CarbonPDF: Question-Answering Reasoning Framework for Assessing Product Carbon Footprint in PDF Documents

Anonymous ACL submission

Abstract

Product sustainability reports provide valuable insights into the environmental impacts of a product and are often distributed in PDF format. These reports often include a combination of tables and text, which complicates their analysis. The lack of standardization and the variability in reporting formats further exacerbate the difficulty of extracting and interpreting relevant information from large volumes of documents. In this paper, we tackle the challenge of answering questions related to carbon footprints within sustainability reports available in PDF format. Unlike previous approaches, our focus is on addressing the difficulties posed by the unstructured and inconsistent nature of text extracted from PDF parsing. To facilitate this analysis, we introduce CarbonPDF-QA, an open-source dataset containing question-answering pairs for each document, along with human-annotated answers. Our analysis shows that GPT-4o struggles to answer questions with data inconsistencies. To address this limitation, we propose CarbonPDF, an LLM-based technique specifically designed to answer carbon footprint questions on such datasets. We develop CarbonPDF by fine-tuning Llama 3 with our training data. Our results show that our technique outperforms current state-of-the-art techniques, including question-answering (QA) systems finetuned on table and text data. Our codes and dataset are available here.[1][2]

1 Introduction

With climate change, sustainability reporting is becoming important, requiring companies to disclose the environmental impact of their products (Olivier M. Schwab, 2022; Rodriguez, Isabel and Caglio, Ariela, 2023). This reporting is essential not only for regulatory compliance but also for demonstrating corporate responsibility and transparency to stakeholders. Additionally, emission data extracted from these report enable life cycle assessments and various models that evaluate environmental impacts and ensure adherence to sustainability commitments (Gupta et al., 2022). However, extracting emissions data from these reports remains largely manual and time-consuming. The lack of standardization and the complex format of these reports containing hybrid data — a mix of tables and text — presents significant challenges for automated extraction and analysis.

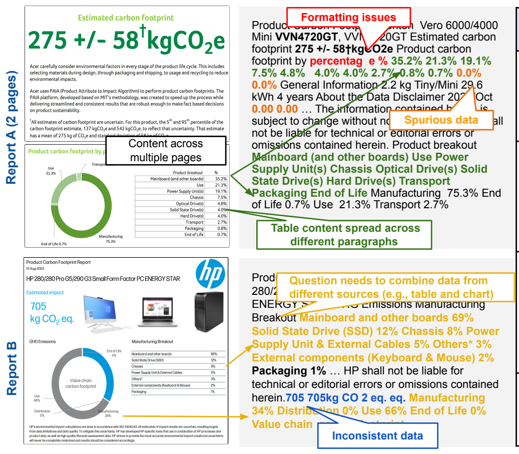
Recent studies have explored the use of question-answering (QA) techniques for analyzing numerical data in hybrid formats, such as tables and text, by framing data analysis as questions (Zhu et al., 2021; Chen et al., 2022). These approaches use language models to interpret hybrid data and perform numerical reasoning, simplifying the extraction and analysis process (Zhu et al., 2024).

However, analyzing hybrid data in carbon sustainability reports presents significant challenges. These difficulties arise because reports are often available as Portable Document Format (PDF) documents, and extracting hybrid data from PDFs is often error-prone. For instance, although tables may appear structured, PDF does not encode this information as tables, unlike HTML or spreadsheets. Instead, PDFs represent tables as a collection of text and lines placed at specific coordinates without any explicit information about rows or columns. This lack of inherent structure makes it difficult to extract and reconstruct tables accurately, as extraction algorithms must infer relationships between text elements based on their positions, which can be complex and unreliable. As shown in Figure 1, the data extracted from a PDF may appear in a different order than expected, even if it looks sequential in the document. This happens because the visual layout of the table does not always match a clear, structured format within the PDF file.

The problem is further complicated by variations

[1] https://anonymous.4open.science/r/carbonpdf-dataset-71C1/

[2] https://anonymous.4open.science/r/carbonpdf-CA2B/


Report A (2 pages)
Report B
Content across multiple pages
Formatting issues
Spurious data
Table content spread across different paragraphs
Question needs to combine data from different sources (e.g., table and chart)
Inconsistent data

Product Carbon Footprint Vero 6000/4000 Mini **VVN4720GT**, VVN4720GT Estimated carbon footprint **275 +/- 58†kgCO2e** Product carbon footprint by percentag e % 35.2% 21.3% 19.1% 7.5% 4.8% 4.0% 4.0% 2.7% 0.8% 0.7% 0.0% 0.0% General Information 2.2 kg Tiny/Mini 29.6 kWh 4 years About the Data Disclaimer 20 /Oct 0.00 0.00 … The information contained h is subject to change without no all not be liable for technical or editorial errors or omissions contained herein. Product breakout **Mainboard (and other boards) Use Power Supply Unit(s) Chassis Optical Drive(s) Solid State Drive(s) Hard Drive(s) Transport Packaging End of Life** Manufacturing 75.3% End of Life 0.7% Use 21.3% Transport 2.7%

Prod 280/2 ENERGY S Emissions Manufacturing Breakout **Mainboard and other boards 69% Solid State Drive (SSD) 12% Chassis 8% Power Supply Unit & External Cables 5% Others* 3% External components (Keyboard & Mouse) 2% Packaging 1%** … HP shall not be liable for technical or editorial errors or omissions contained herein.**705 705kg CO 2 eq. eq. Manufacturing 34% Distribution 0% Use 66% End of Life 0% Value chain**

Reasoning	Question	Answer	Derivation
Word Matching	What are the carbon footprint percentages of the power supply unit and packaging in the VVN4720GT laptop?	[19.1,0.8]	N/A
Max/Min	What is the component with the lowest carbon footprint percentage in the manufacturing breakdown of the VVN4720GT desktop?	{"Packaging": 0.7}	Collect all percentages; Call min( );
Top 3/5	What are the top 3 components with the highest carbon footprint percentages in the manufacturing breakdown of the VVN4720GT laptop?	[{"Mainboard":35.2}, {"Power":19.1}, {"Chassis":7.5}]	Collect all percentages; Sort; Show top 3;
Calculation	What is the carbon footprint of chassis in the VVN4720GT laptop?	20.625	275×7.5%

Figure 1: The CarbonPCF-QA dataset is derived from Product Carbon Footprint (PCF) reports. Examples demonstrate the parsing process used to collect data from the extracted raw text of PDF documents. An accompanying table presents an overview of the various question types included in the dataset.

in how different documents represent tables internally. Content may be spread across different pages or sections, making connections between related data loose or unclear. Additionally, hidden text and numbers encoded within the PDF may not be visible but can be read using programs, resulting in spurious or inconsistent data. Existing state-of-the-art QA systems that handle hybrid data generally assume a structured table format, where the content is free from such anomalies (Zhu et al., 2024). Thus, these systems may struggle when presented with inconsistent content extracted from PDF documents, where table and text data are represented for visual presentation rather than data analysis. Moreover, most QA systems are typically designed to handle reasoning questions over a single table. This limits their effectiveness when dealing with content that spans multiple tables.

In this paper, we address the challenges associated with the problem of hybrid data extracted from sustainability report PDF documents. Our goal is to answer carbon footprint-related questions based on this extracted data from PDF, addressing the challenges posed by inconsistent and loosely connected table and text content. We refer to this data as *inconsistent* because we do not modify it to remove spurious information. Additionally, numbers and text often misalign and can be scattered across multiple paragraphs, complicating direct question answering. To facilitate analysis, we present the CarbonPDF dataset — an open-domain carbon product report in PDF format. This dataset includes a variety of carbon assessment numerical reasoning questions that require extracting information from text, tables, and graphical charts. We developed this dataset through a combination of automated processes and human verification to ensure its accuracy and reliability. To the best of our knowledge, CarbonPDF-QA is the first dataset specifically designed to include inconsistent data containing unstructured tabular and text content.

We also explore how LLMs can effectively manage complex and inconsistent data from sources such as PDFs. In contrast to earlier reasoning methods (e.g.,TAT-LLM (Zhu et al., 2024)), which rely on the assumption of always having the correct context, our approach recognizes the common presence of ambiguous or misleading context in real-world scenarios — a challenge we address in this study. For example, the retrieved document within a RAG system may not always be accurate or relevant. We propose CarbonPDF which generates executable workflows to address a variety of questions. This approach improves the system's reasoning capabilities, allowing it to deliver more accurate answers. In summary, we make the following contributions.

- We introduce CarbonPDF-QA dataset, an open-domain question-answering benchmark for carbon product PDF documents that contain unstructured table and text data. The dataset was created using reports from different companies, with ground truth answers that were manually verified by humans.

- We develop the CarbonPDF model, a QA system designed to handle the complexities of inconsistent or spurious data extracted from PDF documents. Additionally, we created

PDF Statistic		QA Dataset Summary		
Type		Ques. Type	Train	Test
# Company	4	Word Match	7105	1841
# File	1735	Max/Min	1863	486
Avg.char./file	3759	Top 3/5	1242	324
Avg.words/file	539	Calculation	4172	1093
Avg.pages/file	1.76	Total Ques.	14382	3744

Table 1: Statistics of the CarbonPDF-QA dataset

Dataset	#Doc.	Data Source	Content	Ques. Type
SQuAD	736	Text	Wikipedia	RC
HybridQA	13k (tables)	Text Table	Wikipedia	RC
TABFACT	16k (tables)	Text Table	Wikipedia	FV
TAT-QA	182	Text Table	Financial report	RC Arith
CaronPDF -QA	**1735**	**Text Table Chart**	Carbon report	RC Arith

Table 2: QA Dataset comparison. RC: Reading comprehension. FV: Fact Verification. Arith: Arithmetic.

a critic model to select the most relevant responses for accurate answers.

- We conduct extensive experiments and demonstrate that our model outperforms existing state-of-the-art techniques, including RAG and QA systems. Additionally, we perform detailed analyses to showcase the model's capabilities in handling complex numerical reasoning on unstructured table and text data.

2 CarbonPDF-QA Dataset

2.1 Data Collection

Our datasets are derived from computing products' carbon footprint reports, as shown in the left part of Table 1. We collected 1,735 PDF reports from the websites of HP (HP Inc., 2024), Dell (Dell Inc., 2024), Acer (Acer Inc., 2024), and Lenovo (Lenovo Inc., 2024). Each file contains, on average, around 4k characters and 2 pages. In Table 2, we compare our dataset with existing QA datasets. While most datasets focus on text or table-based data, ours also incorporates text extracted from charts. Additionally, our dataset is sourced from carbon reports and offers a novel contribution to QA research by supporting complex arithmetic reasoning tasks. Unlike prior datasets, which are well-formatted and free from issues, our table data is extracted directly from PDFs without formatting. As a result, rows from the same table may overlap across different paragraphs, adding additional complexity to the reasoning process.

To process these reports, we use the PyMuPDF library (PyMuPDF Developers, 2024) to open, parse, and convert the PDF files into text. We developed a Python script to extract product specifications, such as product name, display size, and product weight, as well as carbon-related information, including the total product carbon footprint (PCF) and the carbon footprint percentage of each component in the manufacturing carbon footprint breakdown. To accurately identify and extract relevant text and numerical values, we employ regular expressions, which allow us to efficiently locate and retrieve specific data points from the extracted text (see Appendix A.2). If the regular expression is unable to find the expected pattern or detects multiple values, we discard such documents and exclude them from our dataset. Since the document contains only a single instance of each required data point, this approach ensures accuracy and consistency in data extraction. The extracted text and values are stored in CSV files, which are used to generate the dataset.

2.2 Dataset Preparation

Question Generation The dataset includes various question types, shown in the right part of Table 1. These range from word-matching questions, where answers can be directly extracted from the PDF file, such as the total product carbon footprint or the carbon footprint percentage of a specific component, to more complex questions. The latter requires not only evidence extraction from the PDF document that spans different sections but also arithmetic calculations to derive the final answers. The questions in the dataset focus on various aspects of product carbon footprints, including those related to individual components or multiple components of a product. For each product document, we generate, on average, at least 10 questions that can be answered using information from the PDFs. We wrote a Python script to generate questions based on a predefined question template. A sample template is: "What are the carbon footprints of `[components]` in the `[product_name]` `[product_type]`?" The placeholders are replaced with specific component names (e.g., SSD, display), product names, and product types (e.g., laptop, workstation) from the CSV files, which are extracted from the PDFs.

Each question is paired with a reference document that provides the context for the answer.

Program Generation For each question, the dataset includes a Python program that generates the final answer. These programs are created using predefined templates, each corresponding to a specific question type. The programs consist of simple arithmetic calculations (Appendix A.3), populated with values from the CSV. To ensure accuracy, we use a script to execute these programs and verify that the final output matches the correct answer in the CSV file. Our dataset also includes questions that may yield multiple answers, such as queries about the carbon footprint of both an HDD and a chassis. In such cases, the final answer is structured as a list, with the order of components corresponding to the sequence specified in the question. Finally, the entire dataset is split into a training set and a test set with an 80/20 ratio, ensuring that each set is derived from entirely separate documents. This guarantees no overlap of carbon report documents between the two sets.

Data Validation To validate the data quality, we enlisted five students to verify the ground truth data. These students conducted the data validation as part of their class projects, which involved developing visualization tools for analyzing carbon reports. We divided the students into two teams, distributing the PDFs equally, and extracted content among them for validation. Each team, consisting of at least two students, was tasked with verifying each extracted value. The verification process combined automated checks with manual review. To automate verification, we implemented Python scripts that calculated whether the sum of component-level carbon footprint percentages in the manufacturing breakdown stayed within 99%–101% of the total manufacturing carbon footprint in the PCF. Values outside this range were manually reviewed to identify potential parsing errors. However, we found that some documents had component percentages that did not sum within the 99%–101% range, so we manually verified and corrected the data. Additionally, for each company, we assess whether a product's PCF deviates by more than 2×MAE from the mean PCF. While we identified some instances with significant deviations, these values were actually correct and present in the documents, and therefore were not discarded. Finally, we validated our program generation process by executing the Python programs and comparing their results with the ground truth available in the CSV file. If the results matched, the program was considered correct. For more details, see Appendix A.4.

3 CarbonPDF Design

3.1 Overview

A key design goal of CarbonPDF is to provide accurate, fact-based answers to user queries. However, prior work shows that state-of-the-art LLMs often struggle with maintaining factual accuracy (Mallen et al., 2022). To mitigate this issue, we incorporate Retrieval Augmented Generation (RAG) techniques into our design strategy, leveraging their success in reducing factual errors in knowledge-intensive tasks. Unlike prior work such as TAT-LLM (Zhu et al., 2024), which assumes that the retrieved context from the dataset is always correct, our approach acknowledges that the output from retrievers may not always provide the correct context — a challenge we address in this work.

Figure 2 illustrates our approach's key components and overall workflow. For a given question, the retriever first retrieves relevant documents from the PDF database. Next, the critic model finds the most relevant PDF document containing the answer. The retrieved reference text, which includes unstructured PDF data, is then combined with the question and a set of instructions to guide the carbon model in its reasoning process to derive the final answer. The program-based reasoner generates a program that produces the final answer. A program interpreter then executes the necessary calculations to generate the response. For additional details on the instruction prompt template, please refer to the Appendix A.1.

3.2 Retriever

Document retrieval has traditionally identified relevant documents through keyword matching. Recently, neural network-based approaches, such as Contriever (Izacard et al., 2021), which utilize neural embeddings for retrieval, have been introduced and employed in models like Self-RAG (Asai et al., 2023). SBERT embedding is another neural embedding commonly used for similar texts matching, which is adapted in CaML (Balaji et al., 2023). However, our work uses Term Frequency-Inverse Document Frequency (TF-IDF) embedding as has a high hitrate for our use case.

Figure 3 compares the performance of different retrievers on the test set. For each question, we consider it a hit if the correct document appears in

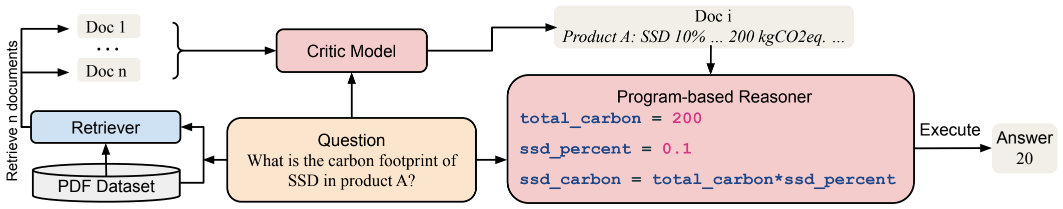


Figure 2: Overview of CarbonPDF.

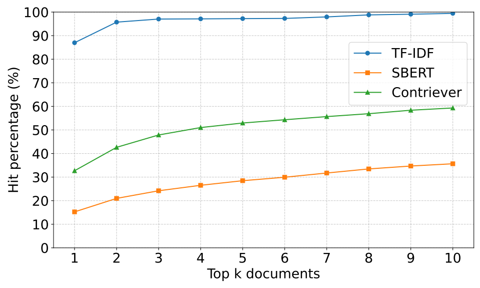


Figure 3: Comparison of retrievers

the top-k retrieved documents. The figure shows that as k increases, the likelihood of retrieving the correct document within the top-k results also increases. TF-IDF achieves the highest match percentage, reaching nearly 100% at k=10, indicating that a relevant document is almost always found within the top 10 results. While we use TF-IDF at present, our approach is flexible enough to support other retrieval techniques.

To retrieve the relevant documents, we first convert the entire document corpus into TF-IDF embeddings using the *sklearn.TfidfVectorizer* function. This function transforms each document into a vector of numerical values, where each dimension represents a term in the corpus, and the value in each dimension corresponds to the term's TF-IDF score. When CarbonPDF receives a question, it also converts this question into a TF-IDF vector. We then compute the cosine similarity between the question vector and the TF-IDF vectors of the documents in our corpus. This similarity metric produces a ranking of documents based on their relevance to the question. The retriever $r(n)$ then selects n documents with the highest similarity scores and passes them to the critic model.

3.3 Critic Model

The goal of the critic model $\mathcal{C}$ is to identify the document that provides the most relevant context for answering a given question from the top-n retrieved documents. Let $\mathcal{D}_{critic} = \{(q_i, r(n))\}_{i=1}^{N}$ be our training dataset, consisting of N question samples, where each question q_i is paired with n documents $r(n)$. In $\mathcal{D}_{critic}$, we label the ground truth document d from the n retrieved documents for each question q_i. We fine-tune the critic on $\mathcal{D}_{critic}$ using a standard conditional language modeling objective, maximizing the likelihood:

$$\max_{\mathcal{C}} \quad \mathbb{E}_{((q,r(n)),d)\sim\mathcal{D}_{critic}} \log p_{\mathcal{C}}(d|q, r(n)) \quad (1)$$

The critic model $\mathcal{C}$ identifies and selects the most appropriate document d for answering question q.

3.4 Program-based Reasoner

LLMs often struggle with reasoning questions that involve complex calculations (Lewkowycz et al., 2022). Recently, program-aided language models have demonstrated effectiveness in overcoming these challenges by leveraging programmatic reasoning (Gao et al., 2023). This approach improves the model's ability to perform complex calculations and produce accurate answers. Building on this insight, we finetune CarbonPDF LLM to generate a Python program to compute the results based on the reference.

The final program generated by CarbonPDF varies based on the type of question and the document's content. Some carbon documents provide the total carbon footprint along with lifecycle breakdowns (e.g., manufacturing, end-of-life, and transport) and detailed breakdowns for individual components (e.g., HDDs, chassis). Other documents may report the carbon footprint of individual components directly without offering a comprehensive lifecycle breakdown.

Similarly, the complexity of the questions may also affect the program generated. For instance, questions requiring detailed calculations for individual components, which depend on factors like the manufacturing process, involve more complex reasoning. To handle such complexity, CarbonPDF employs a multistep approach in its generated programs. Unlike single-step calculations, we store intermediate values in variables and then perform necessary multiplications to derive the answer. In situations with multiple answers, CarbonPDF produces the final answers as a list, maintaining the specified question order.

3.5 Training

We train our CarbonPDF model by fine-tuning Llama 3_{8B} (Meta AI, 2023) using two NVIDIA RTX 6000 Ada GPUs for 2.5 days. The learning rate is set to 2.5e-5, with a per-device training batch size of 8 and gradient accumulation steps of 4. The total number of training epochs is 4. We employ Low-Rank Adaptation (LoRA) (Hu et al., 2021) during training, with 4-bit quantization. The paged Adam optimizer (Kingma, 2014), adapted for quantization, is used to further optimize the training process. To prepare the inputs for training, we compute the largest token length in our dataset and create custom tokenization with left padding. We set the End-of-Sequence token as the pad token to ensure compatibility with causal language models.

4 Evaluation Methodology

4.1 Baseline Techniques

Baselines without LLM. ACT (Gupta et al., 2022) and CaML (Balaji et al., 2023) are model-based carbon estimation techniques that do not rely on large language models (LLMs). ACT calculates the carbon footprint of each component within computer systems using detailed product manufacturing information. On the other hand, CaML associates product names with North American Industry Classification System (NAICS) codes to estimate a carbon footprint per dollar at the industry sector level. Given that CaML consistently provides the same estimate for 'Electronic Computer Manufacturing,' we assume a default price for computing products to calculate the overall carbon footprint. These models are evaluated by estimating and comparing the total carbon footprint of products against ground truth values.

Few-shot without RAG We use Gemini-2.0-flash as our few-shot baseline without RAG (DeepMind, 2024). The model is accessed via the Google API.

Few-shot with RAG We evaluate Self-RAG (Asai et al., 2023), DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025), GPT-4o (Hurst et al., 2024), and Gemini-2.0-flash as few-shot baselines with RAG. Self-RAG retrieves relevant documents and guides the LLM to generate the best possible answer. We provide the exact reference text along with the question to DeepSeek-R1, GPT-4o, and Gemini-2.0-flash to assess their performance on CarbonPDF-QA dataset without fine-tuning.

Fine-tuned with RAG We fine-tuned Llama 3_{8B}, TAT-LLM (Zhu et al., 2024), and our CarbonPDF on the dataset. TAT-LLM was originally fine-tuned on Llama 2_{7B}. For a fair comparison, we fine-tuned it on Llama 3_{8B}. Note that TAT-LLM employs a step-wise pipeline and requires well-formatted tables and text to answer questions. Since our data lacks formatted tables, we used the text section within their prompt template for fine-tuning.

Techniques	RMSE	MAE	EM
Baselines without LLM			
ACT (Gupta et al., 2022)	486.83	323.80	0.00
CaML (Balaji et al., 2023)	435.89	230.70	0.28
Few-shot without RAG			
Gemini-2.0-flash (DeepMind, 2024)	76.61	71.98	0.51
Few-shot with RAG			
Self-RAG$_{Retriever}$ (Asai et al., 2023)	43.55	35.44	22.70
DeepSeek-R1$_{Retriever}$ (Guo et al., 2025)	28.20	21.98	35.87
GPT-4o$_{Retriever}$ (Hurst et al., 2024)	13.37	10.20	49.20
Gemini-2.0-flash$_{Retriever}$	10.37	8.12	56.52
Fine-tuned with RAG			
Llama $3_{8B\ Retriever}$ (Meta AI, 2023)	4.01	3.43	59.70
TAT-LLM$_{Retriever}$ (Zhu et al., 2024)	2.92	2.52	64.13
CarbonPDF$_{Retr+Critic}$	**0.78**	**0.69**	**93.70**

Table 3: Baseline performance comparison.

4.2 Metrics

We use the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) to measure the numerical accuracy of the predicted answers from the gold answers. We also use Exact Match (EM) to measure how often the predicted values exactly match the gold standard (Rajpurkar et al., 2016). For questions with multiple answers, the model is required to match all gold standard answers exactly, including their order, to be considered correct.

5 Results

5.1 Baseline Performance

Table 3 compares CarbonPDF with other baseline techniques. Our technique consistently outperforms all baselines. Model-based approaches such as ACT and CaML show high RMSE and

Type	#Question	RMSE	MAE	EM
Word Match	1841	1.02	0.97	94.41
Max/Min	486	0.54	0.46	95.27
Top 3/5	324	0.40	0.30	92.59
Calculation	1093	0.61	0.43	92.13

Table 4: Performance on different question types.

#Answer	#Question	RMSE	MAE	EM
1	1148	1.57	1.57	95.91
2	878	0.37	0.27	94.53
3	857	0.43	0.28	93.00
4	699	0.51	0.35	90.99
5	162	0.46	0.32	88.89

Table 5: Performance on multi-answer questions.

Task	RMSE	MAE	EM
Retriever+Few-shot	833.09	668.10	23.77
Retriever+Fine-tuned	2.45	2.19	83.92
Retriever+Critic +Fine-tuned w/o Program Reasoning	2.39	1.96	66.83
Retriever+Critic +Fine-tuned (CarbonPDF)	**0.78**	**0.69**	**93.70**

Table 6: Ablation analysis of CarbonPDF.

MAE due to their reliance on general carbon estimates and default values, which lack customization for specific questions. Consequently, their Exact Match (EM) scores are close to zero. LLM baseline without RAG — Gemini-2.0-flash — also exhibits significant errors, with EM values below 1%. This underscores the need for data augmentation to improve the accuracy of model predictions.

While RAG-based approaches show improved performance, they still have lower performance compared to our model. Gemini-2.0-flash performs the best among the few-shot baselines, as it is the latest LLM with a reasoning model. We also compared our technique to Self-RAG, which identifies relevant text and uses them to answer questions. However, the technique struggles with numerical reasoning questions.

Fine-tuned baselines tend to have the best performance. Our model surpasses fine-tuned reasoning-based models like TAT-LLM, which rely on well-formatted tables and text input. This highlights the challenges current QA models face in handling unstructured and inconsistent data in our dataset.

5.2 Performance on Different Question Types

Table 4 summarizes our CarbonPDF performance across different question types shown in Figure 1. The model performs well across all these question types. Max/min questions yield slightly better results, likely due to their simpler reasoning and fewer question variants, which reduce the potential for errors. Word-matching questions exhibit the highest RMSE and MAE due to their larger question count and greater numeric variation, particularly in absolute carbon footprint and percentage-based questions. Calculation questions have the lowest EM, as they involve more complex reasoning, increasing the potential for errors.

5.3 Performance on Multi-answer Questions

We now analyze the impact of questions requiring multiple answers. Table 5 shows the results as we vary the number of answers per question. As the number of required answers increases, the EM score gradually decreases due to the added complexity in evidence extraction and carbon modeling. Most multi-answer questions involve a mix of calculation and word-matching types, which can reduce accuracy. Single-answer questions exhibit higher RMSE and MAE due to their larger volume and the prevalence of queries regarding the total product carbon footprint, which typically has the highest absolute value in the dataset. This leads to greater numerical variation, contributing to the higher error rates. In summary, CarbonPDF performs well across questions with varying numbers of answers. Nonetheless, complex questions with fewer answers tend to show better performance.

5.4 Ablation Study

We conduct an ablation study to evaluate the impact of various components. First, we analyze the effectiveness of few-shot learning in our pipeline. Few-shot learning involves providing a small number of examples at inference time to guide the desired completion, which has been shown to perform well in some tasks (Brown, 2020; Gautier et al., 2022). Thus, we replace the fine-tuning step in CarbonPDF with a few-shot approach, where we provide 4 examples to derive the program for a given question and reference. Table 6 shows the results of this approach. We observe that the few-shot technique does not perform well on our dataset. This is consistent with prior work that indicates few-shot methods struggle with complex reasoning tasks (Brown, 2020; Asai et al., 2023).

The results in the second row are evaluated without the critic model. Comparing them with the last row (CarbonPDF), which utilizes the critic model, we observe that CarbonPDF achieves approximately 10% higher EM while also reducing RMSE and MAE. This shows that the critic model enhances overall accuracy by providing relevant documents to CarbonPDF.

In addition, we evaluated the LLM fine-tuned without the program-based reasoning step. In this variation, we trained it to generate the final answer directly without using the program. CarbonPDF with program-based reasoning improves EM by approximately 27% compared to the version without it. This underscores the importance of program-based reasoning, as it shifts the computational burden to the program interpreter, leading to significantly more accurate arithmetic results.

6 Related Work

QA Datasets There are numerous existing QA datasets. For structured data, such as Knowledge Base (KB) and tables, notable examples include Complex Web Questions (Talmor and Berant, 2018) and TabFact (Chen et al., 2019). Text-based QA datasets include SQuAD (Rajpurkar et al., 2016), SearchQA (Dunn et al., 2017), and DROP (Dua et al., 2019). For multi-hop QA, there are HOTPOTQA (Yang et al., 2018) and HybridQA (Chen et al., 2020). Hybrid datasets also include TAT-QA (Zhu et al., 2021), which integrates tabular and textual content in the financial domain, and TAT-LLM (Zhu et al., 2024), which utilizes well-formatted tabular and textual data to train LLMs on discrete reasoning. Our CarbonPDF-QA dataset stands apart by including inconsistent or spurious data extracted from PDFs, reflecting the challenges of real-world document processing. Furthermore, the tables in our dataset are not well-structured, with column values that may span different paragraphs, complicating data analysis.

QA Reasoning Numerous studies have explored question-answering (QA) systems, including those that use Retrieval-Augmented Generation (RAG) approaches to guide large language models (LLMs) in answering questions (Izacard et al., 2021; Wei et al., 2022; Gao et al., 2023; Guu et al., 2020; Lewis et al., 2020; Asai et al., 2023). Despite these advancements, LLMs often struggle with complex reasoning tasks, particularly numerical reasoning. Recent research has focused on numerical reasoning over tabular, text, and chart data (Zhu et al., 2021; Kim et al., 2021; Li et al., 2022; Zhu et al., 2022; Zhou et al., 2022; Li et al., 2023; Wei et al., 2023; Han et al., 2023), including financial reports (Chen et al., 2021; Yuan et al., 2024). However, the application of these techniques to real-world unstructured and inconsistent data, such as that extracted from PDFs, remains relatively unexplored. Our work addresses this gap by addressing the challenges of inconsistent data in the context of sustainability reports.

Carbon Footprint Analysis Companies frequently employ Lifecycle Assessment (LCA) methodologies to evaluate the environmental impact of their products across the entire lifecycle, from raw material extraction to disposal (Hauschild et al., 2018). Tools like GaBi and SimaPro are widely used for conducting these assessments, producing detailed analyses that are often integrated into sustainability reports (Silva et al., 2017). However, LCA methods require significant manual effort and depend heavily on detailed input data, which companies often do not publicly disclose. Recent advancements have focused on automating carbon footprint analysis through data-driven approaches (Gupta et al., 2022; Balaji et al., 2023). These approaches typically utilize publicly available data, such as industry averages or estimates, which tend to be less accurate than the more precise, company-specific data used in traditional LCA methods. The application of question-answering (QA) systems within the sustainability domain is relatively nascent. To the best of our knowledge, our work is the first to apply QA systems for carbon footprint assessments within sustainability reports.

7 Conclusion

We introduce CarbonPDF-QA, an open-source product carbon footprint QA dataset with comprehensive annotations, comprising around 18k questions of various types. We leverage this dataset to fine-tune CarbonPDF, enabling it to perform reasoning with reference augmentation and generate accurate results through our program-based reasoning approach. We demonstrate its effectiveness through extensive experiments and show that CarbonPDF outperforms the baselines on all metrics. We anticipate that the CarbonPDF-QA dataset and CarbonPDF model will serve as valuable benchmarks and baselines, fostering the development of advanced QA models for PDF documents and carbon footprint estimation.

Limitations

One key limitation of current PDF parsing methods is their difficulty in handling data presented in graphical forms, such as pie charts or bar graphs. These visual elements are often used to convey complex data, but traditional parsing techniques that focus on text extraction struggle with purely graphical content. This limitation poses a significant challenge, as crucial information within these visual elements can be missed or misinterpreted. Since CarbonPDF primarily relies on text data, it cannot effectively answer questions based on content that combines graphs and text. However, our technique remains useful when numerical data is presented alongside these graphs, as it can still extract and analyze this information. In the future, we plan to explore multimodal large language models (LLMs) to perform reasoning on both text and visual data.

Although CarbonPDF can handle various types of questions, there are still limitations. For example, if CarbonPDF is asked about the carbon footprint of processors, but the exact term "processor" does not appear in the text, the system might not provide the expected response, even if related terms like "mainboard" are present. This occurs because the model is not capable of understanding synonyms or recognizing that certain components are subsets of larger systems. A key question for future research is whether large language models (LLMs) can be trained to handle such nuances, improving their reasoning ability to understand related terms and components within a broader context.

Ethics Statement

In this work, we first highlight the challenges of processing PDF documents using examples from our dataset. We then discuss how we collected and processed our CarbonPDF-QA dataset. Following this, we propose our CarbonPDF model, which fine-tunes an LLM (Llama 3_{8B}) to perform program-based reasoning on unstructured PDF data. Our model is developed using open-source tools and datasets to aid in understanding and processing of product sustainability reports.

Data Collection and Licensing

We collected the product carbon footprint reports that are publicly available. Our CarbonPDF-QA dataset consists of content extracted from these reports. We plan to release the data under CDLA-Permissive[3] license. This will allow broad access and use of our dataset, allowing recipients to modify and share the data freely.

During the data collection, all students voluntarily participated in the project. We proposed a Capstone project, which involved the collection, visualization, and analysis of Product Carbon Footprint (PCF) data. The Capstone project description outlined the tasks students were required to complete, and students received credits for their contributions. Students reviewed the available Capstone project descriptions for that semester, and some students selected to work on this project due to their interest in data analysis. We held weekly meetings to assess progress and provide guidance on the code/data. An outcome of this Capstone project was this dataset, which was used for this project.

Potential Risk

While CarbonPDF demonstrates high accuracy, there is still room for improvement. It may produce inaccurate results for certain questions. This may lead to misleading conclusions or errors in evaluating environmental impacts. Since our approach relies on automatic sustainability report analysis, it may introduce potential biases in data or carry risks of misuse. For instance, inaccurate results could lead to misleading conclusions about a company's environmental impact. Additionally, the accuracy of emissions data depends on the PDF source. To mitigate this, companies should evaluate their environmental impact using multiple reliable carbon footprint datasets to ensure a more accurate and comprehensive assessment.

References

Acer Inc. 2024. Acer library document collection (product carbon footprint). Accessed: 2024-08-12.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Self-reflective retrieval augmented generation. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Bharathan Balaji, Venkata Sai Gargeya Vunnava, Geoffrey Guest, and Jared Kramer. 2023. Caml: Carbon footprinting of household products with zero-shot semantic text similarity. In *Proceedings of the ACM Web Conference 2023*, pages 4004–4014.

[3]https://cdla.dev/permissive-2-0/

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint ArXiv:2005.14165*.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2019. Tabfact: A large-scale dataset for table-based fact verification. *arXiv preprint arXiv:1909.02164*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. 2020. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. *arXiv preprint arXiv:2004.07347*.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, et al. 2021. Finqa: A dataset of numerical reasoning over financial data. *arXiv preprint arXiv:2109.00122*.

Zhiyu Chen, Shiyang Li, Charese Smiley, Zhiqiang Ma, Sameena Shah, and William Yang Wang. 2022. Convfinqa: Exploring the chain of numerical reasoning in conversational finance question answering. *arXiv preprint arXiv:2210.03849*.

Google DeepMind. 2024. Gemini-2.0-flash. `https://deepmind.google/technologies/gemini/flash/`. Accessed: 2025-02-14.

Dell Inc. 2024. Dell library document collection (product carbon footprint). Accessed: 2024-08-12.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR.

Izacard Gautier, Lewis Patrick, Lomeli Maria, Hosseini Lucas, Petroni Fabio, Schick Timo, Dwivedi-Yu Jane, Joulin Armand, Riedel Sebastian, and Grave Edouard. 2022. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv: 2208.03299*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Udit Gupta, Mariam Elgamal, Gage Hills, Gu-Yeon Wei, Hsien-Hsin S Lee, David Brooks, and Carole-Jean Wu. 2022. Act: Designing sustainable computer systems with an architectural carbon modeling tool. In *Proceedings of the 49th Annual International Symposium on Computer Architecture*, pages 784–799.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Yucheng Han, Chi Zhang, Xin Chen, Xu Yang, Zhibin Wang, Gang Yu, Bin Fu, and Hanwang Zhang. 2023. Chartllama: A multimodal llm for chart understanding and generation. *arXiv preprint arXiv:2311.16483*.

Michael Z Hauschild, Ralph K Rosenbaum, Stig Irving Olsen, et al. 2018. *Life cycle assessment*, volume 2018. Springer.

HP Inc. 2024. HP library document collection (product carbon footprint). Accessed: 2024-08-12.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Geewook Kim, Teakgyu Hong, Moonbin Yim, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2021. Donut: Document understanding transformer without ocr. *arXiv preprint arXiv:2111.15664*, 7(15):2.

DP Kingma. 2014. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lenovo Inc. 2024. Lenovo library document collection (product carbon footprint). Accessed: 2024-08-12.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

Moxin Li, Fuli Feng, Hanwang Zhang, Xiangnan He, Fengbin Zhu, and Tat-Seng Chua. 2022. Learning to imagine: Integrating counterfactual thinking in neural discrete reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 57–69.

Xiao Li, Yin Zhu, Sichen Liu, Jiangzhou Ju, Yuzhong Qu, and Gong Cheng. 2023. Dyrren: A dynamic retriever-reranker-generator model for numerical reasoning over tabular and textual data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 13139–13147.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*.

Meta AI. 2023. Llama 3 8b. Accessed: 2024-08-10.

Olivier M. Schwab. 2022. Sustainability reporting: why it's important. [Online; accessed 14-August-2024].

PyMuPDF Developers. 2024. PyMuPDF documentation. Accessed: 2024-08-08.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Rodriguez, Isabel and Caglio, Ariela. 2023. Tackling the sustainability reporting challenge. a policy guide. [Online; accessed 14-August-2024].

Diogo Silva, A Oliveira Nunes, Aparecida da Silva Moris, Cassiano Moro, and Thiago Oliveira Rodrigues Piekarski. 2017. How important is the lca software tool you choose comparative results from gabi, openlca, simapro and umberto. In *Proceedings of the VII Conferencia Internacional de Análisis de Ciclo de Vida en Latinoamérica, Medellin, Colombia*, pages 10–15.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yifan Wei, Fangyu Lei, Yuanzhe Zhang, Jun Zhao, and Kang Liu. 2023. Multi-view graph representation learning for answering hybrid numerical reasoning question. *arXiv preprint arXiv:2305.03458*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Ziqiang Yuan, Kaiyuan Wang, Shoutai Zhu, Ye Yuan, Jingya Zhou, Yanlin Zhu, and Wenqi Wei. 2024. Finllms: A framework for financial reasoning dataset generation with large language models. *arXiv preprint arXiv:2401.10744*.

Yongwei Zhou, Junwei Bao, Chaoqun Duan, Youzheng Wu, Xiaodong He, and Tiejun Zhao. 2022. Unirpg: Unified discrete reasoning over table and text as program generation. *arXiv preprint arXiv:2210.08249*.

Fengbin Zhu, Wenqiang Lei, Fuli Feng, Chao Wang, Haozhou Zhang, and Tat-Seng Chua. 2022. Towards complex document understanding by discrete reasoning. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4857–4866.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. *arXiv preprint arXiv:2105.07624*.

Fengbin Zhu, Ziyang Liu, Fuli Feng, Chao Wang, Moxin Li, and Tat-Seng Chua. 2024. Tat-llm: A specialized language model for discrete reasoning over tabular and textual data. *arXiv preprint arXiv:2401.13223*.

A Appendix

A.1 Prompt templates

The template for CarbonPDF training prompt is shown in List1. For testing, we use the prompt in List 2 and ask the model to complete it. List 3 displays the few-shot prompt template. List 4 and List 5 are the prompt templates designed to train and test the model without program-based reasoning.

A.2 Regular expression templates

List 6 presents example regular expressions used for parsing carbon reports from HP. In the PCF reports, we observed that the component carbon footprint percentage typically follows the component name or its abbreviation and ends with a % sign. We also account for decimal points when capturing float values.

A.3 Program templates

List 7 presents the program template for answering calculation-based questions about HP products. Since the total PCF value is always required, it is assigned in the first line of the program. Next, we check whether the question pertains solely to PCF. If so, we add the answer line and terminate the program. Otherwise, we proceed by retrieving the manufacturing percentage within the PCF, as it is always necessary for HP products. We then determine whether the question focuses exclusively on the manufacturing carbon footprint. If this is the case, we compute the product of the manufacturing percentage and the PCF, append the result, and terminate the program. Otherwise, we iterate through all relevant components, summing their respective percentages from the manufacturing breakdown. The program then computes the product of the PCF, the manufacturing percentage, and each component's percentage individually. Finally, the computed values are returned as a list. List 8 shows an example prompt of CarbonPDF, with the question, document, and program.

A.4 Data validation

Figure 4 illustrates our data validation process. The students were divided into two teams, each responsible for running automated verification scripts on the data they collected. As discussed in Section 2.2, these scripts check whether the sum of component percentages falls within 99%–101% and whether a product's PCF value deviates by more than 2× MAE from the mean PCF. Samples that passed both checks were considered validated data. In total, 24 samples failed the sum check, and 56 samples failed the PCF deviation check. Both teams manually reviewed these failed cases. For the 56 PCF deviation failures, the values were verified to be correct and present in the documents, so they were retained. For the 24 sum-check failures, the teams manually inspected and corrected the data when issues were identified. Additionally, these samples were reviewed by the author and revalidated before incorporating them into the final validated dataset.

A.5 Product carbon footprint report examples

Figures 5, 6, 7, and 8 present example PCF reports from HP, Dell, Acer, and Lenovo. Generally, these reports include the company name, product name, total PCF value, value chain carbon footprint breakdown, and manufacturing carbon footprint breakdown on components. The percentage breakdown may be displayed in either a pie chart or a table within the PDF file. Additionally, the reports specify the techniques or standards used to estimate carbon emissions. The reports also provide supplementary details such as product lifetime, usage location, and weight. Lenovo reports include an additional breakdown of transportation methods by percentage.

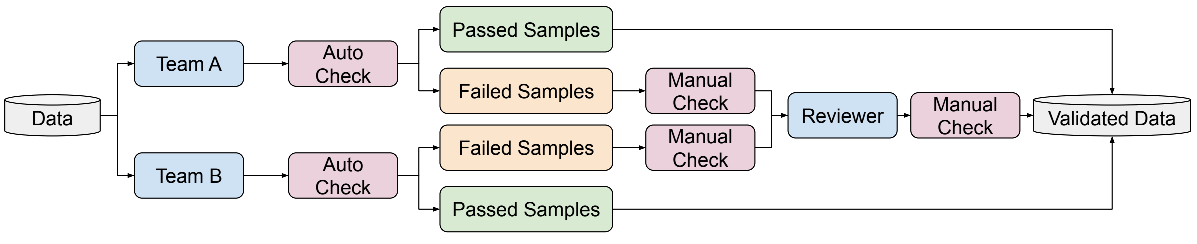
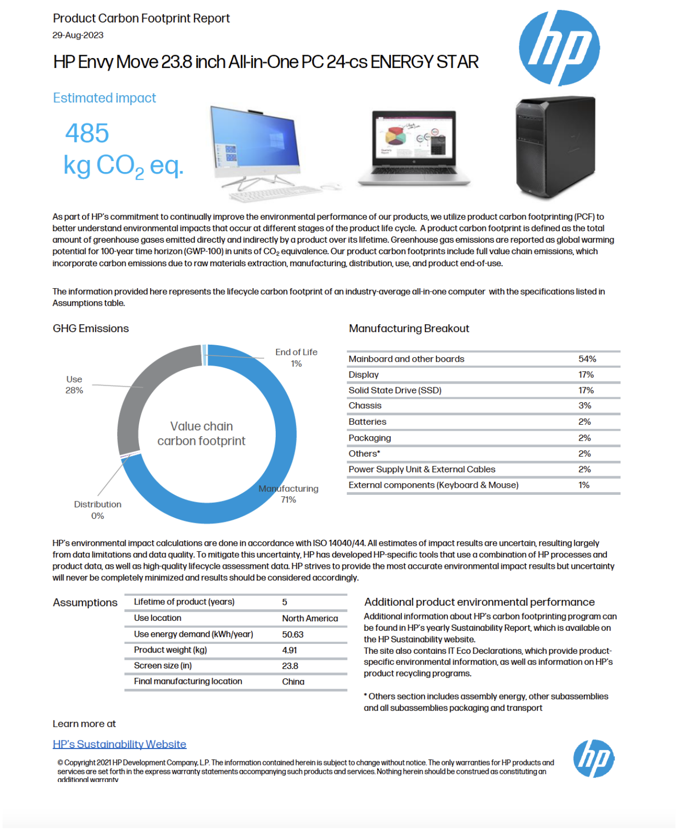


Figure 4: Data validation workflow.



Product Carbon Footprint Report

29-Aug-2023

HP Envy Move 23.8 inch All-in-One PC 24-cs ENERGY STAR

Estimated impact

485

kg CO_2 eq.

As part of HP's commitment to continually improve the environmental performance of our products, we utilize product carbon footprinting (PCF) to better understand environmental impacts that occur at different stages of the product life cycle. A product carbon footprint is defined as the total amount of greenhouse gases emitted directly and indirectly by a product over its lifetime. Greenhouse gas emissions are reported as global warming potential for 100-year time horizon (GWP-100) in units of CO_2 equivalence. Our product carbon footprints include full value chain emissions, which incorporate carbon emissions due to raw materials extraction, manufacturing, distribution, use, and product end-of-use.

The information provided here represents the lifecycle carbon footprint of an industry-average all-in-one computer with the specifications listed in Assumptions table.

GHG Emissions

Value chain carbon footprint

End of Life 1%

Use 28%

Distribution 0%

Manufacturing 71%

Manufacturing Breakout

Mainboard and other boards	54%
Display	17%
Solid State Drive (SSD)	17%
Chassis	3%
Batteries	2%
Packaging	2%
Others*	2%
Power Supply Unit & External Cables	2%
External components (Keyboard & Mouse)	1%

HP's environmental impact calculations are done in accordance with ISO 14040/44. All estimates of impact results are uncertain, resulting largely from data limitations and data quality. To mitigate this uncertainty, HP has developed HP-specific tools that use a combination of HP processes and product data, as well as high-quality lifecycle assessment data. HP strives to provide the most accurate environmental impact results but uncertainty will never be completely minimized and results should be considered accordingly.

Assumptions

Lifetime of product (years)	5
Use location	North America
Use energy demand (kWh/year)	50.63
Product weight (kg)	4.91
Screen size (in)	23.8
Final manufacturing location	China

Additional product environmental performance

Additional information about HP's carbon footprinting program can be found in HP's yearly Sustainability Report, which is available on the HP Sustainability website.
The site also contains IT Eco Declarations, which provide product-specific environmental information, as well as information on HP's product recycling programs.

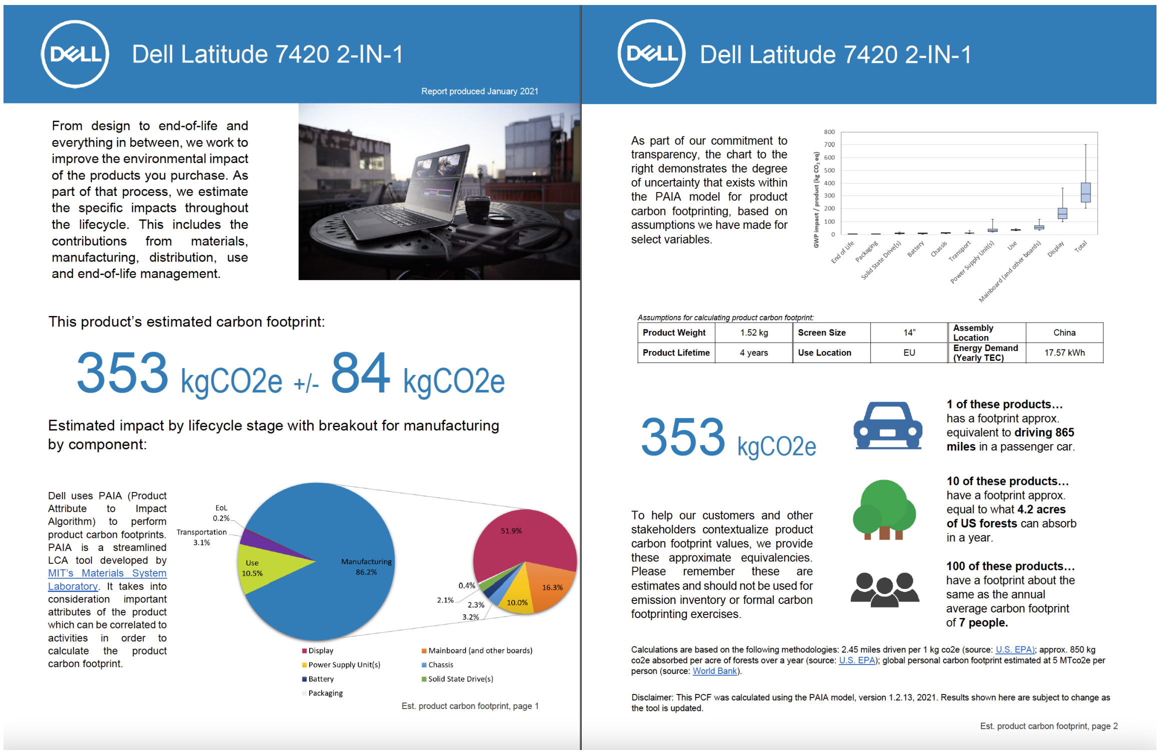
* Others section includes assembly energy, other subassemblies and all subassemblies packaging and transport

Learn more at

HP's Sustainability Website

© Copyright 2021 HP Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty

Figure 5: PCF report example for HP



Dell Latitude 7420 2-IN-1

Report produced January 2021

From design to end-of-life and everything in between, we work to improve the environmental impact of the products you purchase. As part of that process, we estimate the specific impacts throughout the lifecycle. This includes the contributions from materials, manufacturing, distribution, use and end-of-life management.

This product's estimated carbon footprint:

353 kgCO2e +/- 84 kgCO2e

Estimated impact by lifecycle stage with breakout for manufacturing by component:

Dell uses PAIA (Product Attribute to Impact Algorithm) to perform product carbon footprints. PAIA is a streamlined LCA tool developed by MIT's Materials System Laboratory. It takes into consideration important attributes of the product which can be correlated to activities in order to calculate the product carbon footprint.

Est. product carbon footprint, page 1

Dell Latitude 7420 2-IN-1

As part of our commitment to transparency, the chart to the right demonstrates the degree of uncertainty that exists within the PAIA model for product carbon footprinting, based on assumptions we have made for select variables.

Assumptions for calculating product carbon footprint:

Product Weight	1.52 kg	**Screen Size**	14"	**Assembly Location**	China
Product Lifetime	4 years	**Use Location**	EU	**Energy Demand (Yearly TEC)**	17.57 kWh

353 kgCO2e

1 of these products… has a footprint approx. equivalent to **driving 865 miles** in a passenger car.

10 of these products… have a footprint approx. equal to what **4.2 acres of US forests** can absorb in a year.

100 of these products… have a footprint about the same as the annual average carbon footprint of **7 people.**

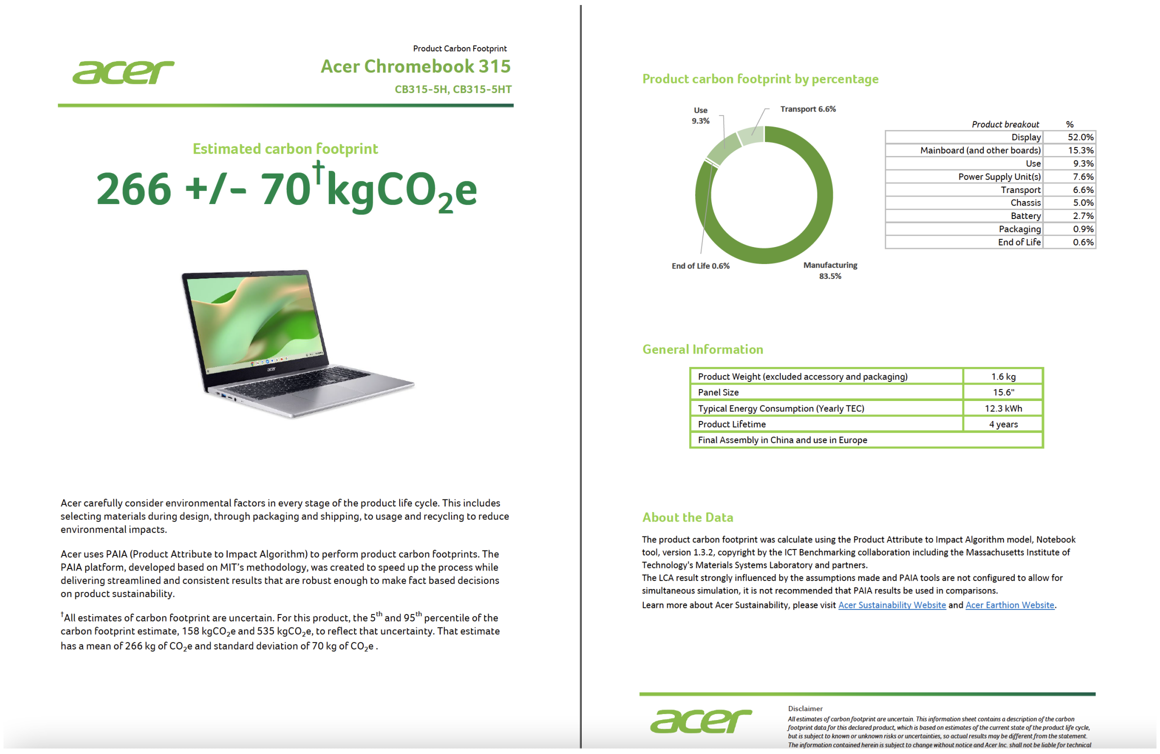
To help our customers and other stakeholders contextualize product carbon footprint values, we provide these approximate equivalencies. Please remember these are estimates and should not be used for emission inventory or formal carbon footprinting exercises.

Calculations are based on the following methodologies: 2.45 miles driven per 1 kg co2e (source: U.S. EPA); approx. 850 kg co2e absorbed per acre of forests over a year (source: U.S. EPA); global personal carbon footprint estimated at 5 MTco2e per person (source: World Bank).

Disclaimer: This PCF was calculated using the PAIA model, version 1.2.13, 2021. Results shown here are subject to change as the tool is updated.

Est. product carbon footprint, page 2

Figure 6: PCF report example for Dell



Product Carbon Footprint

Acer Chromebook 315

CB315-5H, CB315-5HT

Estimated carbon footprint

266 +/- 70[†]kgCO_2e

Acer carefully consider environmental factors in every stage of the product life cycle. This includes selecting materials during design, through packaging and shipping, to usage and recycling to reduce environmental impacts.

Acer uses PAIA (Product Attribute to Impact Algorithm) to perform product carbon footprints. The PAIA platform, developed based on MIT's methodology, was created to speed up the process while delivering streamlined and consistent results that are robust enough to make fact based decisions on product sustainability.

[†]All estimates of carbon footprint are uncertain. For this product, the 5th and 95th percentile of the carbon footprint estimate, 158 kgCO_2e and 535 kgCO_2e, to reflect that uncertainty. That estimate has a mean of 266 kg of CO_2e and standard deviation of 70 kg of CO_2e .

Product carbon footprint by percentage

Product breakout	%
Display	52.0%
Mainboard (and other boards)	15.3%
Use	9.3%
Power Supply Unit(s)	7.6%
Transport	6.6%
Chassis	5.0%
Battery	2.7%
Packaging	0.9%
End of Life	0.6%

General Information

Product Weight (excluded accessory and packaging)	1.6 kg
Panel Size	15.6"
Typical Energy Consumption (Yearly TEC)	12.3 kWh
Product Lifetime	4 years
Final Assembly in China and use in Europe	

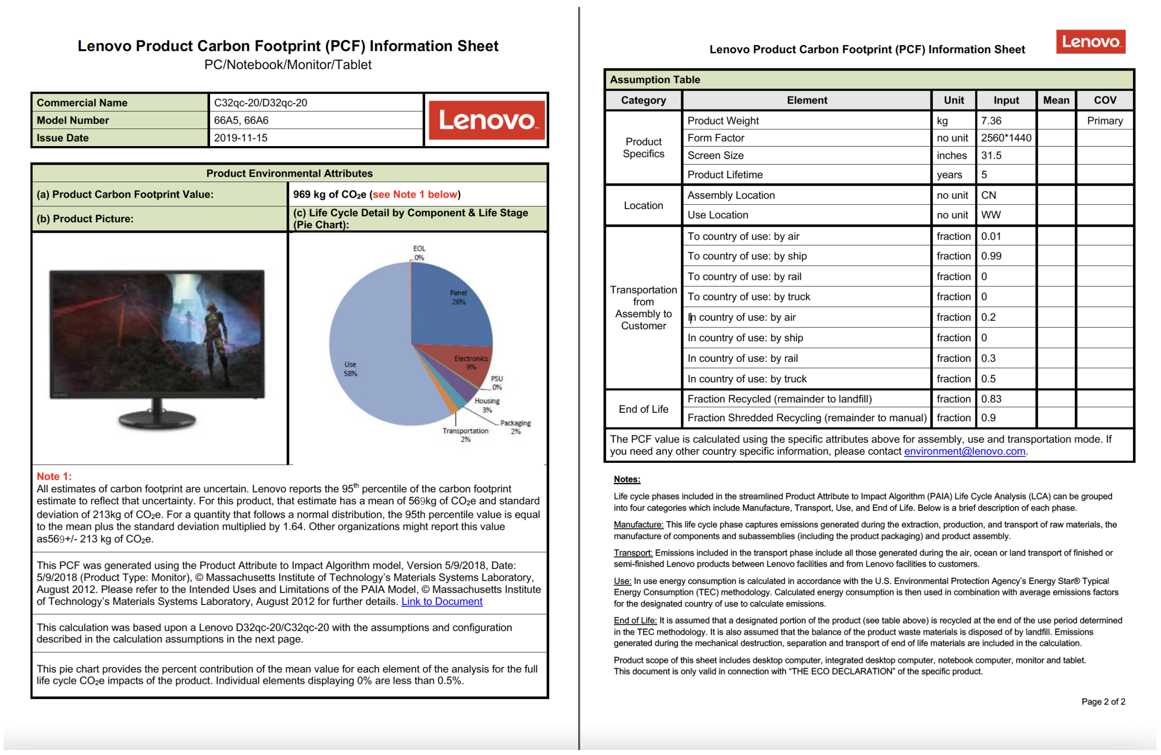
About the Data

The product carbon footprint was calculate using the Product Attribute to Impact Algorithm model, Notebook tool, version 1.3.2, copyright by the ICT Benchmarking collaboration including the Massachusetts Institute of Technology's Materials Systems Laboratory and partners.
The LCA result strongly influenced by the assumptions made and PAIA tools are not configured to allow for simultaneous simulation, it is not recommended that PAIA results be used in comparisons.
Learn more about Acer Sustainability, please visit Acer Sustainability Website and Acer Earthion Website.

Disclaimer

All estimates of carbon footprint are uncertain. This information sheet contains a description of the carbon footprint data for this declared product, which is based on estimates of the current state of the product life cycle, but is subject to known or unknown risks or uncertainties, so actual results may be different from the statement. The information contained herein is subject to change without notice and Acer Inc. shall not be liable for technical

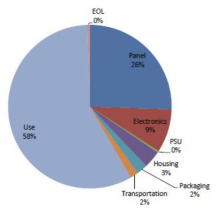
Figure 7: PCF report example for Acer



Lenovo Product Carbon Footprint (PCF) Information Sheet
PC/Notebook/Monitor/Tablet

Commercial Name	C32qc-20/D32qc-20
Model Number	66A5, 66A6
Issue Date	2019-11-15

Product Environmental Attributes	
(a) Product Carbon Footprint Value:	**969 kg of CO_2e (see Note 1 below)**
(b) Product Picture:	**(c) Life Cycle Detail by Component & Life Stage (Pie Chart):**



Note 1:
All estimates of carbon footprint are uncertain. Lenovo reports the 95th percentile of the carbon footprint estimate to reflect that uncertainty. For this product, that estimate has a mean of 569kg of CO_2e and standard deviation of 213kg of CO_2e. For a quantity that follows a normal distribution, the 95th percentile value is equal to the mean plus the standard deviation multiplied by 1.64. Other organizations might report this value as569+/- 213 kg of CO_2e.

This PCF was generated using the Product Attribute to Impact Algorithm model, Version 5/9/2018, Date: 5/9/2018 (Product Type: Monitor), © Massachusetts Institute of Technology's Materials Systems Laboratory, August 2012. Please refer to the Intended Uses and Limitations of the PAIA Model, © Massachusetts Institute of Technology's Materials Systems Laboratory, August 2012 for further details. Link to Document

This calculation was based upon a Lenovo D32qc-20/C32qc-20 with the assumptions and configuration described in the calculation assumptions in the next page.

This pie chart provides the percent contribution of the mean value for each element of the analysis for the full life cycle CO_2e impacts of the product. Individual elements displaying 0% are less than 0.5%.

Lenovo Product Carbon Footprint (PCF) Information Sheet

Assumption Table

Category	Element	Unit	Input	Mean	COV
Product Specifics	Product Weight	kg	7.36		Primary
	Form Factor	no unit	2560*1440		
	Screen Size	inches	31.5		
	Product Lifetime	years	5		
Location	Assembly Location	no unit	CN		
	Use Location	no unit	WW		
Transportation from Assembly to Customer	To country of use: by air	fraction	0.01		
	To country of use: by ship	fraction	0.99		
	To country of use: by rail	fraction	0		
	To country of use: by truck	fraction	0		
	In country of use: by air	fraction	0.2		
	In country of use: by ship	fraction	0		
	In country of use: by rail	fraction	0.3		
	In country of use: by truck	fraction	0.5		
End of Life	Fraction Recycled (remainder to landfill)	fraction	0.83		
	Fraction Shredded Recycling (remainder to manual)	fraction	0.9		

The PCF value is calculated using the specific attributes above for assembly, use and transportation mode. If you need any other country specific information, please contact environment@lenovo.com.

Notes:

Life cycle phases included in the streamlined Product Attribute to Impact Algorithm (PAIA) Life Cycle Analysis (LCA) can be grouped into four categories which include Manufacture, Transport, Use, and End of Life. Below is a brief description of each phase.

Manufacture: This life cycle phase captures emissions generated during the extraction, production, and transport of raw materials, the manufacture of components and subassemblies (including the product packaging) and product assembly.

Transport: Emissions included in the transport phase include all those generated during the air, ocean or land transport of finished or semi-finished Lenovo products between Lenovo facilities and from Lenovo facilities to customers.

Use: In use energy consumption is calculated in accordance with the U.S. Environmental Protection Agency's Energy Star® Typical Energy Consumption (TEC) methodology. Calculated energy consumption is then used in combination with average emissions factors for the designated country of use to calculate emissions.

End of Life: It is assumed that a designated portion of the product (see table above) is recycled at the end of the use period determined in the TEC methodology. It is also assumed that the balance of the product waste materials is disposed of by landfill. Emissions generated during the mechanical destruction, separation and transport of end of life materials are included in the calculation.

Product scope of this sheet includes desktop computer, integrated desktop computer, notebook computer, monitor and tablet.
This document is only valid in connection with "THE ECO DECLARATION" of the specific product.

Page 2 of 2

Figure 8: PCF report example for Lenovo

```
You'll be provided with some questions and a reference. Based on the reference,
    generate the Python program to compute and answer the questions. The program is
    enclosed by triple backticks. The final answer in the program is of list type.
### Question: {question}
### Reference: {reference text}
### Program:
```
{program}
```
```

Listing 1: CarbonPDF training prompt template

```
You'll be provided with some questions and a reference. Based on the reference,
    generate the Python program to compute and answer the questions. The program is
    enclosed by triple backticks. The final answer in the program is of list type.
### Question: {question}
### Reference: {reference text}
### Program:
```

Listing 2: CarbonPDF testing prompt template

```
You'll be provided with some questions and a reference. Based on the reference,
    provide the answer of list type.
Here are some examples.
Example 1:
### Question: What are the carbon footprints of chassis, total, and display in the
    R856T-TCO laptop?
### Reference: Product Carbon Footprint Acer carefully consider environmental
    factors in every stage of the product life cycle ... That estimate has a mean of
     231 kg of CO2e and standard deviation of 46 kg of CO2e . R856T, R856TN, R856LT,
     R856LTN R856T-TCO, R856TN-TCO, R856LT-TCO, R856LTN-TCO Product carbon footprint
     by percentag  e % 47.3% 17.5% 10.0% 8.5% 6.3% 5.8% 3.2% 0.8% 0.6% 0.0% 0.0%
    General Information 1.4 kg 12"" 11.3 kWh 4 years About the Data Disclaimer 2023/
    May ... The LCA result strongly influenced by the assumptions made and PAIA
    tools are not configured to allow for simultaneous simulation, it is not
    recommended that PAIA results be used in comparisons. Product breakout Display
    Mainboard (and other boards) Use Power Supply Unit(s) Transport Chassis
    Typical Energy Consumption (Yearly TEC) Battery Packaging End of Life    Final
    Assembly in China and use in Europe 0.00 0.00 ... Panel Size    Product Weight (
    excluded accessory and packaging)    Product Lifetime Manufacturing  83.1% End
    of Life 0.6% Use 10% Transport 6.3%
### Answer: [13.398, 231.0, 109.263]

Example 2:
### Question: What are the components with the highest and lowest carbon footprint
    percentages in the manufacturing breakdown of the Latitude 5310 2-in-1 laptop?
### Reference: Dell Latitude 5310 2-IN-1 ... This includes the  contributions  from
     materials,  manufacturing, distribution, use and end-of-life management.
 This product, estimated carbon footprint: 299 kgCO2e +/- 58 kgCO2e
 Estimated impact by lifecycle stage with  breakout for manufacturing  by component:
       ...  Product Weight 1.351 kg Screen Size 13.3,  Assembly  Location  China
     Product Lifetime 4 years  Use Location   EU  Energy Demand  (Yearly TEC)  19.43
      kWh Disclaimer: This PCF was calculated using the PAIA model, version 1.2.6,
     2020. Results shown here are subject to change as  the tool is updated.
     Manufacturing 83.1% Chassis & Assembly 4.1% Hard Drive 0.0% SSD 2.7% Power
     Supply 8.8% Battery 1.9% Mainboard and Other Boards 28.0% Display 37.2%
     Packaging 0.4%
### Answer: [{{'display': 37.2}}, {{'packaging': 0.4}}]

Example 3:
### Question: What are the top 5 components with the highest carbon footprint
    percentages in the manufacturing breakdown of the HP ZHAN 66 Pro A G4 All-in-One
     PC desktop?
### Reference: Product Carbon Footprint Report HP ZHAN 66 Pro A G4 All-in-One PC GHG
     Emissions Manufacturing Breakout  Display 26.8% Mainboard and other boards
    25.7% Solid State Drive (SSD) 24.9% Chassis 13.3% Power Supply Unit & External
    Cables 3.3% Others* 3.2% External components (Keyboard & Mouse) 1.8% Packaging
    1.0%   Assumptions  5 North America 72.16 7.3 23.8"" China Learn more at  ... HP
     shall not be liable for technical or editorial errors or omissions contained
    herein. HP shall not be liable for technical or editorial errors or omissions
    contained herein.426kg CO 2eq. kg CO2 Manufacturing54%Distribution M% Use 45%
    End of Life 1% Value chain  carbon footprint  1% Value chain  carbon footprint
### Answer: [{{'display': 26.8}}, {{'mainboard': 25.7}}, {{'ssd': 24.9}}, {{'chassis
    ': 13.3}}, {{'power': 3.3}}]

Example 4:
### Question: What is the carbon footprint of total in the Lenovo L28u-30?
### Reference: Lenovo Product Carbon Footprint (PCF) Information Sheet  PC/Notebook/
    Monitor/Tablet  Commercial Name  Lenovo L28u-30  Model Number  65FA  Issue Date
     2019-08-09  - Revised 8/15/2022 Product Environmental Attributes  (a) Product
    Carbon Footprint Value: 455 kg of CO2e (see Note 1 below)  (b) Product Picture:
    (c) Life Cycle Detail by Component & Life Stage (Pie Chart):  Note 1:   All
    estimates of carbon footprint are uncertain. Lenovo reports the 95th percentile
    of the carbon footprint  estimate to reflect that uncertainty ...
### Answer: [455.0]

Now the questions and reference are shown below. What are the answers to the
    questions?
### Question: {question}
### Reference: {reference text}
### Answer:
```

Listing 3: Few-shot prompt template

```
You'll be provided with some questions and a reference. Based on the reference,
    provide the answer of list type.
### Question: {question}
### Reference: {reference text}
### Answer: {answer}
```

Listing 4: Without program-based reasoning training prompt template

```
You'll be provided with some questions and a reference. Based on the reference,
    provide the answer of list type.
### Question: {question}
### Reference: {reference text}
### Answer:
```

Listing 5: Without program-based reasoning testing prompt template

```
patterns = {
    'ssd': r'Solid State Drive \(SSD\)\s*(\d+(\.\d+)?)%',
    'hdd': r'Hard Drive \(HDD\)\s*(\d+(\.\d+)?)%',
    'batteries': r'Batteries\s*(\d+(\.\d+)?)%',
    'chassis': r'Chassis\s*(\d+(\.\d+)?)%',
    'power supply unit': r'Power Supply Unit & External Cables\s*(\d+(\.\d+)?)%',
    'mainboard': r'Mainboard and other boards\s*(\d+(\.\d+)?)%',
    'display': r'Display\s*(\d+(\.\d+)?)%',
    'packaging': r'Packaging\s*(\d+(\.\d+)?)%',
    'odd': r'(?:Optical Disk Drive \(ODD\)|ODD)\s*(\d+(\.\d+)?)%',
    'external components': r'External components \(Keyboard & Mouse\)\s*(\d+(\.\d+)
        ?)%'
}
```

Listing 6: Example regular expressions for HP data collection

```
program = f"```\ntotal_carbon={total_carbon}"
if len(interests) == 1 and 'total' in interests:
        program += "\nanswer=[total_carbon]"
else:
    program += f"\nmanufacturing_percent={components['manufacturing']}"
    for interest in interests:
        if interest == 'manufacturing':
            program += f"\nmanufacturing_carbon=total_carbon*manufacturing_percent"
        else:
            for name, percent in components.items():
                if interest in name:
                    program += f"\n{name}_percent={percent}"
                    program += f"\n{name}_carbon=total_carbon*manufacturing_percent
                        *{name}_percent"

    program += "\nanswer=["
    for interest in interests:
        if interest == 'total':
            program += 'total_carbon,'
        elif interest == 'manufacturing':
            program += 'manufacturing_carbon,'
        else:
            for name, percent in components.items():
                if interest in name:
                    program += f"{name}_carbon,"
    program = program[:-1]
    program += "]"
program += "\n```"
```

Listing 7: Python program template for HP program generation to answer calculation questions

```
You'll be provided with some questions and a reference. Based on the reference,
    generate the Python program to compute and answer the questions. The program is
    enclosed by triple backticks. The final answer in the program is of list type.
### Question: What are the carbon footprints of manufacturing and display in the HP
    EliteOne 800 G6 24 All in One PC (ENERGY STAR) desktop?
### Reference: Product Carbon Footprint Report 3-Aug-2023 HP EliteOne 800 G6 24 All
    in One PC (ENERGY STAR) GHG Emissions Manufacturing Breakout  Mainboard and
    other boards 32% Display 24% Solid State Drive (SSD) 23% Chassis 10% Others* 4%
    Power Supply Unit & External Cables 4% Packaging 2% External components (
    Keyboard & Mouse) 2%  ... 505 505kg CO 2 eq. eq. Manufacturing 50% Distribution
    0% Use 49% End of Life 1% Value chain  carbon footprint
### Program:
```
total_carbon=505.0
manufacturing_percent=0.5
manufacturing_carbon=total_carbon*manufacturing_percent
display_percent=0.24
display_carbon=total_carbon*manufacturing_percent*display_percent
answer=[manufacturing_carbon,display_carbon]
```
```

Listing 8: Example CarbonPDF prompt