Flamingo: Environmental Impact Factor Matching for Life Cycle Assessment with Zero-Shot ML

Bharathan Balaji
Amazon
Seattle, WA
bhabalaj@amazon.com

Venkata Sai Gargeya Vunnava
Amazon
Seattle, WA
gvunnava@amazon.com

Nina Domingo
Amazon
New York, NY
nggdom@amazon.com

Shikhar Gupta
Amazon
Seattle, WA
gupshik@amazon.com

Harsh Gupta
Amazon
San Francisco, CA
hrshgup@amazon.com

Geoffrey Guest
Amazon
Seattle, WA
gmg@amazon.com

Aravind Srinivasan
Amazon and University of Maryland
Baltimore, MD
srinarav@amazon.com

Abstract

Consumer products contribute to >75% of global greenhouse gas (GHG) emissions, primarily through indirect contributions from the supply chain. Measurement of GHG emissions associated with products is crucial to quantify the impact of GHG emission abatement actions. Life cycle assessment (LCA), the scientific discipline for measuring GHG emissions, estimates the environmental impact of a product. Scaling LCA to millions of products is challenging as it requires extensive manual analysis by domain experts. To avoid repetitive analysis, environmental impact factors (EIF) of common materials and products are published for use by experts. However, finding appropriate EIFs for even a single product can require hundreds of hours of manual work, especially for complex products. We present Flamingo, an algorithm that leverages neural language models to automatically identify an appropriate EIF given a text description. A key challenge in automation is that EIF databases are incomplete. Flamingo uses industry sector classification as an intermediate layer to identify when there are no good matches in the database. On a dataset of 664 products, Flamingo achieves an EIF matching precision of 75%.

1 Introduction

Life cycle assessment (LCA) is a standard method used to estimate GHG emissions associated with an activity or a product. These emissions are often referred to as its carbon footprint, and are reported in terms of global-warming potential in units of mass of carbon dioxide equivalent [Gao et al.(2014)]. LCA estimates emissions in each stage of a product: raw material extraction, manufacturing, transportation, use, and disposal/reuse. The effort to acquire direct measurements for each aspect can be prohibitively expensive [Tasaki et al.(2017)], and therefore, domain experts use the outputs of existing LCA studies to estimate emissions of common materials, products, and activities associated with the life cycle of their subject [Wernet et al.(2016)]. E.g., an LCA on the "production of a cotton t-shirt" might rely on the results of LCAs focused on "cotton production" or "transport by truck". We

Tackling Climate Change with Machine Learning: workshop at NeurIPS 2023.

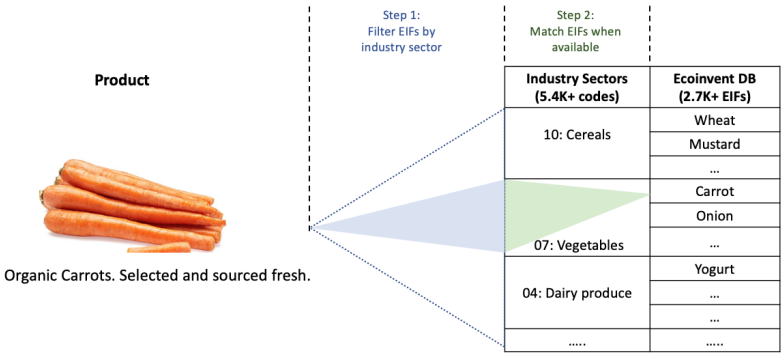


Figure 1: Environmental impact factor (EIF) selection is a key aspect of product carbon footprinting. Flamingo automates selection of an EIF using pre-trained neural language models. It uses industry sector codes to identify when an EIF is not available in the database.

refer to the outputs of these reference LCA studies as *environment impact factors* (EIFs). Databases such as Ecoinvent [Wernet et al.(2016)] collate these EIFs for use by LCA practitioners.

There is a lack of automation tools that integrate with EIF datasets. Often, EIFs are shared as spreadsheets, and experts use string matching or explicit rules to match an activity to an appropriate EIF [Rovelli et al.(2022)]. It can take hundreds of hours to match EIFs for LCA of even a single product [Meinrenken et al.(2012)]. To mitigate manual overhead, we propose using natural language processing (NLP) based machine learning (ML) methods to perform this matching. To our knowledge, we are the first to study use of neural ML based EIF matching. A simple approach is to use neural language models trained on web data for semantic textual similarity [Reimers and Gurevych(2019)], where the EIFs can be matched based on the distance between the embedding of the query and the EIF texts. A major advantage of this approach is that no training data is required, as off-the-shelf pre-trained models can capture synonyms and conceptual relationships.

A key characteristic of the EIF matching problem is that the databases are incomplete. Many products such as mushrooms or socks do not have an EIF, because either no LCA study exists or the published LCAs have not been ingested into the database. LCA experts search across multiple databases for a match, and use an approximate value with higher uncertainty when no match exists, e.g., average EIF of vegetables as a proxy for red bell peppers [Clark et al.(2022)]. However, pre-trained ML models are not trained to identify when an appropriate match does not exist.

We present a novel zero-shot ML algorithm, called Flamingo, that introduces an intermediate classification layer in semantic search to identify when an EIF is missing, and improves the performance of EIF matches. Flamingo classifies the input query text to a standard industry code, and uses semantic text matching to identify the closest EIFs within the industry code. When there are no EIFs available for an industry code, Flamingo predicts that no appropriate match exists. We use an industry sector classification called the Harmonized System (HS) code [Chaplin(1987)], which is specifically designed for categorizing products based on their material composition and manufacturing complexity. HS codes are hierarchically organized, are used globally for import/export taxes, and are refreshed by the World Customs Organization every 5 years [Wolffgang and Dallimore(2011)]. Flamingo exploits the HS code hierarchy to navigate the precision versus recall trade-off associated with an EIF match.

We evaluate Flamingo on a dataset of 664 products from an e-commerce retailer. We use annotations from crowd workers to identify if an EIF predicted by Flamingo is an appropriate match, or if no match exists. Our results show that Flamingo matches EIFs to products with a precision of 75%, and outperforms the semantic text similarity baseline by 8.4%. We open-source our code with a permissive license. Background on LCA and comparison to related work is given in Appendix A.

2 Problem Statement

Given a query text $q_i \in \mathcal{Q}$, our objective is to find an appropriate EIF e_i from a given set of EIFs $\mathcal{E}$. The query text can be a product description or a specific aspect of a product that an LCA expert will attach an EIF to such as the material a product is made of. All EIF datasets include text metadata that

describes its characteristics, and we assume the text is available for query matching. It is possible that there is no appropriate EIF available for a given query, and the algorithm should output $\varnothing$, i.e. 'no match', for such cases.

We assume there is no training dataset available that matches query text to EIFs. It is difficult to obtain high-quality annotations for a dataset that is large enough to train models which generalize to all queries and EIFs. Although we do have a small dataset, we only use it for validation of methods. Even for few-shot methods, we need to have a few labeled examples per EIF, which are in the thousands. It is possible to use EIF text descriptions to reduce the reliance on labeled examples for every class [Zhang et al.(2018)], we leave exploration of such methods for future work.

3 Methods

Given a query text q_i, our objective is to both predict if a matching EIF exists in the database, and retrieve the matching EIF e_i when one exists. Critically, we do not rely on a supervised dataset of query to EIFs commonly used in prior works [Sun et al.(2021), Hu et al.(2019)]. Zero-shot matching algorithms, such as SBERT [Reimers and Gurevych(2019)], can identify the best matching EIF but are inadequate at identifying when the EIF is not a good match based on a distance threshold. We improve on SBERT using industry sector classification as defined through HS codes [Chaplin(1987)]. We only consider the first 6 digits of the HS codes, called HS6, as they are globally applicable.

We use a model M_H that predicts the HS6 code for a given text input τ_i. Given a HS6 code, we can lookup the corresponding HS4 and HS2 codes from the code hierarchy. Therefore, we have: $h_i^\delta = M_H(\tau_i, \mathcal{H})$. We use a zero-shot method that leverages the text description of HS6 codes and finds the best match using SBERT. For a given query text q_i, we first predict its HS codes h_q^δ if not provided. Next, we find the best EIF e_i that matches the given query q_i using the SBERT model M_E. We then predict the corresponding HS code h_e^δ for the EIF e_i using the model M_H. If the HS codes of both the EIF and the query match, and the cosine similarity score is higher than the threshold, we output the best match EIF e_i as the final prediction. Otherwise, the output is 'no match' $\varnothing$. We include three variants of the algorithm, where we match EIFs to query based on their HS2, HS4, and HS6 codes respectively. Reducing the number of digits in the HS code helps reduce the specificity of the classification. Therefore, a higher-level HS code increases the precision of finding an appropriate EIF match at the expense of reducing the precision of predicting 'no match'. We append the HS code we use to the name of algorithm to specify the variants, e.g., FlamingoHS4. We use the 'all-mpnet-base-v2' model from `sbert.net` as it has the highest performance in semantic text similarity benchmarks. We use the default hyper-parameters throughout, where the input sentence is cutoff after 128 tokens (about 100 words).

We use the same model to predict both the HS codes (M_H), and to rank the best match EIFs (M_E). For example, to predict the EIF match for a given query text, the model will output the EIF embedding that is closest to the query embedding as measured through cosine similarity. It is possible that there is a 'no match' even after filtering out EIFs. We use a threshold on cosine similarity distance to further filter out unrelated EIFs, and show the impact of using both a conservative and an aggressive threshold on the algorithm performance.

4 Evaluation

We evaluate EIF predictions by BM25 and SBERT as our baseline algorithms. For BM25, we use the implementation by [Trotman et al.(2014)], and use default hyper-parameters. For both BM25 and SBERT, we use a distance threshold that maximizes their overall performance. For Flamingo, we include the HS2, HS4, and HS6 based predictions of Flamingo. We consider two cosine similarity thresholds for SBERT, 0 (conservative) and 0.5 (aggressive). These choices show the trade-offs in design choices of the Flamingo algorithm, and we avoid tuning these parameters on our dataset.

We consider a dataset of 664 products labeled by the annotation team. As the annotation is performed on a ranked list of EIFs provided by FlamingoZero, recall cannot be accurately measured. Therefore, we report overall Precision@1, Macro Precision, and Weighted Precision scores. Precision@1, or simply precision, indicates the metric is for the top-ranked candidate, and could be generalized to Precision@K for top-K items. It is a common metric used for text ranking [Joulin et al.(2017)]. We

Table 1: Evaluation results for Flamingo with a dataset of 664 products. Ground truth is obtained by majority vote on three annotations per product.

Method	Distance threshold	Precision@1 (%)			Macro Precision	Weighted Precision
		Overall	No Match	Match		
BM25	35	53.0	87.1	5.4	7.8	46.0
SBERT	0.5	66.6	85.8	39.7	13.9	66.8
SBERT + BM25	0.5	59.0	99.0	3.2	8.1	44.5
FlamingoHS2	0.0	67.2	78.3	51.6	**18.0**	71.3
FlamingoHS4	0.0	**75.0**	96.3	45.1	16.4	64.0
FlamingoHS6	0.0	61.4	98.7	9.4	11.0	55.2
FlamingoHS2	0.5	67.8	93.8	31.4	17.3	62.9
FlamingoHS4	0.5	70.2	99.2	29.6	15.7	60.7
FlamingoHS6	0.5	59.5	**99.7**	3.2	9.4	54.2
Human Performance						
Human (Mean)	–	76.4	76.1	75.5	42.0	86.2

also breakdown the overall precision by 'No match' and 'Match'. By design, >50% of the products in the dataset do not have a matching EIF to reflect the importance of predicting 'No match' in practice.

Table 1 shows the results. Both BM25 and SBERT perform quite well, giving 53.0% and 66.6% Precision@1 respectively. The choice of threshold determines the trade-off between increasing the precision of 'Match' vs 'No match'. Flamingo provides an additional method to navigate the same trade-off with HS codes based classification as an intermediate layer, and helps improve performance by increasing the 'No match' precision, and filtering out erroneous EIFs based on their HS code sector. Flamingo increases the probability of 'No match' as we increase the specificity of HS codes from 2 to 6 digits. However, it comes at the cost of decreased 'Match' precision. HS4 codes provide a good trade-off between the two extremes, and yield the best overall performance. Addition of cosine similarity threshold on EIF selection further increases the probability of a 'No match' prediction. The threshold can be adjusted based on the requirements of downstream applications.

The human performance metrics show there is a significant room for improvement in EIF matching algorithms, especially in selection of an appropriate EIF when it is available in the database. Even when cosine similarity threshold is set to 0, and no EIFs are filtered based on HS codes, the best 'Match' precision is $\sim$50%, far below the human-level precision of 75.5%. This observation points to an opportunity to improve on the SBERT model, perhaps by fine-tuning on sentences used in EIFs.

Appendix D includes additional results on a food ingredient dataset, analysis of errors, and impact on carbon footprint estimates.

5 Conclusion and Future Work

We have presented an algorithm that automates EIF matching for a given query text. Our algorithm, Flamingo, requires no training data and exploits industry codes to predict whether an EIF exists in the database as well as identifies the best matching EIF when it exists. While we have focused on GHG emissions throughout the paper, the EIF databases include impact estimates for additional categories such as hazardous wastes, fresh water use, and air pollution. Our algorithms can be easily extended to these impacts.

We can extend text-based matching to image-based matching using CLIP [Radford et al.(2021)], which generates an embedding with an image-to-text correspondence. The key to further improvement in EIF matching performance is to improve the prediction of HS codes from EIF metadata. Future work can consider zero-shot contrastive learning approaches such as MACLR [Xiong et al.(2022)] that can exploit EIF text descriptions to learn correspondence with HS codes. Flamingo provides a promising start on a critical aspect of LCA, but a number of challenges remain in scaling to millions of consumer products. As seen from our results, EIFs of most consumer products in the market are not available in the database, and more EIFs are added manually from publicly available documents. Automation of EIF extraction from documents is a promising avenue of future work.

References

[Bhatia et al.(2016)] K. Bhatia, K. Dahiya, H. Jain, P. Kar, A. Mittal, Y. Prabhu, and M. Varma. 2016. The extreme classification repository: Multi-label datasets and code. `http://manikvarma.org/downloads/XC/XMLRepository.html`

[Bhatia et al.(2015)] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. 2015. Sparse local embeddings for extreme multi-label classification. *Advances in neural information processing systems* 28 (2015).

[Boyer and Moore(1977)] Robert S Boyer and J Strother Moore. 1977. A fast string searching algorithm. *Commun. ACM* 20, 10 (1977), 762–772.

[Bush et al.(1945)] Vannevar Bush et al. 1945. As we may think. *The Atlantic Monthly* 176, 1 (1945), 101–108.

[Cer et al.(2018)] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 conference on empirical methods in natural language processing: system demonstrations*. 169–174.

[Chaplin(1987)] Peggy Chaplin. 1987. An Introduction to the Harmonized System. *NCJ Int'l L. & Com. Reg.* 12 (1987), 417.

[Chen et al.(2021)] Xi Chen, Stefano Bromuri, and Marko Van Eekelen. 2021. Neural machine translation for harmonized system codes prediction. In *2021 6th International Conference on Machine Learning Technologies*. 158–163.

[Clark et al.(2022)] Michael Clark, Marco Springmann, Mike Rayner, Peter Scarborough, Jason Hill, David Tilman, Jennie I Macdiarmid, Jessica Fanzo, Lauren Bandy, and Richard A Harrington. 2022. Estimating the environmental impacts of 57,000 food products. *Proceedings of the National Academy of Sciences* 119, 33 (2022), e2120584119.

[Colomb et al.(2015)] Vincent Colomb, Samy Ait Amar, Claudine Basset Mens, Armelle Gac, Gérard Gaillard, Peter Koch, Jerome Mousset, Thibault Salou, Aurélie Tailleur, and Hays MG van der Werf. 2015. AGRIBALYSE®, the French LCI Database for agricultural products: high quality data for producers and environmental labelling. *OCL* 22, 1 (2015), D104.

[Dhamija et al.(2018)] Akshay Raj Dhamija, Manuel Günther, and Terrance Boult. 2018. Reducing network agnostophobia. *Advances in Neural Information Processing Systems* 31 (2018).

[Ding et al.(2015)] Liya Ding, ZhenZhen Fan, and DongLiang Chen. 2015. Auto-categorization of HS code using background net approach. *Procedia Computer Science* 60 (2015), 1462–1471.

[Du et al.(2021)] Shaohua Du, Zhihao Wu, Huaiyu Wan, and YouFang Lin. 2021. HScodeNet: Combining Hierarchical Sequential and Global Spatial Information of Text for Commodity HS Code Classification. In *Advances in Knowledge Discovery and Data Mining: 25th Pacific-Asia Conference, PAKDD 2021, Virtual Event, May 11–14, 2021, Proceedings, Part II*. Springer, 676–689.

[for Standardization(2006)] International Organization for Standardization. 2006. *Environmental management: life cycle assessment; Principles and Framework*. ISO.

[Gao et al.(2014)] Tao Gao, Qing Liu, and Jianping Wang. 2014. A comparative study of carbon footprint and assessment standards. *International Journal of Low-Carbon Technologies* 9, 3 (2014), 237–243.

[Gormley and Tong(2015)] Clinton Gormley and Zachary Tong. 2015. *Elasticsearch: the definitive guide: a distributed real-time search and analytics engine*. " O'Reilly Media, Inc.".

[Guinée et al.(2011)] Jeroen B Guinée, Reinout Heijungs, Gjalt Huppes, Alessandra Zamagni, Paolo Masoni, Roberto Buonamici, Tomas Ekvall, and Tomas Rydberg. 2011. Life cycle assessment: Past, present, and future. *Environmental Science and Technology* 45, 1 (2011), 90–96.

[Hu et al.(2014)] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. *Advances in neural information processing systems* 27 (2014).

[Hu et al.(2019)] Minghao Hu, Furu Wei, Yuxing Peng, Zhen Huang, Nan Yang, and Dongsheng Li. 2019. Read+ verify: Machine reading comprehension with unanswerable questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6529–6537.

[Huang et al.(2013)] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. 2333–2338.

[Hunt(1974)] Robert G Hunt. 1974. *Resource and environmental profile analysis of nine beverage container alternatives*. Vol. 91. Environmental Protection Agency.

[Ingwersen et al.(2016)] Wesley Ingwersen, Maria Gausman, Annie Weisbrod, Debalina Sengupta, Seung-Jin Lee, Jane Bare, Ed Zanoli, Gurbakash S Bhander, and Manuel Ceja. 2016. Detailed life cycle assessment of Bounty® paper towel operations in the United States. *Journal of cleaner production* 131 (2016), 509–522.

[Jalalzai et al.(2020)] Hamid Jalalzai, Pierre Colombo, Chloé Clavel, Eric Gaussier, Giovanna Varni, Emmanuel Vignon, and Anne Sabourin. 2020. Heavy-tailed representations, text polarity classification & data augmentation. *Advances in Neural Information Processing Systems* 33 (2020), 4295–4307.

[Joulin et al.(2017)] Armand Joulin, Édouard Grave, Piotr Bojanowski, and Tomáš Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. 427–431.

[Kenton and Toutanova(2019)] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*. 4171–4186.

[Khattab and Zaharia(2020)] Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 39–48.

[Krippendorff([n. d.])] Klaus Krippendorff. [n. d.]. Computing Krippendorff's Alpha-Reliability. *Computing* 1 ([n. d.]), 25–2011.

[Meinrenken et al.(2012)] Christoph J Meinrenken, Scott M Kaufman, Siddharth Ramesh, and Klaus S Lackner. 2012. Fast carbon footprinting for large product portfolios. *Journal of Industrial Ecology* 16, 5 (2012), 669–679.

[Mitra et al.(2018)] Bhaskar Mitra, Nick Craswell, et al. 2018. An introduction to neural information retrieval. *Foundations and Trends® in Information Retrieval* 13, 1 (2018), 1–126.

[Nations(1969)] United Nations. 1969. *International standard industrial classification of all economic activities*. UN.

[Pariag(2009)] Peter Pariag. 2009. Classification of services. In *Regional Symposium on Services*. 15–17.

[Radford et al.(2021)] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

[Radford et al.(2018)] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[Ramos et al.(2003)] Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, Vol. 242. Citeseer, 29–48.

[Reimers and Gurevych(2019)] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 3982–3992.

[Robertson et al.(2009)] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[Robertson et al.(1995)] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at TREC-3. *Nist Special Publication Sp* 109 (1995), 109.

[Rovelli et al.(2022)] Davide Rovelli, Carlo Brondi, Michele Andreotti, Elisabetta Abbate, Maurizio Zanforlin, and Andrea Ballarino. 2022. A Modular Tool to Support Data Management for LCA in Industry: Methodology, Application and Potentialities. *Sustainability* 14, 7 (2022), 3746.

[Singhal et al.(2001)] Amit Singhal et al. 2001. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.* 24, 4 (2001), 35–43.

[Sparck Jones(1972)] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

[Sphera(2023)] Sphera. 2023. Life Cycle Assessment Product Sustainability (GaBi) Software. `https://sphera.com/life-cycle-assessment-lca-software/`. (2023).

[Standard(2011)] GHG Protocol Standard. 2011. The greenhouse gas protocol.

[Sun et al.(2021)] Zequn Sun, Muhao Chen, and Wei Hu. 2021. Knowing the No-match: Entity Alignment with Dangling Cases. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 3582–3593.

[Tasaki et al.(2017)] Tomohiro Tasaki, Koichi Shobatake, Kenichi Nakajima, and Carl Dalhammar. 2017. International survey of the costs of assessment for environmental product declarations. *Procedia CIRP* 61 (2017), 727–731.

[Thompson(1968)] Ken Thompson. 1968. Programming techniques: Regular expression search algorithm. *Commun. ACM* 11, 6 (1968), 419–422.

[Trotman et al.(2014)] Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to BM25 and language models examined. In *Proceedings of the 2014 Australasian Document Computing Symposium*. 58–65.

[Vaswani et al.(2017)] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[Wernet et al.(2016)] Gregor Wernet, Christian Bauer, Bernhard Steubing, Jürgen Reinhard, Emilia Moreno-Ruiz, and Bo Weidema. 2016. The ecoinvent database version 3 (part I): overview and methodology. *The International Journal of Life Cycle Assessment* 21, 9 (2016), 1218–1230.

[Wolffgang and Dallimore(2011)] Hans-Michael Wolffgang and Christopher Dallimore. 2011. The World Customs Organization and its Role in the System of World Trade: An Overview. *European Yearbook of International Economic Law 2012* (2011), 613–633.

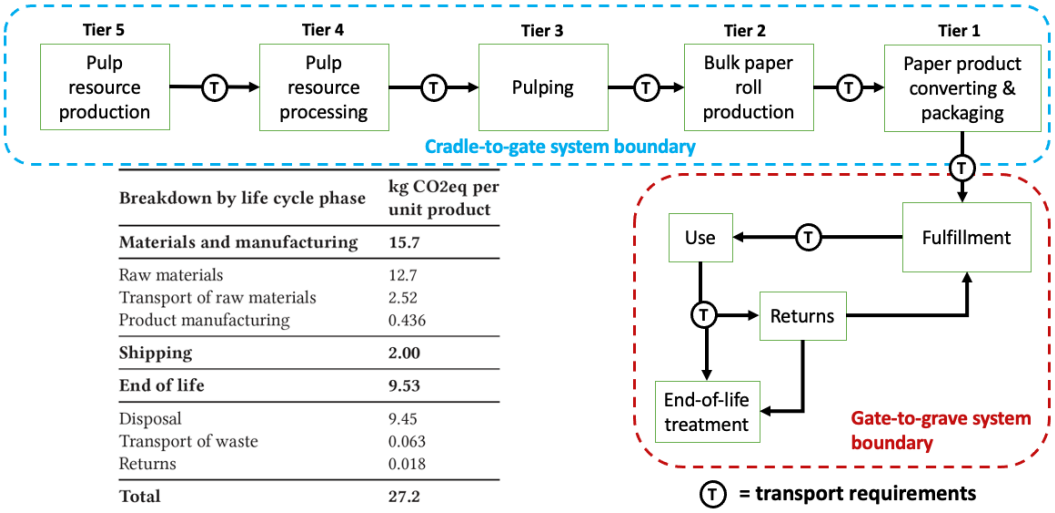
[Xiong et al.(2022)] Yuanhao Xiong, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, and Inderjit Dhillon. 2022. Extreme Zero-Shot Learning for Extreme Text Classification. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 5455–5468.

[Yang et al.(2017)] Yi Yang, Wesley W Ingwersen, Troy R Hawkins, Michael Srocka, and David E Meyer. 2017. USEEIO: A new and transparent United States environmentally-extended input-output model. *Journal of cleaner production* 158 (2017), 308–318.

[Yu et al.(2022)] Hsiang-Fu Yu, Jiong Zhang, Wei-Cheng Chang, Jyun-Yu Jiang, Wei Li, and Cho-Jui Hsieh. 2022. Pecos: Prediction for enormous and correlated output spaces. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4848–4849.

[Zeng et al.(2021)] Weixin Zeng, Xiang Zhao, Jiuyang Tang, Xinyi Li, Minnan Luo, and Qinghua Zheng. 2021. Towards entity alignment in the open world: An unsupervised approach. In *Database Systems for Advanced Applications: 26th International Conference, DASFAA 2021, Taipei, Taiwan, April 11–14, 2021, Proceedings, Part I 26*. Springer, 272–289.

[Zhang et al.(2018)] Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. 2018. Multi-Task Label Embedding for Text Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4545–4553.



Breakdown by life cycle phase	kg CO2eq per unit product
Materials and manufacturing	**15.7**
Raw materials	12.7
Transport of raw materials	2.52
Product manufacturing	0.436
Shipping	**2.00**
End of life	**9.53**
Disposal	9.45
Transport of waste	0.063
Returns	0.018
Total	**27.2**

Figure 2: Life cycle assessment of a paper towel made with 75% recycled materials [Ingwersen et al.(2016)].

A Background and Related Work

In this section, we briefly explain how LCA is used for carbon footprint estimation, and the role of EIFs in performing LCAs. We summarize the prior works on automation of matching EIFs for a given query in the LCA literature, and the key challenges that remain unaddressed. We then summarize the relevant text-matching literature, along with its applicability to the EIF matching problem.

A.1 Life cycle assessment

LCA was introduced as a systematic method to compare product design choices in terms of energy use, waste, and other environmental impacts [Hunt(1974), Guinée et al.(2011)]. It has since been adopted as a standard method for carbon footprinting by both the International Standards Organization [for Standardization(2006)] and the GHG Protocol [Standard(2011)]. LCA can be categorized into two types: Economic Input-Output LCA (EIO-LCA) and Process-LCA. EIO-LCA uses transactions across industries in an economy to obtain an approximate impact assessment at an industry sector level [Yang et al.(2017)]. This type of LCA is associated with aggregation issues, as for example, different types of paper products are assumed to have the same GHG impact per unit of sale price regardless of how they were manufactured. Process-LCA, on the other hand, produce higher granularity carbon footprints through detailed tracking of emissions from each life-cycle stage of a product. Figure 2 shows a Process-LCA of a paper towel as an example, which requires accounting of how the pulp was sourced, how it was processed across multiple suppliers, how materials were transported between stakeholders, what percentage of products sold was returned, and whether the paper roll was composted, landfilled, or recycled. As it is challenging, and sometimes infeasible, to collect direct emissions data in such high detail, LCA experts use EIFs published in prior studies as an estimate of the GHG emissions associated with a product, material, or activity for which they do not have direct measurements [Meinrenken et al.(2012), Rovelli et al.(2022)]. Ecoinvent [Wernet et al.(2016)], GaBi [Sphera(2023)], and AGRIBALYSE [Colomb et al.(2015)] are some of the common EIF databases used in the industry.

Finding the right EIF can be time consuming [Meinrenken et al.(2012)] as exact string match does not capture synonyms (e.g., milk and dairy, maize and corn), abbreviations (e.g., Ni-Cd and Nickel Cadmium), technical terms (e.g., sodium chloride is same as salt), or category relationships (e.g., basmati is a type of rice). A few solutions have been proposed to overcome this challenge in the literature [Meinrenken et al.(2012), Clark et al.(2022), Rovelli et al.(2022)]. meinrenken2012fast propose a linear regression algorithm, where they categorize the query text to an industry sector, and use a regression based on price to estimate the GHG emissions. However, the categorization is done manually, all the EIFs within an industry sector are averaged together which increases the variance of

the estimate, and heuristics are used to remove outliers. clark2022estimating matched ingredients to EIFs for food products. They also rely on a mapping of EIFs to pre-defined food categories (e.g., berries, cheese). In addition, they manually create search terms which are synonyms or sub-types of a given food category (e.g., pecorino is a cheese) so they can improve exact string matches. To reduce variance of emission estimates, they use a three-level hierarchical categorization so that specificity of a match increases when possible, e.g., use strawberry instead of berries. Our algorithm uses a similar hierarchical industry classification, but does not require manual specification of search terms. As a result, our solution scales to all EIFs in the database, and is not limited to food EIFs. In contrast to these prior works, Flamingo does not require manual steps or heuristics, and can match to specific EIFs in the database instead of industry sectors.

ML has been used to automate other aspects of LCA, as covered in a survey by algren2021machine. Specific to EIF estimation, sousa2006product directly estimate carbon dioxide emissions based on product attributes to inform design decisions. They used neural network regression on carefully chosen product descriptors to estimate emissions. They train different models for high and low energy use products. The model was trained on EIFs of 53 products, and predictions on a test set of six products gave an error of 40%. However, their product descriptor required details such as mass, percentage contribution from different type of materials (ceramics, fibers, metals, plastics, etc.), energy source and more. In contrast, we identify if an EIF in the database can be matched to a given text description and do not place any restrictions on the input. We also test our predictions on a much larger dataset of 664 products.

A.2 Text Matching

Text search algorithms were envisioned with the advent of digital computers [Bush et al.(1945), Singhal et al.(2001)]. Much of the early search algorithms were based on exact string matching [Thompson(1968), Boyer and Moore(1977)], and is still prevalent in spreadsheets used for EIF searching by LCA experts. Exact matches work well for small strings, but have poor recall with long search queries. BM25 [Robertson et al.(1995), Robertson et al.(2009)] and TFIDF [Ramos et al.(2003), Sparck Jones(1972)] are probabilistic algorithms that overcome this challenge by weighting words with their relative frequency of occurrence. They support both long search queries and rank results based on a relevance score. BM25 is the default search algorithm in modern databases such as Elastic Search [Gormley and Tong(2015)]. To our knowledge, BM25 has not been considered for EIF search in the literature, and we include it as a baseline algorithm in our evaluation. We do not include exact match as a baseline as it is challenging to identify the keyword to use for the query, and use of the entire product description leads to spurious matches with close to 0% precision.

Neural search algorithms improve on exact string match by learning semantics such as synonyms and contextual relationships [Mitra et al.(2018)]. However, they require a large dataset of queries with matched results for training the model [Huang et al.(2013), Hu et al.(2014)]. No such dataset exists for searching EIFs, and therefore, we focus on zero-shot methods that do not require training data. Neural language models such as BERT [Kenton and Toutanova(2019)] and GPT [Radford et al.(2018)] reduce the reliance on training data by using a self-supervised objective such as predicting the next word in a sentence. Use of the Transformer architecture enables training on web-scale datasets [Vaswani et al.(2017)]. Sentence transformers, called SBERT, build on these works, and have emerged as a strong zero-shot algorithm for semantic text matching [Reimers and Gurevych(2019), Xiong et al.(2022)]. They are trained on web text to create a vector representation (a.k.a embedding) of an input sentence. We can identify if two sentences are similar by measuring the distance between their embeddings. Using SBERT, we can find the closest EIF that matches a query text by identifying the EIF text embedding that is closest to the query embedding. balaji2023caml use SBERT to find EIFs for EIO-LCA, while we focus on finding EIFs for Process-LCA. Unlike Process-LCA EIF databases, EIO-LCA EIF databases are complete by definition as the EIF corresponds to industry sectors defined by the national governments. Therefore, balaji2023caml do not address the problem of identifying when no EIF matches exist in the database. We include SBERT as a baseline in our evaluation.

One of the challenges with a neural search solution is that they are not designed to identify when an appropriate match does not exist [Sun et al.(2021), Dhamija et al.(2018)]. A simple method is to threshold the distance between embeddings beyond which an EIF is not an appropriate

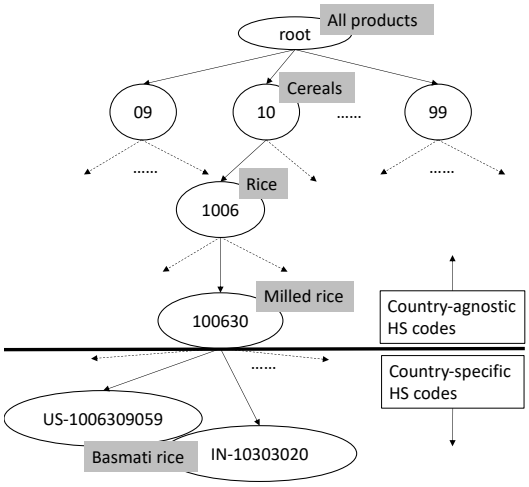


Figure 3: An example of HS codes, along with its hierarchical structure.

match [Zeng et al.(2021)]. However, as we show in our evaluation, a threshold based solution leads to incorrect predictions due to distance miscalibration. The SBERT model can be fine-tuned on a search dataset so that the distances are calibrated [Sun et al.(2021)]. Another method is to train a separate model that determines if the predicted match is correct or incorrect [Hu et al.(2019)]. But these solutions require labeled training data, and do not meet our zero-shot objective. We propose a novel solution, where we first classify the EIFs and the query text to an industry sector. If the industry sector of the query does not match with any entry in the EIF database, we can predict that there is no suitable match for this query.

We use the HS codes for our industry sector classification, which is hierarchically organized from 2 digits to up to 12 digits, where the level of hierarchy is indicated with a 2-digit increment [Chaplin(1987)]. Figure 3 shows an example. A major advantage of HS codes is that they are already used for import/export taxation worldwide, and therefore, datasets that map products to HS codes are readily available. We use up to 6 digits of the HS code (HS6), which translates to $\sim$5.4K unique codes. We leverage an existing dataset of products mapped to HS6 codes to learn a supervised classifier that can predict the HS code for any text input.

Prior works have proposed ML approaches to automate HS code classification as it is used to determine customs tax rate [Ding et al.(2015), Du et al.(2021), Chen et al.(2021)]. chen2021neural consider a neural machine translation approach with a hierarchical loss function, and obtain an accuracy of 85% on a dataset of $\sim$1M records. lee2021classification use a supervised, hierarchical sentence retrieval approach to classify 129K electrical products with an accuracy of 89.6%. du2021hscodenet use two parallel neural networks in a Siamese architecture for classifying HS codes, one network uses a hierarchical sequence of LSTM modules on word vectors of text input, and another uses graph attention network based on word co-occurrence. On a dataset of $\sim$400K samples, they obtain an average accuracy of 85%. We treat HS code classification problem as an example of extreme label classification [Bhatia et al.(2015), Bhatia et al.(2016)]. We use the PECOS algorithm, which leverages the label hierarchy and semantic similarity of labels to improve classification performance [Yu et al.(2022)]. Flamingo is agnostic to the method used to classify HS codes, we use PECOS in our experiments as it achieves state-of-the-art in multiple extreme label classification benchmarks. On our dataset of 746K products, we obtain a classification accuracy of 82% which is commensurate to the results reported in prior works. We also use SBERT to identify the best matching HS codes for a given text description with a zero-shot approach. To our knowledge, zero-shot methods to classify HS codes have not been studied in the literature.

Table 2: Summary of datasets used in the paper

Type	Size	# Match	# No Match	# No clear label	# Used
Product to EIF (small)	100	22	68	10	90
Product to EIF (large)	967	277	387	303	664
Food ingredient to EIF	272	87	177	8	264
Product to HS code	746K	No filter			746K
HS code descriptions	6709	Filter for HS6 codes			5388
Ecoinvent EIFs	19128	Filter for reference products			2770

B Dataset

In this section, we describe the dataset we use to evaluate the performance of Flamingo in matching a query text to an EIF. Table 2 summarizes the datasets used in the paper.

The dataset—which we refer to as D_Q—is a set of 967 products sold by an e-commerce retailer. It includes a variety of products such as ice makers, electric fans, toasters, adhesives, etc. Given a product, we concatenate its title, description, and additional attributes into a single string as input to the classifier. For each product, the dataset includes the ground truth HS6 code as well as a matching EIF from the Ecoinvent dataset [Wernet et al.(2016)], if a match exists. We use the 2017 version of HS codes, and EIFs from the Ecoinvent v3.7. Human annotations are used to validate the EIF matches in the dataset: the annotation process is described in Section C. All the product descriptions are in English—while the text-similarity models we use can generalize to any language, we restrict the language because our annotation team only understands English fluently. We annotate a subset of 100 products by LCA experts to evaluate the quality of annotations by non-experts. To evaluate generalization beyond products, we introduce another dataset of 272 food ingredients that are matched to EIFs with annotations.

Ecoinvent EIFs include metadata such as impact factor name, reference product, units, data quality, valid years, geographic specificity, and industry classification. We use the attribute 'reference product' as the basis for matching EIFs because it provides a non-technical but precise description of the EIF (e.g., wheat, yogurt). Once the EIFs with the reference product are identified, it is easy to add rules to increase specificity by additional attributes such as location. Ecoinvent contains 19K EIFs, but only 3.2K unique reference products. The dataset includes EIFs that are not related to carbon footprint of consumer products such as those related to construction, operation of equipment or transportation of goods. We filter these out to get 2.7K unique EIFs.

We refer to an individual HS code as $h_i^\delta \in \mathcal{H}$, where δ refers to the number of digits in the code. We obtain the text description of HS6 codes from `https://unstats.un.org/`. We use a dataset that maps 746K products to their corresponding HS6 codes for learning a supervised model to predict HS6 code given text as input: we refer to this dataset as D_H. The dataset consists of a variety of products (e.g., shoes, watch, coat), and comprises 2.5K unique HS6 codes. Note that this is a subset of the full set of HS6 codes, $|\mathcal{H}| = 5400$. The distribution of the number of products per HS6 code is skewed, with 10% of HS6 codes accounting for 86% of the products. Despite the skew and reduced cardinality, the HS6 codes in D_H contain a representative set of products, and we expect a large overlap with a generic product query.

C Annotation

In this section, we describe the process of collecting ground truth data of product to EIF mappings via manual annotations. Identifying the best match from thousands of EIFs is a challenging and time-consuming task. To reduce burden, we rank the top-5 EIFs using FlamingoZero, and ask annotators if any of these are a match. In case there are <5 EIFs after filtering based on HS codes, we add the top ranked EIFs from the full set until we have a total of 5 options. We also include the option 'No match' and 'Not sure' to capture both missing EIFs in the database and uncertainty in annotation. We acknowledge that such an annotation system does not measure recall accurately as there could be EIFs in the database which may be an appropriate match, but are not captured in the top ranked EIFs by Flamingo. Nevertheless, it does capture the precision of the algorithm

Table 3: Performance of EIF matching using different industry sector codes

Industry Code	Distance threshold	Accuracy (%)	Macro F1	Weighted F1
HS4	0.0	75.0	14.3	68.0
HS4	0.5	70.2	12.2	62.1
ISIC	0.0	63.4	4.4	62.4
ISIC	0.5	68.7	5.1	62.7
CPC	0.0	52.7	5.5	45.7
CPC	0.5	59.9	6.6	48.9
CPC to HS2	0.0	50.6	11.0	42.4
CPC to HS2	0.5	57.8	10.3	45.4

accurately. The recall of the ranked list can be further improved by using a mix of different algorithms such BM25 [Trotman et al.(2014)], universal sentence encoder [Cer et al.(2018)], or with use of late interaction models such as ColBERT [Khattab and Zaharia(2020)] that re-rank a candidate list. Our contribution and focus here is on improving the precision of identifying if an EIF is an appropriate match after it has been ranked by a search method. We defer measuring and improving recall of EIF ranking to future work, we expect our results to improve further if recall improves. All our baseline methods, except BM25, rely on the same SBERT model for EIF ranking. Therefore, the results still provide a controlled experiment to compare methods.

In our instructions to annotators, we provide multiple examples of product to EIF mapping, including those with 'no match'. More than 80% of randomly sampled products do not have an appropriate EIF in the database. To balance this skew, our dataset for annotation includes all the products for which we find an EIF, and randomly sample an equivalent number of products for which there is no match. We use the top-5 ranked EIFs without an HS code filter for these products for a consistent annotation experience.

We use three independent annotations per product and use the majority vote to reduce labeling noise. Our annotators are experienced in labeling tasks. Each annotation took 30 seconds on average. Despite our efforts to reduce annotation burden, our annotators provided feedback that the task was challenging. They had to constantly look up technical terms, such as 'kenaf' and 'mercerizing': some tasks took as much as 5 minutes. Of the 967 products annotated, 28% had unanimous agreement, 44% had a valid majority vote, and the remaining had split votes with no clear majority. Another 5.6% of the products had 'Not sure' as the majority vote. We only consider the 664 products which have a valid majority, and use the majority voted label as the ground truth for our experiments. The final product dataset has 58% 'no match', and 48 unique EIFs with a long-tail distribution.

Krippendorf's Alpha is a measure of inter-annotator agreement for classification tasks, with 0 and 1 representing perfect disagreement and agreement respectively [Krippendorff([n. d.])]. The Alpha for our annotations is 0.28, which is similar to values reported for long-tailed classification tasks in the literature [Jalalzai et al.(2020)].

To further validate our dataset, we asked two LCA experts to annotate a subset of 90 products. Picking one of the experts arbitrarily as the ground truth, we measure the precision of both our expert and non-expert annotations. We find that non-experts have a precision of 78.6% on average, and are comparable to expert precision of 76.1%. With majority vote, the non-expert agreement with expert annotations increase to 85.9%. Therefore, we consider our non-expert annotations to be of sufficient quality to be considered as ground truth. We present the full results in Table **??**.

D Additional Results

D.1 Alternatives to HS code prediction

There are multiple industry classification systems that can be an alternative to HS codes. We look at two choices: International Standard Industrial Classification (ISIC) [Nations(1969)] and Central Product Classification (CPC) [Pariag(2009)]. They are pertinent choices as they are already included as part of the EIF metadata in the Ecoinvent database [Wernet et al.(2016)]. However, we do not have the ISIC or CPC codes of the products in our dataset, so we use the SBERT model to predict the best

Table 4: Performance of EIF matching using different industry sector codes

Industry Code	Distance threshold	Precision@1 (%)
HS4	0.0	75.0
HS4	0.5	70.2
ISIC	0.0	63.4
ISIC	0.5	68.7
CPC	0.0	52.7
CPC	0.5	59.9
CPC to HS2	0.0	50.6
CPC to HS2	0.5	57.8

Product Description	Product HS code	Best EIF by Semantic Matching	EIF HS code	Label
10k Gold Imported Crystal March Birthstone Ring	Jewelry	Precious metal for jewelry	Jewelry	Precious metal for jewelry
Dark Roast Whole Bean Coffee	Coffee; roasted	Coffee, green bean	Coffee; not roasted	Coffee, green bean
Car tire brush	Brooms, brushes, mops	Vacuum cleaner	Vacuum cleaners	No match
Women's Belice Ballet Flat 100% synthetic	Footwear	Textile, non-woven polyester	Fabrics, woven	No match

Figure 4: Examples of errors by Flamingo. Most errors can be categorized into the five types shown.

ISIC and CPC codes for each product using text descriptions. The United Nations also publishes correspondence tables between CPC and HS codes, we include this as an additional option to identify HS codes for EIFs instead of predicting with the SBERT model. Table 4 shows the performance metrics on the 664 product dataset.

EIF matching based on HS codes outperform the rest of the options across all metrics. Upon manual inspection, we find that HS6 codes are more granular with a deeper hierarchy compared to ISIC and CPC. The assignment of ISIC and CPC codes in the EIF database is also subjective, e.g., some mappings correspond to 3 digit CPC codes while others correspond to 5 digits. There are also errors in mapping between CPC and HS codes in the correspondence tables, and directly predicting the HS codes reduces the errors compared to mapping correspondence data across two sources.

D.2 Analysis of errors

We randomly sampled 50 data points incorrectly predicted by the FlamingoZeroHS4 from Table 1, and manually analyzed them to understand the reasons behind the erroneous outputs. Figure 4 shows a few examples of the errors, categorized into five types. A majority of the errors (40%) corresponded to electronic cables such as those for charging or connecting components (row 4 in the figure). There is an EIF in Ecoinvent, called ‘cable, unspecified’, which captures all of these different types of cables. We find that SBERT is poor at matching text such as these, where it needs to understand the set relationship across closely related EIFs or HS codes. Some HS codes contain descriptions of similar nature, e.g., ‘not elsewhere classified’, and are a common source of errors in matching. Methods which capture such set relationships or exploit the hierarchy in a different manner could overcome this issue.

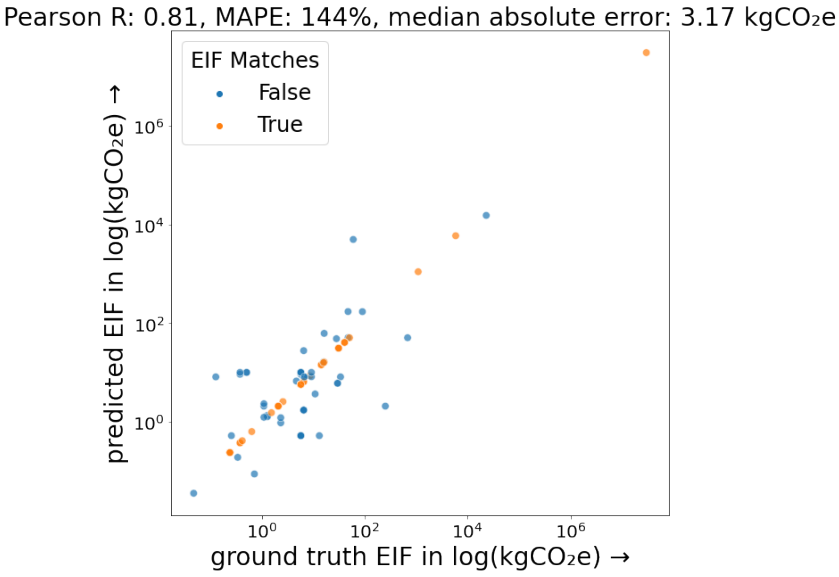


Figure 5: Impact of EIF misprediction error on downstream carbon footprint estimate. In most cases, the error in prediction does not lead to a large error in carbon emissions, however, there are a few outliers that cause large deviations.

In some cases (22%, row 2 in figure), the error is due to mismatch between HS codes even when the EIF predicted is the same as the selected ground truth. For example, a toner module is mapped to the HS code 'ink for printing' by the SBERT model, whereas the product maps it to a related HS code 'printing machinery'. In this case, the two related HS codes are not close to each other in the hierarchy, so using a 2-digit or 4-digit version of the code did not remove the error. A related source of error (8%, row 5 in figure) is when EIF predicted is not a match even when the HS codes match. In this case, the HS code category is too broad and does not differentiate between unrelated EIFs. Such errors point to the limitation of using HS codes as an imperfect classification system. A data-driven knowledge base can potentially address such issues by capturing relationships across concepts in a multi-dimensional manner.

About 20% of the errors were due to semantic matching errors (row 1 of figure), and point to opportunities for improving on SBERT models. There were a few errors due to human mislabeling as well (10%, row 3 of figure), which could be reduced by improving the annotation procedure. Finally, we found an intriguing but rare error not shown in Figure 4. In one case, the product was a package of keyboard and mouse. The humans annotated the ground truth EIF as keyboard, whereas Flamingo predicts the EIF as mouse. Such errors can be addressed if we break down composite products to their individual items.

D.3 Impact of prediction error on carbon footprint

We analyze the impact of an EIF misprediction on downstream use cases using the corresponding greenhouse gas emission values in units of kilograms of carbon dioxide equivalent ($kgCO_2e$). We used the 'reference product name' metadata in the Ecoinvent dataset to represent an EIF, but there could be multiple entries for a single reference product based on variations such as region of manufacture or system boundary. For this analysis, we compute the average of these individual values following similar practice from prior works [Meinrenken et al.(2012), Clark et al.(2022)]. We use the metrics Pearson's correlation (Pearson R), mean absolute percentage error (MAPE), and median absolute deviation (MAD) to characterize the errors.

We use the predictions by FlamingoZeroHS4 variant of the algorithm for this evaluation. Our analysis is limited by the known EIFs in the dataset. Of the 664 products in our dataset, we only have ground truth EIF for 277 products as the rest do not have a match. Of these 227 products, 130 are predicted to have a match by FlamingoZeroHS4. 125 of the 130 products are predicted correctly, and yield low

Table 5: Evaluation results for Flamingo with a dataset of 264 food ingredients. Ground truth is obtained through annotation by a non-expert worker.

Method	Distance threshold	Precision@1 (%)			Macro Precision	Weighted Precision
		Overall	No Match	Match		
BM25	0.1	23.1	19.8	30.0	13.4	56.9
SBERT	0.5	34.8	14.1	**77.0**	26.1	86.2
FlamingoZeroHS4	0.0	70.0	75.7	57.4	41.0	76.1

error – Pearson R: 0.95, MAPE: 18%, MAD: 0 $kgCO_2e$. This statistic indicates that misprediction of 'no match' are the major source of errors.

To further understand the impact of errors due to semantic matching, we analyze the error after removing the HS code based filter. Predictions for 172 of 277 products are correct, and the corresponding error metrics are – Pearson R: 0.81, MAPE: 144%, MAD: 3.9 $kgCO_2e$. Figure 5 compares the carbon emission values of the ground truth EIF to that of the predicted ones. We use a log/log plot to cover the wide range of values, trends look similar in linear scale. Overall, there is a high correlation in predicted and ground truth values, the errors are small in absolute terms but larger in relative terms. There are a few anomalous predictions that cause an error in multiple orders of magnitude, four points have >1000% error and an additional 5 points have >100% error. Some examples of errors include predicting a microwave as HVAC unit (8118%), incorrect battery chemistry (324%), and incorrect paper type (112%).

D.4 Generalization to food ingredients

LCA experts often match EIFs to materials, manufacturing processes, and not just products. We evaluate if Flamingo generalizes to use cases beyond our product dataset using a separate dataset of food ingredients. Unlike the product descriptions, the ingredients consist of only a few words, e.g. 'organic diced tomatoes', 'purified water', 'organic licorice'. Our dataset consists of 272 ingredients, and we follow the same steps as Section C to annotate the ground truth. Due to the labor-intensive nature of annotations, we only use one worker for this micro-benchmark and do not have a consensus based label. Unlike generic products, basic materials such as food ingredients have a higher probability of finding a match in the EIF dataset. From the annotations, we find 87 ingredients that match to EIFs and 177 that do not have a match.

Table 5 summarizes the results. We report the results of the FlamingoZeroHS4 algorithm, and compare it with SBERT and BM25 baselines. In this case, the HS code for the ingredient is provided as part of the dataset, and we rely on Flamingo to predict the HS codes for both EIFs and ingredients. The results follow the same trends as those from product dataset experiments, although the improvement compared to baselines is even larger in this case (35.2%). As an example, the best semantic match for 'pineapple juice' is 'pineapple'. The HS codes for these two strings are different, and Flamingo correctly predicts 'No match'.

From an experience point of view, LCA experts find Flamingo to be useful and report significant time savings. A skilled practitioner used Flamingo to complete LCA of 15K food products in just 15 minutes, a task that would have previously taken 40 hours of manually mapping EIFs.