Using LLM-based Prompt Chaining Workflows to Extract Energy Program Data from Unstructured Reports

Aaron Brown, Jacob Moul, Xinyi Gu, DNV and Dana Nilsson, NYSERDA[1]

ABSTRACT

The availability of complete, structured, high-quality data is critical for both program implementers and evaluators to quantify the full value of energy programs. However, program data is often provided to program administrators in unstructured report formats like text, tables, and figures with varying format, content, and detail. This paper describes an approach that utilizes the expanding capability of large language models (LLMs) to enhance program data capture and help demonstrate program energy impacts. This LLM solution was used to extract data from unstructured text to estimate the energy savings of a Real Time Energy Management program. This program provides cost-sharing incentives to support the installation of energy management systems and recommends additional energy savings measures. To receive incentives, each vendor must report the project's energy savings and the implementation status of recommended measures. The LLM solution provides a cost-effective way to understand the program's direct and indirect impact achievements for commercial, small to medium business, and industrial customers.

Results from this study show that LLM-powered prompt chaining workflows can enhance program implementation and evaluation studies by effectively extracting data from unstructured documents. We were able to analyze over 1,000 reports with <5% error and omission rate. With this workflow we extracted a wide range of data and information while maintaining strict data security through use of a secure on-premise (non-cloud) solution. Combining this tool with more typical data management practices can enable implementers and evaluators to cost-effectively leverage all available data, and this paper will discuss the future opportunities unlocked by this new way of working.

Introduction

Energy management programs require participants to submit documentation to verify project activities and outcomes, while also aiming to minimize reporting burden. As a result, programs often receive information in a variety of formats, such as project reports, invoices, plans, forms, and other documentation, submitted as PDFs, Word documents, or scanned files. These files often contain critical information – measured energy savings, installed equipment specifications, implementation status of recommended measures, and site-level savings. However, that information is embedded in narrative text, tables, or figures rather than captured in structured, queryable fields. From a program management perspective, this means the data exists, but it is not accessible.

The practical consequence is that program administrators and evaluators can typically only work with the subset of data that can be manually reviewed and keyed into a database or tracking system. Manual extraction is resource-intensive and therefore limited in scale: it is feasible for a small sample of reports or for a narrow set of high-priority fields, but it cannot realistically cover the full range of information contained across hundreds or thousands of

[1] *The views expressed in this paper are those of the authors and do not necessarily reflect the views of the New York State Energy Research and Development Authority (NYSERDA).*

submitted documents. Data that is never extracted remains largely invisible – it cannot be aggregated, analyzed, or used to demonstrate program impact. Many good alternatives exist to manage/sample the population to evaluate the impact of a program but we might imagine what value we left out by not having the ability to cast a wider net on information

This gap between data collected and data used represents both a measurement challenge and a missed opportunity. For program evaluators, data limits may reduce the ability to quantify savings, verify measure implementation, and produce defensible impact estimates. For program administrators, it constrains the ability to monitor performance, identify underperforming vendors, and communicate program value to funders and regulators. As programs are asked to demonstrate increasingly rigorous accountability for claimed savings, the limitations of manual data management become a binding constraint.

Recent advances in large language models (LLMs) offer a potential path forward. LLMs are capable of reading and interpreting unstructured text – including the kind of varied, inconsistently formatted prose found in vendor reports – and extracting specific data elements at scale. When deployed as part of an AI system, where the LLM is directed by a defined workflow to process documents, extract targeted information, and return structured outputs, LLMs can perform extraction tasks that would otherwise require significant manual labor. This capability is now sufficiently mature and cost-effective to be considered a practical tool for program implementation and evaluation, not merely a research concept. In expanding the number of documents that can be analyzed along with increase total attributes, the richness of the existing and future analyses may compound in interesting ways that support other RTEM and non-RTEM programs as well. In general, these capabilities may expand the scope of verification work, improve stratification for more efficient sampling, improve program transparency and improve reproducibility of analysis and program outcomes.

This paper describes the application of this approach to a Real Time Energy Management (RTEM) program evaluation. RTEM provides cost-sharing incentives to support the deployment of software and services that leverage energy management system (EMS) data to monitor building performance in real time. These systems analyze data to identify underperforming equipment, detect inefficiencies, and recommend operational and maintenance actions that can reduce energy use and improve system performance. To receive incentives, vendors must submit project reports documenting energy savings and the implementation status of recommended measures. These reports vary considerably in format, length, and detail, and historically, the information they contain has been only partially captured for program tracking. We developed and deployed an LLM-based prompt chaining workflow to extract structured data from this corpus of unstructured reports. This paper presents the system design, extraction methodology, and results, and discusses the broader implications for program implementation and evaluation practice.

The design of this system was shaped by several practical constraints common to program evaluation contexts. These included strict data security requirements that limited deployment to an on-premise environment, highly variable and non-standardized input documents, and the need to process a large volume of reports within typical evaluation timelines. These constraints influenced both the selection of model architecture and the development of a structured, prompt-driven workflow. Approaches requiring extensive retraining, external data transfer, or highly standardized inputs were not feasible. In addition, system design and deployment were aligned with applicable New York State and NYSERDA guidance related to artificial intelligence governance, including requirements for data security, controlled

environments, and human oversight (New York State ITS 2024; NYSERDA n.d.). Framing the approach within these constraints helps clarify where LLM-based extraction methods can provide practical value relative to traditional alternatives.

The remainder of this paper is organized as follows. The Background section describes the RTEM program in greater detail, characterizes the data challenge posed by unstructured vendor reports, and reviews prior work on automated data extraction vs the architecture of the LLM-based prompt chaining workflow, the on-premise deployment approach, and the data extraction and validation pipeline. The Results section reports on the volume of data extracted, site-level savings accuracy and limitations. The Discussion section interprets these findings and addresses scalability, limitations, and future opportunities. The paper concludes with practical recommendations for program administrators and evaluators considering similar approaches.

Background

Real Time Energy Management Program

RTEM programs are administered by utilities or state energy agencies to accelerate the deployment of advanced energy management systems across numerous sectors they serve. These programs recognize that sophisticated monitoring and analytics capabilities – though increasingly cost-effective – still represent a meaningful upfront investment for many building owners, and they use cost-sharing incentives to close the financial gap. By subsidizing hardware, software, and installation costs, RTEM programs aim to lower the adoption barrier and expand the installed base of actively monitored facilities.

EMS deployment under an RTEM program may include sensors and sub-meters for key energy end uses, a data acquisition layer that transmits readings to an analytics platform, and dashboards and/or alerts (which in turn may be informed by Machine Learning or AI-drive algorithms) that make energy data actionable for facility managers. Beyond the monitoring infrastructure itself, a defining feature of RTEM programs is the measure recommendation component: participating vendors are expected not only to install the EMS but to actively analyze the data it produces and recommend energy conservation measures (ECMs) based on observed operational patterns. These recommendations may address controls sequences, scheduling, setpoint optimization, equipment maintenance, or capital upgrades, and they represent a significant portion of the program's anticipated energy impact.

In this program, vendors must document both the RTEM system installation and their measure recommendations in project reports submitted to the program administrator. These reports constitute the primary evidence record for program evaluation. For evaluation purposes, the reports must establish that the EMS was properly installed, that the recommended measures are appropriate and technically sound, and that the claimed energy savings are credible and supported by measured or calculated data. The completeness and quality of these reports therefore directly determines the quality of the program record and the confidence with which evaluated savings can be reported.

The Data Challenge

Converting Documents to Plain Text. Vendor reports submitted to RTEM program administrators can arrive in a variety of formats. The main formats analyzed in this study were PDFs and word-processed documents, but reports may also include scanned images, embedded

spreadsheets, and proprietary engineering file formats. Converting these documents to plain text – the input format required by most text processing systems, including LLMs –is a technically non-trivial step that introduces error even before any data extraction is attempted.

For digitally-native PDFs, text extraction tools can recover character sequences, but the spatial layout of the original document is not preserved in a simple text stream. Multi-column layouts collapse into a single sequence that may interleave content from separate columns; tables are rendered as flat strings of values stripped of their row-column relationships; and figure captions, headers, and footers may appear inline with body text. Ligature characters, non-standard fonts, and embedded encoding artifacts further degrade the quality of extracted text. Though we didn't address in this study, scanned documents require an additional optical character recognition (OCR) step, which introduces character-level errors at a rate that depends on scan quality, font, and page condition. Even at low raw error rates, these mistakes compound across multipage technical reports and can corrupt numerical values, unit labels, and critical field identifiers. By the time you are reading this, many new capabilities, including the ability to process images and PDFs have become common components of multimodal models (Poznanski et al. 2025).

Extracting Structured Data from Plain Text. Once documents are converted to plain text, the extraction task itself poses a second layer of challenges. Vendor reports are written to communicate findings to a human reader, not to populate a database. The same data element – for example, the annual energy savings claimed for a specific measure – may appear in a formatted summary table in one report and in a sentence such as "implementation of the recommended cooling tower optimization is expected to yield approximately 47,000 kWh annually" in another. Synonymous terminology, varied units, and inconsistent levels of precision make rule-based pattern matching unreliable across a heterogeneous corpus (Jurafsky and Martin 2023).

Extraction is further complicated by the narrative structure of technical reports. Context matters: a savings figure cited in a discussion of projected future measures carries a different meaning than the same figure cited as a verified baseline-period result. A keyword-and-value extractor cannot distinguish these cases; the system must read and interpret the surrounding language to assign meaning correctly. Reports may also contain multiple savings figures corresponding to different scenarios, time periods, or measure categories, requiring the extraction system to select the appropriate value rather than return the first numerical match. Critical for this analysis was to capture as much of the data over time as well. Reports for the same customer can span several years.

The cumulative effect of these challenges explains why manual extraction has remained the default approach in program administration despite its cost. Human analysts can navigate ambiguous formatting, interpret contextual meaning, and exercise judgment when information is incomplete or contradictory. Replicating this capability in a scalable, automated system requires a technology that can read and reason about natural language – a capability that has only recently become feasible with the maturation of modern LLMs.

Prior Work

Information extraction from unstructured text is a long-standing research area in natural language processing (NLP), encompassing tasks such as named entity recognition, relation extraction, and structured slot-filling (Jurafsky and Martin 2023). Early approaches relied on hand-crafted rules and finite-state automata, which were accurate within narrow domains but

required significant engineering effort to adapt to new document types. Statistical and machine learning methods improved generalization but still required labeled training examples from the target domain – a constraint that limits their applicability to specialized corpora such as energy program reports, where annotated datasets are scarce.

The introduction of transfer learning for NLP fundamentally changed this landscape. Early work demonstrated that a language model pretrained on a large general corpus could be fine-tuned for downstream text classification tasks with minimal labeled data – a paradigm established by ULMFiT (Howard and Ruder 2018). BERT (Devlin et al. 2019) extended this approach using transformer architectures, showing that large pretrained models could be fine-tuned for a wide range of downstream NLP tasks with small amounts of task-specific labeled data. Subsequent work on generative models – including the GPT family (Brown et al. 2020) and their successors – showed that sufficiently large models could perform information extraction through in-context prompting alone, without any task-specific fine-tuning. This capability, known as few-shot or zero-shot learning, dramatically lowered the barrier to deploying extraction systems on specialized document types, because it eliminated the requirement for a labeled training corpus. More recent work on chain-of-thought prompting demonstrated that guiding a model through intermediate reasoning steps further improves accuracy on complex extraction and classification tasks (Wei et al. 2022).

The concept of structuring LLM interactions within a defined workflow – an "agentic" architecture - emerged as a practical pattern for handling multi-step document processing tasks (Wang et al. 2024). Rather than relying on a single model call to process an entire document, agentic systems decompose the task into stages: document ingestion, text segmentation, field-specific prompting, output parsing and validation, and exception handling. This architecture improves reliability on long or complex documents and enables human review to be concentrated at validation checkpoints rather than applied document-by-document.

Applications of these methods within the building energy domain remain relatively nascent. Prior work has explored NLP for mining energy-related information from online sources, classifying ECMs from text descriptions in work order records, and extracting equipment specifications from product datasheets (Khanuja and Webb 2024). Incomplete and inconsistent reporting puts a constraint on evaluation rigor, with remedies including prescriptive reporting templates and database-driven data entry. These approaches address the problem prospectively – by requiring future reports to conform to a standard – but cannot recover data from historical reports submitted before standardization, nor can they accommodate the inherent variability that persists even when templates are provided.

The specific challenge of extracting program performance data from vendor-submitted reports across an entire program portfolio – where document volume, format diversity, and data security requirements all constrain the solution space – has not been addressed in the published literature. This paper describes a system designed to meet those constraints and presents results from its deployment on a live RTEM program portfolio.

Methodology

To address the challenge of extracting consistent data from diverse vendor reports, we developed a LLM-based prompt chaining tool. Given the non-deterministic nature of LLM outputs, the system was designed with a structured validation process that incorporates qualified human supervision at multiple stages to confirm accuracy and completeness. This approach utilizes the capabilities of Large Language Models (LLMs) to follow the prompt chaining

workflow, read, and interpret unstructured documents. The system extracts plain text and tables from PDF and word processing documents. The system was designed to extract a wide range of data and information-from specific energy savings figures to qualitative descriptions of measure implementation status-while adhering to strict data governance protocols.

Accounting for specific project needs, a critical requirement for this study was data security. Program documentation often contains sensitive customer information, which precluded the use of public cloud-based LLM APIs. As a result, our solution was implemented as a secure on-premise (non-cloud) application. This local deployment ensures that no data ever leaves the secure evaluation environment. The use of local LLM outputs was further combined with traditional data management practices and manual quality assurance workflows to enable evaluators to cost-effectively leverage all available data. This facilitates a more comprehensive data collection process for analysis of the entire program population.

System Architecture

The system is architected as a LLM-based prompt chaining workflow rather than a simple text-processing script. In this context, an LLM is invoked with the capability of observing its environment (the document text), following prompt instructions to elicit information from the document (identifying relevant sections, tables, and specific attributes), and following prompts to extract specific information as structured output to complete the workflow.

The architecture consists of three primary layers:

- **Document Processing Layer**: Handles the ingestion of PDF and Word documents, converting them into, machine-readable text while attempting to preserve structural cues like headings and table layouts.
- **Reasoning & Extraction Layer**: The data extraction workflow was powered by a locally hosted LLM. This layer receives the processed text and a schema of target data fields. It uses a combination of approaches including "chain-of-thought" and instruction-following prompt engineering strategies to locate relevant information, resolve ambiguities (e.g., distinguishing between proposed and installed savings), and format the output.
- **Data Management Layer**: After a series of postprocessing steps, the LLM structured output was transformed and organized into spreadsheets that were used for (1) quality assurance activity, (2) enforcing data types and (3) enabling downstream analysis.

On-Premise Deployment

Deployment mechanisms for LLMs are limited because they use specialized hardware (GPUs). There are cloud hosted commercial services and self-hosted, on-premise implementations using open-weight LLMs (e.g., Llama 3). Identification and selection of a cloud hosted LLM would require vendor qualification and potentially extend schedules as a result. For this RTEM program evaluation, utilizing Llama 3.3 with the on-premise approach was necessary to meet strict data security requirements and project schedules.

We deployed the system in a self-hosted datacenter behind corporate firewalls to ensure the data could be protected. The hardware infrastructure utilized enterprise datacenter-grade GPUs sufficient to run quantized versions of high-performance open-weight LLMs. This setup provided two key advantages:

- **Data Sovereignty**: Complete assurance that sensitive vendor and customer data remained within the organization's control boundary.
- **Cost Predictability**: A fixed hardware investment rather than variable per-token API costs, which is particularly important when processing large volumes of verbose text.

Data Extraction Pipeline

Data Inputs. The vendor reports processed by this system reflect the full range of variability characteristic of practitioner-authored technical documents. Reports ranged in length from a dozen pages to over fifty pages and were transmitted as PDFs and word processing documents. Within this corpus, no two vendors used the same report template. Some vendors produced highly structured documents with consistent section headings, numbered and well formatted tables, and standardized summary pages; others submitted narrative-style reports in which savings figures, measure descriptions, and implementation status were interspersed throughout the body text without clear delineation.

Table formats posed a particular challenge and opportunity because the structure, conciseness and consistency makes it easier to apply our LLM data extraction techniques. Still, savings summary tables varied in structure: some organized measures by row with columns for energy type, baseline usage, and savings; others grouped measures in various ways. Column headers were inconsistent – the same data element might be labeled "Annual kWh Savings," "Estimated Savings (kWh/yr)," or simply "kWh" across different vendors. Some reports presented savings in combined electric-and-gas tables, while others separated fuels into distinct sections of the document. In several cases, the same report contained multiple savings tables corresponding to different reporting periods, requiring the extraction system to distinguish between them and associate each with the correct time frame. The system also needed to handle reports spanning multiple years for the same customer, where cumulative and incremental savings figures coexisted within a single document or across a series of documents. Despite the challenges, these are issues that can be effectively managed with a well-designed LLM-based prompt chaining workflow, as described below.

Extraction Pipeline. The extraction pipeline transforms these raw files into structured records through a multi-step process. Each step manages the document variability described above to produce outputs suitable for both automated downstream processing and human review.

The prompt chaining workflow allowed the system to be built with flexibility in mind so that it could be iteratively improved and adapted to the specific characteristics of the documents being processed. Each step in the workflow would (1) invoke an LLM or (2) programmatically process the LLM output to prepare for the next step or save output from the extraction workflow. For example, a summary table of measure level savings consists of (1) a step to identify the presence of a table, (2) a step to extract each measure level savings name/ID and (3) a varying number of steps to extract the associated savings.

Following a common pattern, each step in the workflow could be customized to the task. Each workflow step applied one or more prompt engineering techniques to improve the accuracy of the extraction. Prompt engineering strategies included:

- Chain-of-thought prompting[2]
- Instruction-following prompting[3]
- Few-shot prompting with examples from the document corpus[4]
- Iterative prompting with feedback loops to refine outputs[5]
- Post-processing steps to enforce data types and resolve ambiguities
- Structured output formatting to facilitate downstream parsing[6]
- Exception handling to manage edge cases and errors[7]

While every use of an LLM has some susceptibility to inaccuracy, the above techniques in combination were used to constrain and manage the workflow to minimize this (note that combination of these techniques limits the manual review process, but do not eliminate it). With each application of these techniques, we can customize and minimize the level of ambiguity in the resulting information that is extracted.

Output Format. The final step of the pipeline produces structured outputs. Internally, the system organizes extracted data into machine-readable key-value pairs (JSON) and exports into tabular (CSV) and spreadsheet formats for analysis.

Output structuring also enforces consistency across the dataset and dictates the data types for each extracted attribute (e.g., numeric, categorical, free text). Enforcement at this stage is critical for being able to ensure downstream analysis can be performed without manual manipulation of the data. This structuring also includes units of measure if relevant.

Validation

This project underwent extensive accuracy checks and validation to ensure the reliability of the extracted data. Validation consisted of two primary phases:

- **Iterative validation during development**: During the development of the extraction pipeline, we applied an iterative validation approach by qualified human supervision from AI and subject matter staff. During development and after each major change to the system, we ran a batch of documents through the pipeline and manually reviewed the outputs for accuracy and completeness. This process allowed us to identify specific failure modes (e.g., misidentification of savings tables, incorrect parsing of numerical values) and refine our prompt engineering strategies and post-processing rules accordingly.

[2] Chain-of-thought prompting is a prompt engineering technique that instructs an LLM to work through intermediate reasoning steps before producing a final answer.
[3] Instruction-following prompting is a prompt engineering technique that gives an LLM explicit task directions, constraints, and output requirements.
[4] Few-shot prompting is a prompt engineering technique that provides example input and output to an LLM
[5] Iterative prompting breaks the task into primary and subordinate tasks so that the data extraction step invoked is narrow enough in scope to not dilute the LLMs capability to extract necessary data
[6] Structured output is a prompt engineering technique where the LLM response is in a specific format that can be verified easily
[7] Exception handling has its roots in programming and in this context intercepts malformed responses from the LLM and retries or logs and exits the process safely to continue further extraction

- **Final validation on a random sample**: In order to ensure the system's performance was robust across the full corpus of documents, a random sample of the processed reports (20% of the dataset) was selected for qualified human review. The checks within this sample were used to assess the accuracy and consistency of extracted data, show that LLM performance was consistently applied, and to highlight any remaining limitations or failure modes.

Results

Scale of Data Extraction

The LLM-enabled workflow processed over 1,000 vendor reports. Each report contained 30-60 distinct extracted attributes, including site-level energy savings (electric and gas), measure-level savings breakdowns, measure implementation status, equipment descriptions, and metadata. On average, the system processed approximately 30 reports per hour on the on-premise hardware described in the Methodology section, with processing time varying based on document length and number of measures.

Site-Level Savings Accuracy

The key metric we used to assess extraction accuracy was site-level savings, defined as the total reported energy savings per customer site. Because some reports do not describe measure-level savings, site-level figures capture a broader and more consistent set of the total reports and provide the most representative measure of system performance across the full corpus.

Sampling 20% of reports for analysis, the system achieved an error and omission rate of under 5%.

Under constraints of timeline and budget, this was a fair outcome for the number and variability of the documents analyzed.

Discussion

Unlocking Previously Inaccessible Data

In this study, the LLM-based prompt chaining workflow processed over 1,000 vendor reports and extracted 30–60 attributes per report with site-level error and omission rates under 5%. We recorded errors when the site-level savings were captured incorrectly, when site level savings were recorded but did not actually exist in the report and when site-level savings did exist but were not reported through LLM prompt chaining workflow. This produced a volume of structured data that would not have been feasible through manual review alone, given the labor hours required to read, interpret, and key in data from each report.

The practical consequence is a more complete program record. RTEM programs are expected to demonstrate that incentive expenditures produce measurable energy savings, and the strength of that demonstration depends in part on the completeness of the underlying data (SEE Action 2012). In many evaluation contexts, limited access to billing data and inherent

uncertainty in billing-based analysis can constrain the ability to fully characterize program impacts. Under typical manual workflows, only a fraction of submitted reports are fully reviewed - savings that are documented by vendors but never extracted into the tracking system remain unavailable for analysis. We estimated it would take at least 5 humans working 24-hours per day to match the same volume of data extracted. In this case, processing the full corpus allowed evaluators to construct a more complete picture of program impacts, including both direct savings from EMS installation and indirect savings from recommended measures, for a larger share of program participants than would otherwise have been reviewed.

Scalability and Generalizability

The architecture described in this paper – a LLM powered prompt chaining workflow with specialized steps, each applying targeted prompt engineering strategies – was designed for the RTEM use case, but its components are not inherently program-specific. Adapting the pipeline to a different program would require (1) defining a new schema of target extraction fields, (2) assembling a representative sample of documents for iterative development and validation, and (3) investing in the prompt engineering and testing cycle described in the Methodology section. The effort involved in that adaptation is non-trivial but substantially less than training a custom LLM, primarily because no labeled training corpus is needed.

At a throughput of approximately 30 reports per hour on enterprise-grade GPU hardware, a portfolio of several hundred reports – typical of many state-level programs – could be processed in a matter of days. This timeline is compatible with standard evaluation schedules. Larger portfolios would require additional hardware or parallelization, though the marginal cost of processing additional documents is low relative to the fixed cost of system development.

Limitations and Caveats

Several limitations should be noted. First, extraction accuracy is bounded by the quality of the input documents. Reports with heavily degraded scans, tables embedded as images, or highly idiosyncratic formatting remain challenging and account for the majority of observed errors. As noted in the Results section, multimodal LLMs that can process document images directly offer a promising path to mitigating these issues as they continue to mature (Poznanski et al. 2025).

Second, the system requires meaningful upfront investment in prompt engineering, pipeline development, and iterative validation before it can be deployed on a new document type. While this investment is substantially lower than training a custom LLM – because no labeled training corpus is required – it is not zero. The prompt chaining workflow used here helps manage this complexity by allowing individual step in the prompt chain to be refined independently, but the initial development cycle still requires domain expertise and careful testing.

Third, an on-premise deployment imposes constraints on LLM selection. The open-weight LLMs available for local hosting – while increasingly capable – may not match the performance of the largest commercial or open-weight LLMs on certain extraction tasks. For this study, Llama 3.3 provided sufficient accuracy for our target use case, but requires extensive use of many prompt engineering techniques and human effort therein to craft the corresponding prompts. LLMs at the frontier of performance can achieve better results with significantly less intervention through extensive prompt engineering. The trade-off between data security and

LLM capability is a practical consideration that each program must evaluate based on its own data governance requirements.

Future Opportunities

The approach described in this paper opens several avenues for future work. First, as multimodal models become more accessible for on-premise deployment, the document processing layer can be simplified or eliminated, allowing the system to work directly from source PDFs without an intermediate text conversion step. This would reduce the error introduced by document conversion and expand the range of report formats the system can handle.

Second, the extracted data can be used to support program functions beyond evaluation. Structured data on measure implementation status, for example, could feed program dashboards that allow administrators to monitor vendor performance, identify sites where recommended measures have not been implemented, and target follow-up interventions. Similarly, aggregated data on measure types and savings magnitudes could inform future program design by revealing which measure categories contribute the most to portfolio-level savings.

Third, the iterative development workflow itself can be improved. As the extracted dataset grows, it becomes possible to use validated outputs as training or evaluation data for refining prompts, fine-tuning models, or benchmarking alternative extraction approaches. This creates a virtuous cycle in which each round of extraction improves the tools available for the next.

Finally, if automated extraction becomes routine in program administration, it may have secondary effects on reporting quality. Vendors who know that submitted reports will be systematically analyzed – rather than selectively sampled – may be more attentive to completeness and consistency in their documentation while program administrators can provide more timely feedback about the content they have received. Whether this effect materializes in practice would need to be assessed over multiple program cycles.

Conclusions

This paper described a LLM-base prompt chaining system that extracts structured data from unstructured vendor reports submitted to a Real Time Energy Management program. The system processed over 1,000 reports with an error and omission rate under 5% for site-level savings, indicating that LLM-based extraction can serve as a useful complement to traditional data management practices in energy program evaluation. On-premise deployment using open-weight LLMs satisfied the data security requirements of this engagement while providing sufficient extraction accuracy for population-level analysis.

For program implementers and evaluators considering similar approaches, we offer the following practical recommendations:

- **Start with a well-defined extraction schema.** The most important design decision is specifying what data fields to extract and what constitutes a correct value for each. This schema drives prompt engineering, validation criteria, and downstream analysis.
- **Invest in iterative validation.** System accuracy improves substantially through repeated cycles of extraction, manual review, and prompt refinement. Building this iterative

workflow into the project plan - rather than treating validation as a final step - produces better results and reduces rework.

- **Evaluate the data security trade-off explicitly.** On-premise deployment provides strong data governance but limits model selection. Organizations should assess whether their data sensitivity requirements necessitate local hosting or whether approved cloud environments offer an acceptable alternative with potentially higher model performance.
- **Plan for integration, not replacement.** LLM-based extraction techniques are most effective when combined with existing data management workflows. The extracted data should feed into established quality assurance processes and analytical frameworks rather than bypassing them.

The gap between data collected and data used in energy programs is a persistent challenge. The experience described in this paper suggests that LLM-based tools, when carefully developed and validated, can help narrow that gap – yielding more complete data, more supportable savings estimates, and better visibility into program performance.

Acknowledgments

The authors would like to acknowledge several contributors to this work. From DNV, we thank Ben Jones, Kora Dreffs, and Praga Meyyappan for their contributions to the development and review of this paper, as well as Jacob Moul and Xinyi Gu for their work on the underlying tool. From NYSERDA, we appreciate the AI Governance team for their review of this paper and collaboration on AI approvals, including Osama Sehgol, Laura Geel, and Rebecca Gagnon. We also thank the NYSERDA RTEM program staff, Cody Glavey-Weiss and Michael Reed, for their collaboration on this project.

AI Disclosure: AI-assisted tools used: Custom, purpose-built tool. Used for: brainstorming, expansion of human written content, editing. Extent: light editing, drafting and proposing multiple versions of sections for brainstorming purposes. Human review: The author reviewed and edited throughout to ensure the submission reflects actual work done.

References

Brown, Tom, et al. "Language Models are Few-Shot Learners." *Advances in neural information processing systems* 33 (2020): 1877–1901.

Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019.

Howard, J., and S. Ruder. 2018. "Universal Language Model Fine-tuning for Text Classification." arXiv preprint arXiv: 1801.06146. <https://arxiv.org/abs/1801.06146>.

Jurafsky, D., and J. H. Martin. 2023. *Speech and Language Processing*, 3rd ed. draft. Stanford, CA: Stanford University. <https://web.stanford.edu/~jurafsky/slp3/>.

Khanuja, Apoorv, and Amanda Webb. 2024. "Can LLMs Understand EEMs? Using Large Language Models to Manage Building Energy Efficiency Measure Data." *IBPSA-USA SimBuild 2024 Conference*.

New York State Energy Research and Development Authority (NYSERDA). n.d. Doing Business with NYSERDA. <https://www.nyserda.ny.gov/About/Doing-Business-with-NYSERDA/>.

New York State Office of Information Technology Services (ITS). 2024. Acceptable Use of Artificial Intelligence Technologies. <https://its.ny.gov/system/files/documents/2024/01/nys-p24-001-acceptable-use-of-artificial-intelligence-technologies-_1.pdf>.

Poznanski, J., J. Borchardt, J. Dunkelberger, R. Huff, D. Lin, A. Rangapur, C. Wilhelm, K. Lo, and L. Soldaini. 2025. " olmOCR: Unlocking Trillions of Tokens in PDFs with Vision Language Models." arXiv preprint arXiv: 2502.18443. <https://arxiv.org/abs/2502.18443>.

SEE Action (State and Local Energy Efficiency Action Network). 2012. *Energy Efficiency Program Impact Evaluation Guide*. Washington, DC: U.S. Department of Energy. <https://www.energy.gov/sites/default/files/2014/05/f15/emv_ee_program_impact_guide.pdf>.

Wang, Lei, et al. "A survey on large language model based autonomous agents." *Frontiers of Computer Science* 18.6 (2024): 186345.

Wei, Jason, et al. "Chain-of-thought prompting elicits reasoning in large language models." *Advances in neural information processing systems* 35 (2022): 24824-24837