforms the alternative.

V. Conclusions

In the first use case of this paper, an AI solution is compared to a more traditional ML solution for the problem of extracting metadata from engineering drawings. The results highlight the unique strengths and challenges of each approach, particularly in manufacturing use cases. ML relies heavily on data science and extensive engineering efforts to develop domain-specific solutions tailored to limited datasets. These solutions are often highly efficient during inference, providing robust performance for compatible problems. In contrast, the potential of AI is to leverage vast datasets to automate the creation of these solutions, reducing the need for data science. However, AI's data-hungry nature can be a significant drawback, especially when domain-specific algorithms can deliver comparable or superior performance with smaller, proprietary datasets. In manufacturing contexts, where data may be scarce or highly specialized, traditional ML approaches may offer effective and efficient solutions.

In the context of regulated data like CUI, it is important to recognize the divide between state-of-the-art, hyperscale, commercial AI models and those models available in secure clouds compliant with regulatory regimes like FedRAMP, such as AWS GovCloud and Microsoft Azure GCC. The Foundation and OpenAI models available in those two clouds, respectively, significantly lag the latest models deployed in commercial offerings.

Additionally, using generative AI models has risks related to privacy, interpretability, and provenance of responses. Given the security requirements around CUI, it would not be viable to train a generative model on CUI and then allow that model to generate unconstrained output to users without authorization to access the full training data set. This paper presents two solutions involving LLMs that mitigate these risks. In one case, the LLM is fine-tuned with sensitive data; in the second, the LLM is completely open-source. In both, the LLMs power a detection algorithm. By their nature, these detection solutions are constrained to return subsets of the input document, preventing data leakage and preserving privacy. This approach ensures that the benefits of AI can be harnessed while maintaining data security and cybersecurity compliance, benefiting the DIB as parts and technical data continue to get more complex and as CMMC rolls out across the industry.

Acknowledgments

The author thanks Paperless Parts colleagues Hank Portney and Luke Duros for the reviewing the methodology and report and Joseph St. Pierre for his dedicated work to design and deploy a SageMaker environment within the FedRAMP boundary. We are intentional. We are persistent. We are a team.

REFERENCES

[1] L. Uchitelle, "The U.S. Still Leans on the Military-Industrial Complex," *The New York Times*, September 2017, accessed: 2024-07-12. [Online]. Available: https://www.nytimes.com/2017/09/22/business/economy/military-industrial-complex.html

[2] R. Ross, M. McEvilley, J. C. Oren, R. Graubart, D. Bodeau, and R. McQuaid, "Protecting controlled unclassified information in nonfederal systems and organizations," NIST, Tech. Rep. NIST SP 800-171, 2016, accessed: 2024-07-12. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-171r2

[3] Defense Federal Acquisition Regulation Supplement, "DFARS 252.204-7012 (b)(2)(ii)(D)," 2016, accessed: 2024-07-12. [Online]. Available: https://www.acq.osd.mil/dpap/dars/dfars/html/current/252204.htm#252.204-7012

[4] NIST, "Security and privacy controls for information systems and organizations," NIST, Tech. Rep. NIST SP 800-53, 2020, accessed: 2024-07-12. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-53r5

[5] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "Layoutlm: Pre-training of text and layout for document image understanding," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, Aug. 2020. [Online]. Available: http://dx.doi.org/10.1145/3394486.3403172

[6] Y. Xu, Y. Xu, T. Lv, L. Cui, F. Wei, G. Wang, Y. Lu, D. Florencio, C. Zhang, W. Che, M. Zhang, and L. Zhou, "LayoutLMv2: Multi-modal pre-training for visually-rich document understanding," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, C. Zong, F. Xia, W. Li, and R. Navigli, Eds. Online: Association for Computational Linguistics, Aug. 2021, pp. 2579–2591. [Online]. Available: https://aclanthology.org/2021.acl-long.201

[7] Y. Huang, T. Lv, L. Cui, Y. Lu, and F. Wei, "Layoutlmv3: Pre-training for document ai with unified text and image masking," in *Proceedings of the 30th ACM International Conference on Multimedia*, ser. MM '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 4083–4091. [Online]. Available: https://doi.org/10.1145/3503161.3548112

[8] G. Kim, T. Hong, M. Yim, J. Nam, J. Park, J. Yim, W. Hwang, S. Yun, D. Han, and S. Park, "Ocr-free document understanding transformer," 2022. [Online]. Available: https://arxiv.org/abs/2111.15664

[9] J. Villena Toro, A. Wiberg, and M. Tarkian, "Optical character recognition on engineering drawings to achieve automation in production quality control," *Frontiers in Manufacturing Technology*, vol. 3, 2023. [Online]. Available: https://www.frontiersin.org/journals/manufacturing-technology/articles/10.3389/fmtec.2023.1154132

[10] "HighQA," https://www.highqa.com, 2024, accessed: 2024-07-12.

[11] "1factory," https://www.1factory.com, 2024, accessed: 2024-07-12.

[12] "Werk24," https://www.werk24.io, 2024, accessed: 2024-07-12.

[13] AWS, "Amazon sagemaker," https://aws.amazon.com/sagemaker/, 2024, accessed: 2024-07-12.

[14] ——, "Aws services in scope by compliance program," https://aws.amazon.com/compliance/services-in-scope/FedRAMP/, 2024, accessed: 2024-07-12.

[15] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive nlp tasks," 2021. [Online]. Available: https://arxiv.org/abs/2005.11401

[16] HuggingFace, "Layoutlmv3," https://huggingface.co/docs/transformers/en/model_doc/layoutlmv3, 2024, accessed: 2024-07-12.