Emission Factor Recommendation for Life Cycle Assessments with Generative AI

Bharathan Balaji,[*,†] Fahimeh Ebrahimi,[†] Nina Gabrielle G Domingo,[‡] Venkata Sai Gargeya Vunnava,[‡] Abu-Zaher Faridee,[¶] Soma Ramalingam,[†] Shikha Gupta,[†] Anran Wang,[†] Harsh Gupta,[§] Domenic Belcastro,[†] Kellen Axten,[†] Jeremie Hakian,[†] Jared Kramer,[†] Aravind Srinivasan,[‖] and Qingshi Tu[⊥]

†*Amazon, Seattle, WA 98121, USA*

‡*Amazon, New York, NY 10018, USA*

¶*Amazon, Arlington, VA 22202, USA*

§*Amazon, East Palo Alto, CA 94303, USA*

‖*University of Maryland and Amazon, College Park, MD 20742, USA*

⊥*The University of British Columbia, Vancouver, British Columbia, V6T 1Z4, Canada*

E-mail: bhabalaj@amazon.com

Abstract

Accurately quantifying greenhouse gas (GHG) emissions is crucial for organizations to measure and mitigate their environmental impact. Life cycle assessment (LCA) estimates the environmental impacts throughout a product's entire lifecycle, from raw material extraction to end-of-life. Measuring the emissions outside of a product owner's control is challenging, and practitioners rely on emission factors (EFs) – estimates of GHG emissions per unit of activity – to model and estimate indirect impacts. However, the current practice of manually selecting appropriate EFs from databases is time-consuming, error-prone, and requires expertise. We present an AI-assisted method

leveraging natural language processing and machine learning to automatically recommend EFs with human-interpretable justifications. Our algorithm can assist experts by providing a ranked list of EFs or operate in a fully-automated manner where the top recommendation is selected as final. Benchmarks across multiple real-world datasets show our method recommends the correct EF with an average precision of 86.9% in the fully-automated case, and shows the correct EF in the top 10 recommendations with an average precision of 93.1%. By streamlining EF selection, our approach enables scalable and accurate quantification of GHG emissions, supporting organizations' sustainability initiatives and progress toward net-zero emissions targets across industries.

Synopsis: We present an AI-assisted method that streamlines emission factor matching, improving its scalability and accuracy, thereby supporting effective environmental policy and sustainability initiatives.

Keywords: emission factor recommendation, life cycle assessment, carbon footprint, large language model, retrieval augmented generation, environmental impact assessment

Introduction

Organizations worldwide have pledged to achieve net-zero carbon emissions and publicly release their annual greenhouse gas (GHG) emissions as part of their sustainability reports.[1–3] The Greenhouse Gas Protocol[4] is the widely adopted standard used in estimating GHG emissions, and it is based on the principles of Life Cycle Assessment (LCA).[5,6] The GHG Protocol requires declaration of both direct and indirect emissions. Direct emissions (Scope 1) include emissions from owned sources such as fuel combustion by a vehicle. Indirect emissions are further categorized into two types: Scope 2 emissions are associated with the purchase of electricity, steam, heat, or cooling for operations, while Scope 3 emissions encompass all other indirect emissions along the value chain such as those from raw material extraction, manufacturing processes, transportation, product use, and end-of-life.

Accurately measuring the numerous sources of indirect emissions across the supply chain can be prohibitively expensive and sometimes infeasible.[7,8] Consequently, practitioners rely on databases of *emission factors* (EFs) that provide GHG emissions associated with core materials, products, and industries on a per-unit basis, e.g., kg CO2-eq per kg of cotton produced or kg CO2-eq per kWh of electricity.[9–11] Practitioners acquire these EF datasets, map each business activity to an appropriate EF, and estimate their footprint by scaling the EF. The total carbon footprint is the sum of these individual emission estimates. However, the manual selection of EFs from databases containing hundreds or thousands of entries is a slow and expensive process.[9,12] Our motivation for this work stems from observing LCA scientists, e.g. co-author Gargeya Vunnava, spending weeks of their time on EF matching. To address these challenges, we aim to develop automated EF recommendation algorithms.

LCA practitioners rely on background datasets such as Ecoinvent,[10] Sphera[13], and USEEIO[11] to characterize the emissions associated with upstream (raw material extraction, transportation to factory) and downstream (transportation to wholesalers, retailers, and consumers, and disposal of the product) processes. The exact emission value depends on the allocation[14] and impact assessment methods[15] used. An EF is a combination of the background dataset, allocation, and impact assessment method. Practitioners of LCA either pre-calculate their EFs using a fixed allocation and impact assessment method, and then select each EF manually, or they manually select a background dataset, allocation, and impact assessment methods to calculate a specific EF on the fly. We focus on the former approach, with global warming potential defined by the IPCC sixth assessment report.[16] However, our methods can be generalized to background dataset recommendation.

There are two primary approaches to LCA: process-based and Environmentally Extended Input-Output analysis-based (EEIO).[17] A process-based LCA (pLCA) is a bottom-up approach that tracks all the inputs (i.e., material and energy) and outputs (i.e., emissions and environmental waste) of a product across its supply chain. This framework allows practitioners to investigate the source-specific impacts of a product and identify hotspots in the supply

chain. For example, the Ecoinvent database provides a pLCA-based EF for the activity 'pear production' as 0.31 CO_2e per kg of pear (Ecoinvent v3.9.1, cut-off system, IPCC AR6 impact assessment, location: China).[10] These emissions arise from all the activities in the supply chain starting from growing the pears, farm cultivation and maintenance, fertilizer/pesticide application, water needs, harvesting, and distribution to the final markets. Since individual activities can emit different types of GHG gases (e.g., Methane, Nitrous oxide, etc.), all such GHG emissions are converted to a single unit of mass of carbon dioxide equivalent (or CO_2e) as defined by the IPCC[16] and are available as EFs in datasets.

In contrast, EEIO-LCAs take a top-down macroeconomic approach using supply-use tables to estimate the emissions associated with the production and trade of a given good or service at an industry sector level. Sector-level environmental data such as water withdrawals, greenhouse gases, and energy extraction are collected and normalized to a unit currency based on the gross economic output of each sector.[11] For instance, the US Environmental Protection Agency's EEIO model provides an EF of 0.5 kg CO_2e per US dollar for the 'Fruit and Tree Nut Farming' sector (USEEIO v2.0.1, based on 2017 IO data). A $2 pear will be mapped to this sector, and estimated to have an impact of 1 kg CO_2e.

Exact string matching, commonly used in LCA tools for EF recommendation, fails to capture semantic information in the text such as synonyms (e.g., milk and dairy, maize and corn), or abbreviations (e.g., Ni-Cd and Nickel Cadmium). Prior works have proposed use of machine learning (ML) solutions to overcome these problems.[9,18] A key challenge is the lack of labeled datasets in LCA, making it difficult to train supervised text classification models that generalize to all types of products. To overcome this issue, recent works,[19] including our own such as CaML[20] and Flamingo,[12] have leveraged off-the-shelf semantic text matching models trained on web-scale data without any fine-tuning (zero-shot). CaML[20] focuses specifically on EEIO-LCA, using semantic similarity matching to map products to NAICS industry codes for carbon footprinting. Flamingo,[12] on the other hand, addresses pLCA, introducing an intermediate classification layer to improve 'no match' precision with

semantic text matching for environmental impact factors. These neural network models take text as input, and output a vector representation, called an embedding, such that semantically similar texts are placed closer in the vector space.[21] The similarity between two texts can then be measured by the distance between their embeddings.

Embedding models depend on clear text descriptions to create accurate vector representations. Such high-quality text is available from data sources used in prior works, such as e-commerce product pages.[20] However, we find that organizations often do not have such high-fidelity data in practice. Their information comes from sources like enterprise resource planning systems, which use abbreviations, domain specific terms, and which may contain multiple languages. E.g., 'FAC.WRC.OAL0508IN9.GRU - Combina o chave' describes a combination wrench from our procurement dataset. Asking users to create clean text descriptions manually is burdensome, and directly feeding the inputs to the embedding model returns incorrect results. Another challenge with embedding models is that they are trained on generic web-based definition of semantically similar data that may not align with the definitions associated with GHG emissions. E.g., for the query 'macbook', the fruit 'apple' shows up with a high similarity score. Our study aims to overcome these shortcomings with the use of large language models (LLMs).

LLMs have emerged as powerful tools for natural language processing, capable of generating human-like text across various domains. These models, such as GPT[22] and Claude[23] are trained on vast amounts of textual data with a self-supervised auto-regression objective, enabling them to capture intricate patterns and generate contextually relevant text. These models are further optimized through instruction tuning on question-answer pairs, improving their ability to respond to prompts in a coherent manner and provide responses favored by humans. However, LLMs often exhibit limitations, including hallucinations, inconsistencies, and lack of grounding in factual knowledge.[24]

Retrieval Augmented Generation (RAG) aims to mitigate these limitations by combining the generative capabilities of LLMs with the ability to retrieve and incorporate relevant

information from external knowledge sources.[25] RAG systems typically consist of a retriever module that identifies relevant passages from a knowledge base with an embedding model, and a generator module (an LLM) that uses the retrieved information to generate the final output. This approach has shown promise in improving the factual consistency and grounding of generated text, particularly in open-ended question answering and dialogue tasks.[26] Prompts, or the input text provided to LLMs, play a crucial role in their performance. As LLMs are primarily trained on self-supervised learning objectives, their outputs can be sensitive to the prompts used.[27] Prompts serve as the initial context for the model, guiding its generation process. Effective prompts can elicit desired behaviors from LLMs, enabling more accurate and relevant generation across a wide range of tasks.[28]

The key distinction between LLMs and embedding models in handling domain-specific text lies in their training objectives and operational capabilities. While embedding models are trained to create fixed vector representations that capture semantic similarity between texts,[21] LLMs are trained to predict and generate text sequences.[22] This fundamental difference in purpose leads to several important advantages for LLMs when handling unclear text descriptions. When embedding models encounter cryptic text like 'FAC.WRC.OAL0508IN9.GRU', they attempt to map it directly to a vector space based on their training data. However, since such abbreviated or technical notation rarely appears in their training corpora (which typically consists of natural language text[29]), they struggle to create meaningful representations. The embedding models cannot transform or interpret the text - they can only represent it as-is in the vector space.

In contrast, LLMs' generative capabilities allow them to actively process and transform the input text. Through their training on next-token prediction across vast amounts of technical documentation,[30] they learn to decode abbreviated patterns and expand them into complete descriptions. This is not just pattern matching, but rather a learned ability to generate contextually appropriate expansions of technical shorthand. For example, when encountering 'FAC.WRC', the LLM can generate a complete description like 'Factory Wrench'

based on its understanding of common industrial terminology and abbreviation patterns. The instruction-tuning phase further enhances this capability by teaching LLMs to perform implicit tasks like text standardization. While embedding models remain static in their representation approach, LLMs can dynamically adapt their output based on the implied need for clarification or expansion. This ability to actively transform and explain cryptic text, rather than just represent it, is what enables LLMs to succeed where embedding models struggle.

Use of RAG ensures we can provide domain specific context to the LLM as it may not be part of its training data, which cannot be done with embedding models. For example, we include information such as process description and system boundary in our algorithm. RAG also helps reduce the context information sent by selecting a subset of EFs to be considered with semantic matching, which assists in both improved performance and reduced costs in using LLMs.

Our scope of research is orthogonal to prior works that have used ML for addressing data gaps in life cycle inventory and characterization factors.

Recent studies have revealed that inventory data gaps remain a fundamental challenge in LCA and illustrated how ML approaches can contribute to bridging these gaps. For example, Write et al.[31] and Zargar et al.[32] highlighted that methods based on supervised learning algorithms, such as artificial neural networks, extreme gradient boosting, and similarity-based models, can effectively estimate missing inventory flows when sufficient training data is available. However, these methods face limitations including challenges in uncertainty assessment and dependency on high-quality training datasets. Wright et al.[31] also suggested that emerging technologies like Internet of Things (IoT) and blockchain platforms could enhance data collection and validation, while Zargar et al.[32] emphasized the need for diversifying data sources and exploring novel algorithms such as graph neural networks. Addressing these limitations could significantly improve LCI data completeness and reliability, ultimately enabling more accurate environmental impact assessments of products and processes.

Recent studies have also highlighted ML as a promising approach to address gaps in characterization factors. Li et al.[33] demonstrated that XGBoost algorithm can predict average ecotoxicity values (logEC50) for chemicals lacking empirical data, revealing that approximately 13.6% of ecotoxicity impacts from coal power generation in China were previously underestimated due to missing characterization factors. Similarly, von Borries et al.[34] identified 13 priority parameters for ML model development in toxicity characterization, showing ML approaches could potentially predict values for 8-46% of marketed chemicals based on limited available measured data (1-10%). These findings align with Romeiko et al.[35], who found ML applications in LCA improve prediction accuracy, pattern discovery, and computational efficiency across inventory development, impact assessment, and interpretation stages. Together, these studies suggest ML offers a viable pathway to enhance the comprehensiveness and accuracy of LCA through systematic prediction of missing characterization factors, though challenges remain in data availability, model selection, and uncertainty quantification.

To our knowledge, we are the first to leverage LLMs for EF recommendation and validate their effectiveness in enhancing EF prediction accuracy. In our work, we leverage LLMs and RAG, and carefully design prompts to improve emission factor selection by providing relevant context, retrieving information from EF datasets using an embedding model, and guiding the LLM to align with the domain knowledge of life cycle assessment.

Datasets and Methods[a]

Our objective is to match a query text with an appropriate EF from a given dataset. The query text typically describes a product or service that needs an EF assignment. We use EF datasets with clear text descriptions for this matching process. A recommended EF is considered 'appropriate' if a human would agree with its assignment to the query. For

[a]The methods discussed are for research purposes only, and are not indicative of Amazon's business use cases for carbon footprinting.

each query, the algorithm must return exactly one matching EF. In cases where multiple appropriate EFs exist, selecting any one of them is acceptable. For EEIO-LCA datasets, the algorithm always finds an appropriate EF since these datasets comprehensively cover all economic sectors. However, for pLCA datasets, the algorithm should return 'no match' when no suitable EF exists for the query.

pLCA is useful for a granular analysis of emissions associated with an activity, whereas EEIO gives a first-order approximation of the estimated emissions using industry sector level data. The EF recommendation strategy differs for these two types of LCA. For pLCA, each EF requires detailed measurements for a given activity published in the literature or released by the manufacturer, and some products or activities may not have available EFs. For example, the Ecoinvent dataset has an EF for 'pear' and 'orange', but does not have an EF for 'plum' or 'grapefruit'.[10] In such cases, an EF recommendation algorithm needs to return that an appropriate choice is not available for the given input. EEIO-LCA EFs, on the other hand, are designed to have broad coverage of products or services, encompassing all industries in an economy, including catch-all 'miscellaneous' sectors for any missing industries.[11] Therefore, the EF recommendation algorithm for EEIO-LCA needs to provide an appropriate EF as long as the given input data is valid.

Our algorithm, named Parakeet[b], uses a combination of generative LLMs and embedding models in a retrieval augmented generation pattern to identify an appropriate EF, when available. We use the LLM to integrate domain-specific instructions via prompts, and ask it to create a plain-language description of the input query. We feed the paraphrased plain description to the embedding model, and find the top-10 semantically similar EFs from the dataset by sorting them on similarity score. We feed the top-10 EFs back to the LLM, and ask it to recommend an EF to use given the previous context. We collate the query, LLM

[b] We use the name "Parakeet" from a species of small, intelligent parrots known for their ability to learn and repeat human speech. Like the bird, our algorithm learns to mimic human abilities to match business activities into appropriate emission factors. Parakeet also explains itself in natural language, similar to a human. Additionally, parakeet's green color symbolically aligns with green innovation in carbon footprint assessment.

outputs, and top-10 EFs for human review and verification. Our algorithm can directly ingest real-world business data lacking clear descriptions assumed by prior approaches. It provides human-readable justifications to enable expert verification.

Parakeet differs in its details for EEIO and process LCA due to the structure of EFs in the respective datasets. Figure 1 illustrates the EEIO recommendation process through a detailed example. We use the General Text Embedding (`gte-large`) models for semantic text matching,[36] Claude 3 Sonnet as our LLM,[23] and include one example in each prompt. These parameters and models yield the best performance among the ones we evaluated. The algorithm is designed to be modular, and it is easy to switch between different models.

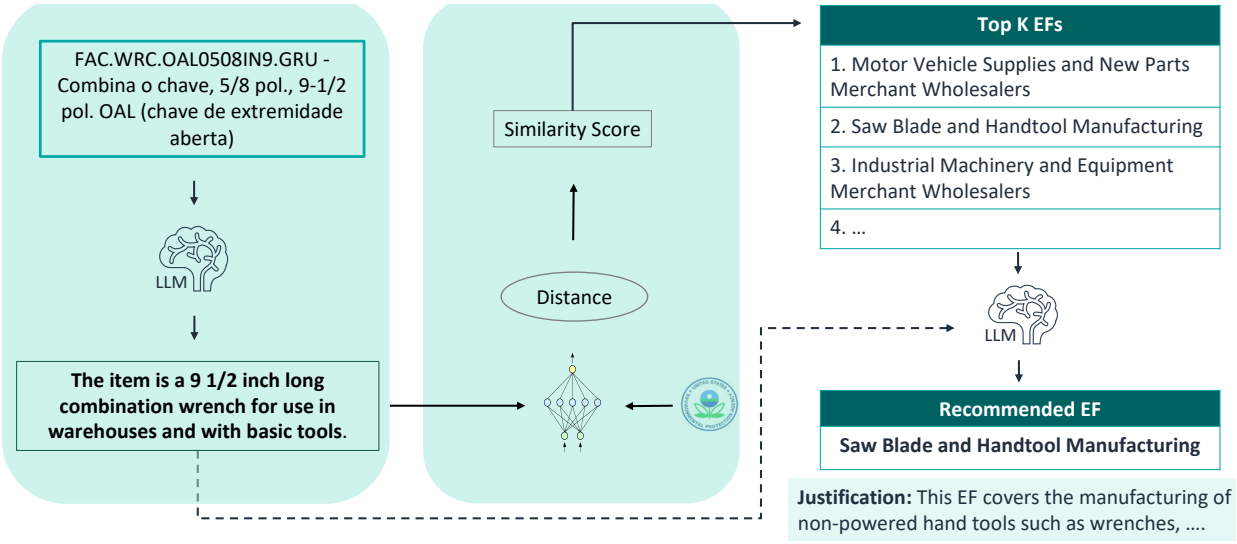


Figure 1: An example of Parakeet's EEIO EF recommendation process. First, Parakeet paraphrases the given activity description into plain text. Next, using an embedding model, it retrieves the most semantically similar EFs to the given activity. Finally, these similar EFs are fed into the LLM model, which recommends the best matched EF and generates a justification for this recommendation.

The first step of the algorithm is to produce a plain-language description of the query using an LLM. The given query can consist of multiple fields such as name, description, company, and category. Feeding this input directly into an embedding model yields imprecise retrieval of EFs as these models are trained on full sentences available on the web rather than invoices or manufacturing data. In our prompt, we ask the LLM to be an LCA expert. We include specific instructions such as inclusion of information relevant to material and

manufacturing phase of LCA, and expansion of relevant abbreviations. All of our prompts include an example for the LLM to follow, referred to as a "1-shot" prompt in machine learning literature[22], where the shots refer to number of examples. In the Numbers of Examples in Prompts section, we further examined the impact of the number of examples in prompts on the performance of the model. We also include all the prompts used in our study in Supplemental Information (SI).

The second step of the algorithm is to feed the plain language text to an embedding model to create a vector representation of the text data. Embedding models are neural network based machine learning models trained on web-scale data using a self-supervised objective, where no human provided labels are used during training. Examples include predicting the next word in a sentence,[37] or comparing similar and dissimilar sentences.[38] The large scale training data enables the model to generalize across domains. These models are modified to output a numeric vector representation of a given input text, and are further fine-tuned on explicitly labeled similar/dissimilar sentences to improve their text matching performance.[21] We use the general text embedding (gte) model[36] that is trained on ~800M sentences in the self-supervised stage, and fine-tuned with ~3M labeled sentences. Cosine similarity is used as the measure of distance between two embeddings. We use the `gte-large` variant, which occupies 1.6GB of memory, and outputs 1024-dimensional embedding for a given sentence. Similar to the LLM models, we picked the state-of-the-art embedding models available to us at the time of experimentation. The cost of embedding models is outweighed by both the human time as well as any of the LLM models we use in Parakeet. We gauged the relative capability of these models from the Massive Multilingual Text Embedding Benchmark.[39,40] In general, Parakeet can be used with any of these models, and future improved version of embedding models can be incorporated easily. In the Choice of Embedding Model section, we examined the performance of our model using five other embedding models.

We create embedding representations for all EFs in the dataset and compute their cosine similarity with the query embedding. The top-10 EFs are then ranked in ascending order

based on their distance scores. In the final step, we feed the top-10 EFs to the LLM and ask it to recommend the best matching EF along with a justification. The prompt includes the history of the paraphrasing step, and additional context explaining that the ranked list of EFs is obtained with semantic text similarity. We used the state-of-the-art LLM models available to us in our experiments, as the cost of using the LLM models outweighs the human time it takes to pick an EF without assistance. These LLM models include the series of models from Claude,[41] Llama,[42] and Mistral.[43] We have excluded models which failed to give us well-formed responses, e.g., when the data format was not readable with our code. We gauged the relative capability of these models with open benchmark leaderboards such as Chatbot Arena.[29] We also evaluate smaller models such as Claude 3 Haiku to assess the impact on performance and cost of recommendation. In general, Parakeet can be used with any of these models, and future versions of LLM models can be incorporated easily. We used Claude 3 Sonnet as our default LLM and evaluated how different LLMs affect the performance of Parakeet. The Choice of LLM section presents these comparative results. We also evaluated the impact of each component through ablation studies by separately removing LLM paraphrasing, semantic matching (embedding models), and LLM EF recommendations to test their contribution to the model's overall performance. The Ablation Analysis section presents the results of these experiments.

To collect a ground truth EF for each query, we compile the given query, paraphrased query, recommended EF, and the top-10 EFs, and send it to a human annotator for review. The human can choose to select one of the top-10 EFs, mark that the given query is unclear, that the list of EFs is irrelevant, or that they are 'not sure'. Figure 2 shows an example. We provide annotators of Parakeet with guidance to help them understand the steps to annotate and the science behind it. The annotation process involves selecting the most appropriate EF from a list of options that best matches a given input activity description. Annotators are instructed to carefully read and understand the input text, consider AI-generated interpretations and recommendations, and make selections based on their understanding. The

instructions emphasize the importance of identifying cases where none of the recommended EFs are a match or when the input text is unclear. In order to create a robust ground truth for each of our datasets, we asked for triple votes from the annotation team and applied majority voting to establish ground truth. Collecting ground truth this way introduces bias, as the annotator is only exposed to the EFs short-listed by the model. For example, it is possible annotators pick a different answer if the short list had more choices. While we minimize this bias with clear instructions and by including options like 'none of the options match', ideally we would have asked the annotators to pick the EF based on the full list of thousands of choices. However, it is challenging and time consuming to collect sufficient data to perform such an analysis — the exact problem we are trying to solve in this work.

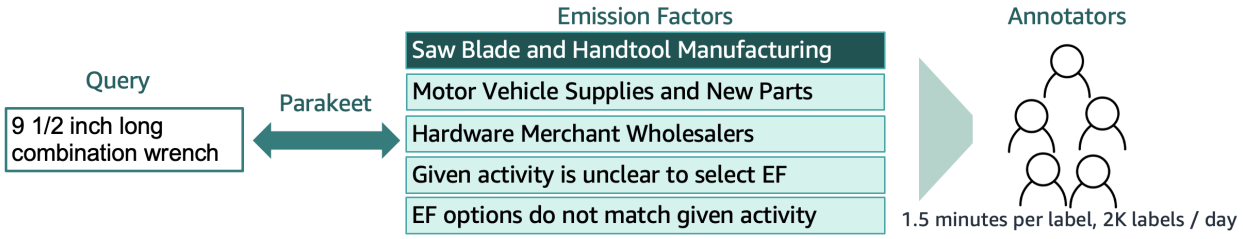


Figure 2: We provide the query, the paraphrased text, the recommended emission factor, and top-ranked list of emission factors to an annotator. The annotator can choose to override the Parakeet recommendation, indicate that the input data or provided EFs are inappropriate, or express uncertainty about making a choice.

We have only experimented with cradle-to-gate emission factors (EFs) in this research. All of the EFs we use in both USEEIO and Ecoinvent datasets are cradle-to-gate EFs, covering various reference units such as kg CO2-eq per kg of material produced, kg CO2-eq per kWh of electricity consumed, or kg CO2-eq per ton-km of transportation. It's important to note that Parakeet does not currently piece together EFs as practitioners often do in real-world applications, such as combining raw material production with manufacturing processes. In process-based EF, for a given query (like 'carrot juice'), we require the EF to cover the entire cradle-to-gate system boundary. If available data only covers partial processes (e.g., if the best match is 'carrot production'), then we mark the ground truth as 'no match'. We evaluate the performance of Parakeet with four metrics:

Precision@K: Given a list of queries with a matching item, Precision@K measures the

fraction where the correct item is ranked within the top-K results.[12,44] The value of K measures the performance depending on how the algorithm is used. Top-1 refers to the fully automated solution, if the metric is 95%, there is a 5% chance that the chosen EF is incorrect. For Top-10, we expect a human to inspect the recommended EFs and select one of them as the EF to use for LCA. In this case, Precision@10 of 95% means that the human will be able to find a match among the top-10 recommendations 95% of the time. We present a maximum of 10 choices to the user. We picked 10 based on search engine user experience research, which has shown that more choices increase mental load and frustration.[45] We also instruct the LLM to rank the choices with the most relevant to the least, so that the user spends less time going through the short-list.

Mean Absolute Percentage Error (MAPE): Rather than text matching of emission factors, MAPE measures the prediction accuracy from a carbon footprint perspective by comparing the $kgCO_2e$ values of the predicted and correct EFs. For each input, we calculate the error between the CO_2 impact values associated with the predicted EF and the ground truth EF. For instance, if the correct EF for 'lentil' has an impact value of 0.77 $kgCO_2e$ per kg ('market for lentil'), and the model predicts an EF with an impact value of 0.71 $kgCO_2e$ per kg ('lentil production'), the absolute percentage error for this prediction would be 7.8%. The below formula shows how MAPE is calculated for n data points.

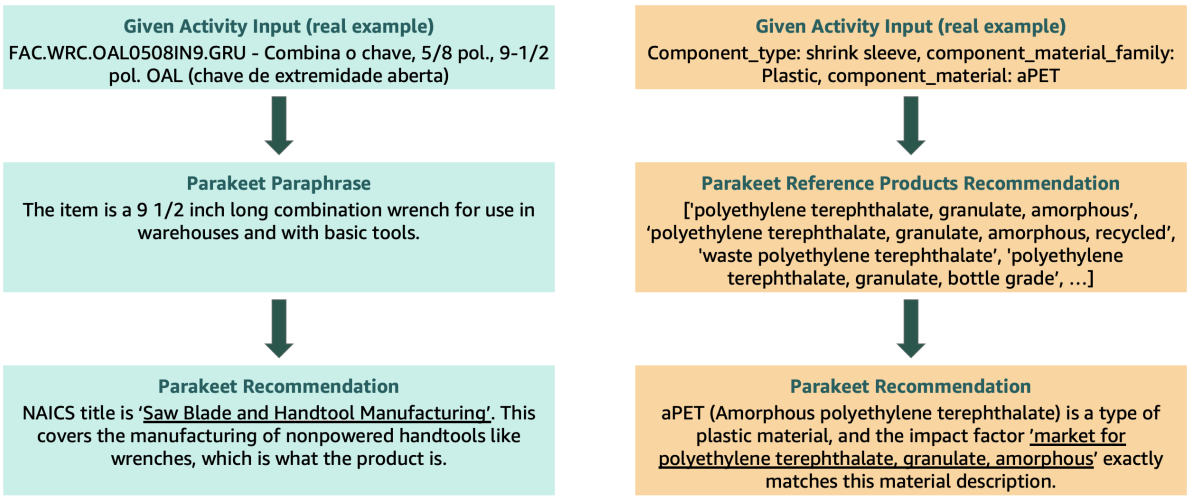
$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{\text{Actual}_i - \text{Predicted}_i}{\text{Actual}_i} \right|$$

Latency: It indicates the time it takes to return the EF recommendation given the input. It is an important measure of both cost and responsiveness of the algorithm. Smaller LLMs have both lower latency and cost but also have lower performance.[23] All of the experiments were conducted on a MacBook Pro with an Apple M3 Pro chip and 16GB memory.

Match Precision: In the case of pLCA, where there might be data points that do not have an exact matched EF, both annotators and the model label them as 'no match'. Match Precision calculates the precision on only data points that have a valid EF (not 'no match')

in the ground truth.

Figure 3 shows an overview of the variations in the steps of the algorithm for EEIO LCA and pLCA. We explain the details of the variations in the algorithm along with the EF datasets used in Supplementary Information (SI).



(a) EEIO LCA **(b)** Process LCA

Figure 3: Two examples of Parakeet mapping product descriptions to EFs. Figure (a) shows Parakeet suggesting the product is a 'combination wrench' from the given activity description and using that it can recommend the most appropriate NAICS title (EEIO LCA). Figure (b) shows an analogous example for process LCA where Parakeet finds the closest reference product and associated impact factor for a product it paraphrases as 'shrink sleeve' from the given activity description.

Business Activity Datasets

We evaluate Parakeet on four EEIO-LCA and two pLCA datasets.

EEIO - Proprietary Procurement Products: 3.9K unique products from a proprietary Procurement catalog. This dataset covers a diverse range of items commonly used in offices and warehouses, including knee braces, locks, metal detectors, and printer toner: e.g., 'AM.BRC.KNEE.XL - Brace, Knee, Neoprene, Open Patella, Extra Large', in the category 'Medical & First Aid Supplies'.

EEIO - Austin Invoices: Sample of 2.1K invoices from City of Austin, Texas, USA government records.[46] City level expenditures such as steel, printers, irrigation systems: e.g., 'Gloves Work Mechanic Synthetic Leather SZ Large'.

EEIO - KatanaML Invoices: Dataset of 1.1K unique invoices collected for machine learning research in data extraction, includes a variety of items such as rugs, game console, and furniture:[47] e.g., 'October Mist Deer Metal Side Table by Rosemary Millette'.

EEIO - Proprietary Heavy Equipment Data: A dataset of 3.7K products that include a variety of items, such as Air Handler Unit, and Exhaust Fans, categorized by materials, labor, and freight. For example, 'Camboard Rating Plug Change and Bus Bar Modification' is under the category 'Generator' and the subcategory 'Labor'.

pLCA - Food.com Ingredients: We extract a random sample of 2K ingredient entries from a dataset of recipes.[48] The dataset consists of a variety of ingredients such as chocolates, carbonated drinks, herbs, and steak.

pLCA - Proprietary Grocery Packaging: 5.5K (195 unique) data points on packaging used in grocery products, including cardboard boxes, paper, various types of plastic: e.g., 'HDPE High density polyethylene' used in primary packaging of plastic bags.

Results and Discussion

We evaluate the overall performance of Parakeet, for both EEIO and pLCA, with the above six datasets. We use the Katana ML invoices and Food.com ingredients dataset to further characterize each step used in the algorithm. These are public datasets, and we make our annotated groundtruth datasets available as part of our open-source repository for reproducibility and further research in this area.

Table 1 summarizes evaluation across our six datasets. For these experiments, we used Claude 3 Sonnet LLM and `thenlper/gte-large` embedding model. Parakeet shows strong performance in accurately recommending EFs across diverse real-world business activity data. Precision@1 indicates that the annotator agrees with the top-recommended EF being an appropriate choice. Precision@10 indicates that the annotator picked one among the top-10 EFs recommended by Parakeet. For pLCA cases, the shortlist provided to annotators

Table 1: Performance of Parakeet for EEIO-LCA and pLCA EF recommendation across our six datasets

Dataset	Precision@1	Precision@10	Dataset Size
EEIO LCA EF Recommendation			
Govt of Austin Invoices	93.5	98.8	2159
Katana ML Invoices	97.1	100	1121
Procurement Products	90.9	98.1	3980
Heavy Equipment	82.2	93.5	1803
Process LCA EF Recommendation			
Food.com Ingredients	71.0	72.9	1956
Grocery Packaging	82.2	89.2	195

may contain fewer than 10 EFs, as it depends on the number of EFs associated with the matching reference products. However, we maintain a maximum of 10 EFs in the shortlist. For EEIO datasets, we observed that limiting recommendations to the top-5 EFs maintained the same precision, as the correct EF was typically found within the top-5 ranked suggestions. The higher Precision@10 scores indicate that annotators almost always pick an option among the short-listed EFs. These strong results showcase Parakeet's ability to handle the technical terminology and abbreviations found in business data, underscoring the robust and generalizable nature of Parakeet's EF selection capabilities.

Regarding computational requirements, our primary performance metric was latency - the time required to generate an EF recommendation. Using a MacBook Pro with an Apple M3 Pro chip and 16GB memory, we observed that latency primarily depends on the LLM response time and input length. Average response times were 11.2 seconds for EEIO and 17.1 seconds for pLCA data points. We analyzed token usage and associated costs using Claude 3 Sonnet. Each EEIO EF recommendation required an average of 2.4K input tokens and 340 output tokens, costing approximately $0.012 per data point. For pLCA EF recommendations, the model processed an average of 11.2K input tokens and generated 520 output tokens, costing $0.041 per data point.

Error Analysis

EEIO-LCA Analysis

We perform the error analysis on the Katana ML Invoices dataset, where we get a Precision@1 of 97.1% and Precision@10 of 100%. These results demonstrate Parakeet's strong performance in accurately recommending EFs across diverse real-world business activity data. For the 2.9% of data points (33 out of 1,121) where the model predicted the wrong EF, our analysis identified two main types of errors. First, the model incorrectly selected the EF for items related to software and console games like PlayStation or Nintendo, focusing on the software and video manufacturing aspect rather than the correct EF of 'Doll, Toy, and Game Manufacturing'. For example, for the item 'Microsoft Xbox Series X Console Pre-Order', the model incorrectly selected 'Software and Other Prerecorded Compact Disc, Tape, and Record Reproducing'. Second, the model made wrong EF selections for book titles, choosing 'Research and Development in the Social Sciences and Humanities' for the book 'The Neuroscience of Emotion A New Synthesis by Ralph Adolphs' when the correct EF should have been 'Book Publishers'. The model lacks sufficient context about book titles and maps these items based solely on their descriptions, leading to inaccurate EF predictions. These errors indicate that domain specific instructions to the LLM may be necessary in order to improve performance further.

pLCA Analysis

We analyzed 567 (out of 1,956) data points where the top match predicted by the model was incorrect for the Food.com ingredients dataset, resulting in a Precision@1 of 71.0% for pLCA EF recommendation. Recall that for pLCA, the EF needs to be an exact match with the given input in order to be correct. For example, if 'carrot production' is the closest matching EF for 'carrot juice', then the output should be 'no match' as the emissions associated with juicing of carrot are not included. We find that this definition plays a

crucial role in the design and performance of the algorithm. In ∼93% of these errors, the model incorrectly recommended an EF while the ground truth was 'no match'. For example, the model selected 'market for grape' for 'wine', even though the correct answer was 'no match'. We instructed the LLM to select an EF that captures all processes performed on an item, but in some cases, it failed to follow this guidance. This mismatch could lead to underestimations or overestimations of environmental impacts, as omitting relevant processes might miss important contributions, while including unrelated processes could add erroneous impacts. For the LLM-based method, these results underscore the need to incorporate LCA-specific decision rules that align with the reasoning of LCA practitioners. Such rules, though not systematically documented in current literature, would improve alignment with real-world LCA practices by guiding the model to select EFs that account for all essential processes in a product's lifecycle.

For the 354 data points in the Food.com dataset that have a valid EF in the ground truth, the Match Precision of the model is 98.1%, indicating that in 98.1% of cases the predicted EF was the same as the ground truth. Examples of incorrect predictions include 'melon production' instead of 'market for melon' for the input 'winter melon', and 'tap water production, underground water without treatment' instead of 'market for tap water' for the query 'tap water'. In Ecoinvent, EF terms like 'market for X' indicate the inclusion of transportation emissions for distribution, while 'production for X' does not. We instructed the LLM to prioritize 'market for X' when available, but for ∼2% of data points, the LLM failed to follow this guidance. These mismatches could lead to underestimations of environmental impacts, as transportation emissions within supply chains may be omitted, potentially under-representing impacts for certain industrial or business sectors.[49] This also highlights the importance of prompt engineering that incorporates LCA-specific terminology and design principles, such as prioritizing terms that reflect the full product lifecycle. This approach would better align model predictions with LCA methodologies, reducing the risk of omitting relevant supply chain emissions.

The remaining ∼5% of errors were due to human misjudgments. Our instructions state that the EF must exactly match the product provided, and if the EF does not cover all emission-causing activities in making the product, then it is considered a 'no match'. For example, in the case of 'lime peel', the model returned 'no match' because existing EFs like 'lime' do not account for the peeling process, while the human selected 'market for lime'. In another case, the model correctly identified 'no match' as the appropriate EF for 'ground lamb', as it refers to the process of grinding lamb meat, which is not covered by any of the provided impact factors, while the human selected 'market for sheep for slaughtering, live weight'.

We have included additional results such as the impact of the choice of LLM model, the embedding model, and extension of the algorithm to recommend region-specific EFs in SI.

Discussion

We have used Parakeet to recommend EFs for carbon footprinting across six domains and 11.2k data points. Users of Parakeet find it convenient as we can process thousands of data-points at a time with batch inference, reducing both time and costs associated with carbon footprints. Parakeet assigns 8K EFs per day automatically, and 2K EFs per day with review by six humans in the loop. Our automated EF recommendation algorithm can be both used as a fully-automated service as well as with a human in the loop. Our algorithm generalizes to both process-based LCA and EEIO LCA emission factor datasets. The algorithm outputs human-interpretable justifications to facilitate third-party verification.

To understand the challenges in finding EFs, we requested co-author Abu-Zaher Faridee to perform EEIO EF mappings without use of Parakeet and document his effort. He was new to the project, LCA and EFs, and we used it as an opportunity to measure the time it takes for a non-expert to perform EF mapping. He noted that the EF mapping process was slow and often frustrating, sometimes spending up to 15 minutes on a single item, like 'a bag of ice melts'. For this item, the lack of clear information in the name and description made it

difficult to choose the right NAICS code. Initial searches on NAICS.com did not yield useful results, so he turned to search engines and expanded to broader product categories, such as "snow removal equipment", to gather enough context. Despite these efforts, determining the correct code remained challenging, highlighting the ambiguities in the process. He completed mapping for 55 items over about 4 active hours spread across two weeks, needing breaks to avoid fatigue and maintain accuracy. A significant learning curve was evident, as the first 25 items took considerably longer than the last 30, and a 40% disagreement rate emerged when the annotator rechecked some of their earlier work. Consulting with LCA scientists, co-authors Gargeya Vunnava and Nina Domingo, helped refine their approach and identify areas for improvement, though the process continued to be demanding, especially for complex or unfamiliar products. We have designed Parakeet to address these challenges.

Parakeet is especially good at recommending EEIO-LCA EFs, with an average Precision@1 of 90.54%, where the task is akin to identifying an industry category for the given input. While we have demonstrated state-of-the-art performance, we still recommend using Parakeet with a human-in-the-loop, especially for the data points used for critical decisions, or those that represent significant environmental impact. We acknowledge that automated EF matching remains challenging, especially for process-based LCA where the algorithm needs to assess system boundary overlap and comprehend domain-specific terminology. In pLCA, we require an 'exact match' that covers all emission-causing activities involved with the given input. Although we explicitly defined these criteria in the prompt, LLMs frequently selected EFs that incompletely covered the emissions associated with given activities rather than recommending 'no match' when appropriate.

The quality and reliability of human annotations were ensured through a rigorous process. First, annotators underwent training sessions and were provided with detailed annotation instructions and resources. Second, we implemented a triple-vote system where each data point was independently annotated by three different annotators, with the final decision determined by majority voting. This approach resulted in high inter-annotator agreement

rates of 94.54% and 97.12% for Food.com Ingredients and Katana ML Invoices datasets, respectively. For process-based datasets, we conducted additional manual inspections to verify the accuracy of the selected EFs. Despite these quality control measures, we observed discrepancies in decision-making among human annotators when choosing the most appropriate EFs. These discrepancies exist across multiple industrial categories, which stems from the ambiguity of individual annotator's interpretation of the information from the EF database. This is a reflection of the current practice in LCA modeling which relies heavily on individual's interpretation of the subject matter, which is affected by the varying degree of domain expertise. Future research can explore establishing a boundary of "acceptable ambiguities" for LCA by surveying the LCA researchers and practitioners. Given the acceptance criteria, it becomes easier to design algorithms that improve in performance.

From a practitioner perspective, Parakeet assists in estimating Scope 3 GHG emissions of a company. Practitioners can assign EEIO EFs to all the financial transactions to estimate the footprint for their portfolio, identify the hotspots, and then perform pLCA for the products they would like to reduce emissions for. Automated EF matching algorithms such as Parakeet significantly streamline the process, reducing the time and expertise needed. The reduction in required resources is especially crucial for small and midsize enterprises (SMEs) to produce LCA results for their products and services. The LCA results from these SMEs, many of whom are suppliers to large industrial and business activities, will help fill the current gap in the Scope 3 emission reporting, where suppliers' LCA information is scarce for most sectors.

Overall, our experiments have demonstrated the benefits of Parakeet to enhance EF recommendations. We have demonstrated that our algorithm outperforms the state-of-the-art approaches based on embedding models by over 65% in terms of average precision for the top-recommended EF.[12,19,20] We also open-source our code and release new public benchmarks based on real-world datasets, enabling broader research and development in this area.

Acknowledgement

The authors are grateful to Amazon for sponsoring this research. Aravind Srinivasan's contribution to this publication is not part of his University of Maryland duties or responsibilities.

Supporting Information Available

The following file is available free of charge.

- **Supporting Information: Emission Factor Recommendation for Life Cycle Assessments with Generative AI** (PDF): Additional information on prompts, NAICS codes, and Parakeet implementation.

References

(1) Black, R.; Cullen, K.; Fay, B.; Hale, T.; Lang, J.; Mahmood, S.; Smith, S. Taking stock: A global assessment of net zero targets. *Energy & Climate Intelligence Unit and Oxford Net Zero* **2021**, *23*.

(2) The Climate Pledge Be the planet's turning point: Join the world's top companies—and take action now to reach net-zero carbon by 2040. `https://www.theclimatepledge.com/`, accessed 2025-03-14.

(3) US Securities and Exchange Commission SEC proposes rules to enhance and standardize climate-related disclosures for investors. 2022; `https://www.sec.gov/newsroom/press-releases/2022-46`, accessed 2025-03-14.

(4) World Resources Institute and World Business Council for Sustainable Development The Greenhouse Gas Protocol Standard. 2011; `https://ghgprotocol.org/sites/default/files/standards/ghg-protocol-revised.pdf`, accessed 2025-03-14.

(5) Hauschild, M. Z.; Rosenbaum, R. K.; Olsen, S. I. *Life cycle assessment*; Springer, 2018.

(6) ISO, I. O. f. S. *Environmental management: life cycle assessment; Principles and Framework*; ISO, 2006.

(7) Tasaki, T.; Shobatake, K.; Nakajima, K.; Dalhammar, C. International survey of the costs of assessment for environmental product declarations. *Procedia CIRP* **2017**, *61*, 727–731.

(8) Balaji, B.; Guest, G.; Vunnava, V. S. G.; Kramer, J.; Srinivasan, A.; Taptich, M. Scaling carbon footprinting: Challenges and opportunities. Proceedings of the AAAI Symposium Series. 2023; pp 35–39.

(9) Meinrenken, C. J.; Kaufman, S. M.; Ramesh, S.; Lackner, K. S. Fast carbon footprinting for large product portfolios. *Journal of Industrial Ecology* **2012**, *16*, 669–679.

(10) Wernet, G.; Bauer, C.; Steubing, B.; Reinhard, J.; Moreno-Ruiz, E.; Weidema, B. The ecoinvent database version 3 (part I): overview and methodology. *The International Journal of Life Cycle Assessment* **2016**, *21*, 1218–1230.

(11) Ingwersen, W. W.; Li, M.; Young, B.; Vendries, J.; Birney, C. USEEIO v2. 0, the US environmentally-extended input-output model v2. 0. *Scientific Data* **2022**, *9*, 194.

(12) Balaji, B.; Vunnava, V. S. G.; Domingo, N.; Gupta, S.; Gupta, H.; Guest, G.; Srinivasan, A. Flamingo: Environmental Impact Factor Matching for Life Cycle Assessment with Zero-shot Machine Learning. *ACM Journal on Computing and Sustainable Societies* **2023**, *1*, 1–23.

(13) Sphera Product Life Cycle Assessment Solutions. **2025**, accessed 2025-03-14.

(14) Schrijvers, D. L.; Loubet, P.; Sonnemann, G. Developing a systematic framework for consistent allocation in LCA. *The International Journal of Life Cycle Assessment* **2016**, *21*, 976–993.

(15) Jolliet, O.; Margni, M.; Charles, R.; Humbert, S.; Payet, J.; Rebitzer, G.; Rosenbaum, R. IMPACT 2002+: a new life cycle impact assessment methodology. *The international journal of life cycle assessment* **2003**, *8*, 324–330.

(16) Kikstra, J. S.; Nicholls, Z. R.; Smith, C. J.; Lewis, J.; Lamboll, R. D.; Byers, E.; Sandstad, M.; Meinshausen, M.; Gidden, M. J.; Rogelj, J.; others The IPCC Sixth Assessment Report WGIII climate assessment of mitigation pathways: from emissions to global temperatures. *Geoscientific Model Development* **2022**, *15*, 9075–9109.

(17) Yang, Y.; Ingwersen, W. W.; Hawkins, T. R.; Srocka, M.; Meyer, D. E. USEEIO: A new and transparent United States environmentally-extended input-output model. *Journal of cleaner production* **2017**, *158*, 308–318.

(18) Sousa, I.; Wallace, D. Product classification to support approximate life-cycle assessment of design concepts. *Technological Forecasting and Social Change* **2006**, *73*, 228–249.

(19) Luo, B.; Liu, J.; Deng, Z.; Yuan, C.; Yang, Q.; Xiao, L.; Xie, Y.; Zhou, F.; Zhou, W.; Liu, Z. AutoPCF: A Novel Automatic Product Carbon Footprint Estimation Framework Based on Large Language Models. Proceedings of the AAAI Symposium Series. 2023; pp 102–106.

(20) Balaji, B.; Vunnava, V. S. G.; Guest, G.; Kramer, J. CaML: Carbon footprinting of household products with zero-shot semantic text similarity. Proceedings of the ACM Web Conference 2023. 2023; pp 4004–4014.

(21) Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019; pp 3982–3992.

(22) Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; others Language models are few-shot learners. *Advances in neural information processing systems* **2020**, *33*, 1877–1901.

(23) Anthropic The Claude 3 Model Family: Opus, Sonnet, Haiku. 2025; `https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf`, accessed 2025-03-14.

(24) Chiesurin, S.; Dimakopoulos, D.; Cabezudo, M. A. S.; Eshghi, A.; Papaioannou, I.; Rieser, V.; Konstas, I. The Dangers of trusting Stochastic Parrots: Faithfulness and Trust in Open-domain Conversational Question Answering. Findings of the Association for Computational Linguistics: ACL 2023. 2023; pp 947–959.

(25) Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; others Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* **2020**, *33*, 9459–9474.

(26) Izacard, G.; Grave, É. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume. 2021; pp 874–880.

(27) Liu, X.; Zheng, Y.; Du, Z.; Ding, M.; Qian, Y.; Yang, Z.; Tang, J. GPT understands, too. *AI Open* **2023**, *5*, 208–215.

(28) Lester, B.; Al-Rfou, R.; Constant, N. The Power of Scale for Parameter-Efficient Prompt Tuning. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. 2021; pp 3045–3059.

(29) Chiang, W.-L.; Zheng, L.; Sheng, Y.; Angelopoulos, A. N.; Li, T.; Li, D.; Zhang, H.; Zhu, B.; Jordan, M.; Gonzalez, J. E.; Stoica, I. Chatbot Arena: An Open Platform

for Evaluating LLMs by Human Preference. 2025; https://huggingface.co/spaces/lmarena-ai/chatbot-arena-leaderboard, accessed 2025-03-14.

(30) Penedo, G.; Kydlíček, H.; Lozhkov, A.; Mitchell, M.; Raffel, C.; Von Werra, L.; Wolf, T.; others The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale. The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track.

(31) Mba Wright, M.; Tan, E. C.; Tu, Q.; Martins, A.; Parvatker, A. G.; Yao, Y.; Sunol, A.; Smith, R. L. Life Cycle Inventory Availability: Status and Prospects for Leveraging New Technologies. *ACS Sustainable Chemistry & Engineering* **2024**, *12*, 12708–12718.

(32) Zargar, S.; Yao, Y.; Tu, Q. A review of inventory modeling methods for missing data in life cycle assessment. *Journal of Industrial Ecology* **2022**, *26*, 1676–1689.

(33) Li, D.; Qin, J.; Hong, J. Toward a comprehensive life cycle aquatic ecotoxicity assessment via machine learning: Application to coal power generation in China. *Journal of Cleaner Production* **2024**, *445*, 141373.

(34) von Borries, K.; Holmquist, H.; Kosnik, M.; Beckwith, K. V.; Jolliet, O.; Goodman, J. M.; Fantke, P. Potential for machine learning to address data gaps in human toxicity and ecotoxicity characterization. *Environmental Science & Technology* **2023**, *57*, 18259–18270.

(35) Romeiko, X. X.; Zhang, X.; Pang, Y.; Gao, F.; Xu, M.; Lin, S.; Babbitt, C. A review of machine learning applications in life cycle assessment studies. *Science of The Total Environment* **2024**, *912*, 168969.

(36) Li, Z.; Zhang, X.; Zhang, Y.; Long, D.; Xie, P.; Zhang, M. Towards General Text Embeddings with Multi-stage Contrastive Learning. *Submitted 2023-08-07, arXiv 2308.03281 [cs.CL],* https://arxiv.org/abs/2308.03281, (accessed 2025-03-14).

(37) Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. 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; pp 4171–4186.

(38) Oord, A. v. d.; Li, Y.; Vinyals, O. Representation learning with contrastive predictive coding. *Submitted 2019-01-22, arXiv 1807.03748 [cs.LG],* `https://arxiv.org/abs/1807.03748`, (accessed 2025-03-14).

(39) Muennighoff, N.; Tazi, N.; Magne, L.; Reimers, N. MTEB: Massive Text Embedding Benchmark. Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics. 2023; pp 2014–2037.

(40) MMTEB: Massive Multilingual Text Embedding Benchmark. 2025; `https://huggingface.co/spaces/mteb/leaderboard`, accessed 2025-03-14.

(41) Anthropic Meet Claude. 2025; `https://www.anthropic.com/claude`, accessed 2025-03-14.

(42) Meta Llama. 2025; `https://www.llama.com/`, accessed 2025-03-14.

(43) Mistral AI Mistral Models Overview. 2025; `https://docs.mistral.ai/getting-started/models/models_overview/`, accessed 2025-03-14.

(44) Joulin, A.; Grave, É.; Bojanowski, P.; Mikolov, T. Bag of Tricks for Efficient Text Classification. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. 2017; pp 427–431.

(45) Kelly, D.; Azzopardi, L. How many results per page? A study of SERP size, search behavior and user experience. Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval. 2015; pp 183–192.

(46) Nguyen, J. Purchase Order Quantity Price detail for Commodity/Goods procurements. 2021; https://catalog.data.gov/dataset/purchase-order-quantity-price-detail-for-commodity-goods-procurements, accessed 2025-03-14.

(47) Kozłowski, M.; Weichbroth, P. Samples of electronic invoices. 2021; https://data.mendeley.com/datasets/tnj49gpmtz/2, accessed 2025-03-14.

(48) Li, S. Food.com Recipes and Interactions. 2019; https://www.kaggle.com/datasets/shuyangli94/food-com-recipes-and-user-interactions, accessed 2025-03-14.

(49) O'Donnell, B.; Goodchild, A.; Cooper, J.; Ozawa, T. The relative contribution of transportation to supply chain greenhouse gas emissions: A case study of American wheat. *Transportation Research Part D: Transport and Environment* **2009**, *14*, 487–492.