A GREENABILITY EVALUATION SHEET FOR AI-BASED SYSTEMS

JOEL CASTAÑO FERNÁNDEZ

Thesis supervisor: SILVERIO JUAN MARTÍNEZ FERNÁNDEZ (Department of Service and Information System Engineering)

Thesis co-supervisor: XAVIER FRANCH GUTIÉRREZ (Department of Service and Information System Engineering)

Degree: Bachelor's Degree in Data Science and Engineering

Thesis report

Facultat d'informàtica de Barcelona (FIB)
Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona (ETSETB)
Facultat de Matemàtiques i Estadística (FME)

Universitat Politècnica de Catalunya (UPC) - BarcelonaTech

I would like to express my deepest gratitude to my supervisors, Silverio-Martínez Fernández and Xavier Franch, for their support, insightful guidance, and patience throughout this project. Their knowledge and experience have been key for this thesis.

Moreover, I extend my sincere appreciation to my advisor, Justus Bogner from the University of Stuttgart. His feedback was invaluable to this project. His reviews and in-depth discussions helped shape this research, enhancing its rigour and relevance.

Abstract

Background: The rise of machine learning (ML) systems has increased their environmental impact due to the enhanced capabilities and larger model sizes. However, information about how the carbon footprint of ML models is measured, reported, and evaluated remains scarce and scattered.

Aims: This project, based on an analysis of 1,417 ML models and associated datasets on Hugging Face, the most popular repository for pretrained ML models, aims to provide an integrated solution for understanding, reporting, and optimizing the carbon efficiency of ML models. Moreover, we implement a web-based application that generates energy efficiency labels for ML models and visualizes their carbon emissions.

Method: We conduct a repository mining study on the Hugging Face Hub API on carbon emissions and answer two research questions: (1) how do ML model creators measure and report carbon emissions on Hugging Face Hub?, and (2) what aspects impact the carbon emissions of training ML models? Furthermore, we develop an interactive tool that leverages these findings to provide users with a tangible method for understanding and improving their ML models' carbon efficiency.

Results: The study uncovers a slight decrease proportion of carbon emissions-reporting models in Hugging Face, a decrease in reported carbon footprint on Hugging Face over the past two years, and a continued dominance of NLP as the main application domain with an increasing share of computer vision models. It also identifies correlations between carbon emissions and various attributes, such as model size, dataset size, and ML application domains. Leveraging these results, the developed tool can generate an energy label for a given ML model, representing its carbon efficiency from a multi-dimensional perspective. It also enables users to visualize carbon emissions across various ML models.

Conclusions: With less than 1% of models on Hugging Face currently reporting carbon emissions, the project underscores the need for improved energy reporting practices and the promotion of carbon-efficient model development within the Hugging Face community. To address this, we offer a web-based tool that produces energy efficiency labels for ML models, a contribution that encourages transparency and sustainable model development within the ML community. It enables the creation of the energy labels, while also providing valuable visualizations of carbon emissions data. This integrated solution serves as an important step towards more environmentally sustainable AI practices.

Keywords

repository mining, software measurement, sustainable software, green AI, carbon-aware ML, carbon-efficient ML

Contents

1. Introduction

The project is carried out at the GESSI (Software and Service Engineering) research group at UPC. It starts as a proposal from the GAISSA project “Towards green AI-based software systems: an architecture-centric approach” [1]. The purpose of this project is to define and implement a greenability evaluation sheet to classify AI-based systems in levels according to their energy efficiency.

In this section we will motivate the purpose of the project and contextualize it in the margin of software sustainability and green AI. Moreover, we will address the current state of the art for green AI and the creation of the greenability evaluation sheet for AI-based systems.

1.1 Motivation

1.1.1 Sustainability and Green Software

Sustainability awareness has gained global importance over the last years, driven by various initiatives that emphasize the need for reducing humanity’s overall carbon footprint [2].

This is particularly true in the field of information and communication technologies (ICTs), which have emerged as potential contributors to sustainability. However, ICTs can also have a negative impact on the environment; while the technology can be used to make many processes more efficient optimizing pipelines and reducing the carbon footprint, it can be responsible for environmental damage with an increase in energy consumption. Andrae and Edler’s projections show the global electricity consumption of ICT increasing from 1,500 TWh (8% share) in 2010 to up to 30,700 TWh (51%) in 2030 [3], highlighting the need for sustainable practices in IT research to minimize the negative impact of ICT on the environment [4].

Moreover, from a corporate standpoint, sustainability has grown to be an essential factor for company values and brand perception. Profit-oriented businesses may initally seem incompatible with eco-friendly perspectives; however, companies neglecting sustainability risk criticism, losing market share and overall revenue. With growing consumer sustainability concerns [5], organizations are incresingly adopting Corporate Social Responsibility (CSR) objectives objectives based on the ISO 26000 standard [6] to align with these values.

1.1.2 Green AI

As a subset of ICTs, systems incorporating machine learning (ML), i.e., ML-based systems [7], are especially popular lately. Nevertheless, this popularity in ML capabilities has come with an increase in model size and training time [8]. This raises concerns about the environmental impact of the growing field of ML, prompting calls for the development of inclusive and environmentally friendly ML-based systems. This aligns with global efforts to reduce carbon emissions in all sectors of society.

While several recent studies shed light on how energy efficiency can be increased during model training (e.g., during hyperparameter tuning [9]), little is known about the actual emissions of most published ML models, or even if creators of ML models consider and report energy-related aspects when publishing their models. Potential sources for this information are public repositories of pretrained models. One well-known example of such repositories is the Hugging Face Hub [10], which has emerged as the most popular platform for pretrained language models and associated datasets [11], and also for computer vision or audio processing models. In spite of this widespread use, there is currently a lack of understanding how carbon emissions are reported in this repository.

In this bachelor thesis, our primary goal is to fill this knowledge gap and contribute to sustainable AI practices by developing a *greenability* sheet—an energy label for machine learning models. This label is designed to provide ML practitioners, researchers and corporate entities with an intuitive and informative reference guide, aiding them in steering their model development towards more sustainable practices and aligning with their CSR objectives. To construct this greenability sheet, we leverage insights gleaned from an in-depth analysis of carbon emissions of ML models during training, particularly those housed on Hugging Face.

We have adopted repository mining [12] as our method of choice, as it enables us to quantitatively analyze the expansive Hugging Face dataset of models and their carbon emissions. Hugging Face stands as an ideal data source due to its extensive collection of pre-trained models, widespread use among ML practitioners, and available metadata. The Hugging Face Hub API and HfApi class further facilitate our systematic extraction of repository information.

By analyzing the carbon emissions of ML models on Hugging Face and using these findings to construct our greenability sheet in form of a web-app, we aim to offer valuable insights and recommendations to ML practitioners and researchers. This approach not only aids them in reporting and optimizing the carbon efficiency of their models but also contributes to creating a more environmentally sustainable AI landscape.

Therefore, the two main contributions of this project are as follows:

- **Study of Machine Learning Model Carbon Efficiency**: This project conducts an in-depth exploration of carbon consumption in AI, focusing on ML models available on the Hugging Face Hub. The research presents insights into carbon consumption trends, the lack of standardized reporting, and factors influencing carbon usage in AI models. Our study also introduces a systematic approach for classifying models based on their carbon efficiency, providing a standard for reporting and comparing the energy impact of AI systems. This classification takes into account multiple parameters including CO2 emissions, model size, dataset size, and reusability. The comprehensive dataset we have curated and the detailed analysis performed serve as valuable resources for future research in this area.
- **Web Application for Energy Efficiency Evaluation**: Complementing the insights from the study, the project also delivers a practical tool for the AI community. The web application provides an intuitive and user-friendly platform that allows practitioners to evaluate the energy efficiency of their models and gain an understanding of their environmental impact. It allows users to generate energy labels for new models based on user-provided parameters or view labels for existing models. Moreover, the application offers visualizations that enable an in-depth exploration of carbon emissions data for ML models, aiding users in identifying potential areas for energy efficiency improvement. We integrate with Google Sheets via Google Cloud Storage API, ensuring easy update and access of the models' dataset, allowing the app to be scalable.

Data availability statement: The datasets, code, and detailed documentation, are available on our GitHub's repository [1]. The repository enables researchers and ML practitioners to reproduce and extend the results of our study. It contains Jupyter notebooks with the data extraction, preprocessing, and analysis scripts, along with the raw, processed, and manually curated datasets used for the analysis and the code of the web-app. We have

[1] GitHub's repository: `https://github.com/GAISSA-UPC/ML-EnergyLabel`

provided explanations on how to navigate the data source, how to use it, and how the provided data, code, and tools are used in the steps of the methodology described in the study.

1.2 Background and Related Work

In this section, we explain the most important concepts to understand this study and present related work in the area.

1.2.1 Sustainability and Energy Consumption of Software

In general, sustainability represents the capacity of something to last a long time [2], but is also associated with the resources used [2] for a particular activity. Regarding software sustainability, the Karlskrona Manifesto [13] describes five dimensions: technical, economic, social, individual, and environmental. Our study exclusively targets the latter dimension. The movements to increase energy efficiency and reduce the carbon footprint of software have been called *Green IT* and *Green Software* [14], and the complementary concept for artificial intelligence is *Green AI* [8] [15]. Within Green AI, the training of ML models is especially important because its long duration and extensive use of computing hardware can make it especially resource-intensive [16].

One of the most important aspects of environmental software sustainability is *energy consumption*, which is usually measured in *joule* (J) or *kilowatt-hours* (kWh) [17]. In this sense, we may want to know how much energy a certain operation consumed, e.g., the training of an ML model. Most tools usually measure and report the average energy or power consumption for a certain timeframe [18].

While measuring energy consumption is useful to judge energy efficiency, the extent of harmful greenhouse gases like carbon dioxide (CO_2) depends on how the electrical energy was produced. Therefore, a more accurate concept to measure climate impact is the *carbon dioxide equivalent* (CO_2e), colloquially known as the *carbon footprint* [17]. It is reported in mass units, e.g., $kgCO_2e$ or gCO_2e, and is the suggested metric in the greenhouse gases standard ISO 14064 [19]. A model trained in an Icelandic data center with geothermally produced electricity may still consume substantial energy, but can have a marginal carbon footprint. While many more tools exist to measure software energy consumption, libraries like `CodeCarbon` [20], Lacoste et al.'s ML workload calculator [21], `Carbontracker` [22] or `Eco2AI` [23] can estimate the carbon footprint.

1.2.2 Hugging Face Hub

Since training complex ML models requires considerable expertise and resources, it is desirable to reuse pretrained models. The company Hugging Face, Inc. provides a community platform to achieve this. Originally founded in 2016 as an NLP company, Hugging Face gained popularity for open-sourcing their NLP models [11] and for creating an easy-to-use Python library for NLP transformers [24]. Today, one of their most important offers is the Hugging Face Hub, which is an open platform to publish ML models and datasets. For documentation and reproducibility, the Hub adopted Mitchell et al.'s *Model Cards* idea [25]. Each published model can provide a `README.md` plus additional metadata, e.g., performance metrics or tags. This also includes a carbon footprint attribute (*co2_eq_emissions*), with guidelines on how to report it [26]. Lastly, a convenient option to train and publish models is to use the *AutoTrain* feature [27], which is the AutoML infrastructure of Hugging Face. Models published via AutoTrain automatically include their training carbon footprint.

1.2.3 Streamlit for Dashboard Development

To construct the interactive dashboard for generating the energy label for ML models, we will be using Streamlit [28]. This free and open-source framework allows for rapid building and sharing of data science and machine learning web applications. Streamlit's Python-based library is specifically designed for machine learning engineers, providing an easy-to-use environment for developing web apps. It offers compatibility with numerous Python libraries, making it a valuable tool for our project. With Streamlit, we will build an intuitive dashboard where users can navigate through various visualizations on Hugging Face carbon emissions and add new models, generating their energy label.

1.2.4 Related Work

Since Schwartz et al.'s seminal paper on Red vs. Green AI in 2020 [8], research in this area has increased steadily. A recent literature review on Green AI by Verdecchia et al. [29] identified 98 primary studies, with the most prevalent topics being energy monitoring of ML models throughout their lifecycle (28 papers), energy efficiency of hyperparameter tuning (18), and energy footprint benchmarking of different ML models (17). From a more foundational perspective, García-Martín et al. [30] synthesized guidelines and tools to estimate the energy consumption of ML models. Similarly, Patterson et al. [31] proposed four general practices to reduce the carbon footprint of ML training, namely the 4Ms: *model*: selecting efficient model architectures, e.g., sparse models; *machine*: using processors optimized for ML training, e.g., tensor processing units (TPUs); *mechanization*: using optimized cloud data centers; and *map*: picking a data center location with clean energy.

Several studies also conducted energy consumption benchmarks of various model characteristics or training methods, especially for deep learning. Yarally et al. [32] compared different hyperparameter optimization methods and layer types, while Xu et al. [33] analyzed the impact of network architectures, training location, and measurement tools on both energy consumption and carbon footprint. Verdecchia et al. [34] analyzed how exclusively modifying the dataset can reduce energy consumption, and Brownlee et al. [35] conducted one of the few studies that analyzed the tradeoff between accuracy and energy consumption during hyperparameter optimization not only for model training, but also for the inference stage. Regarding ML frameworks, Gergiou et al. [15] performed an in-depth comparison of the energy consumption between TensorFlow and PyTorch. With respect to ML domains (NLP, vision, etc.) and its carbon emissions, Bannour et al. [36] evaluated the carbon footprint of NLP methods and its measuring tools. Lastly, in the context of mobile applications containing neural networks, Creus et al. [37] published a registered report at ESEM'21 about identifying design decisions with impact on energy consumption.

Closely related to our endeavor is an interview study by Jiang et al. [38]: they synthesized practices and challenges for the reuse of pretrained models in the Hugging Face ecosystem, and complemented their interviews with a security risk analysis enriched with data queried from the Hugging Face Hub. However, energy consumption was not part of their study, and also not mentioned by their interviewees.

Several studies have also been conducted on the idea of energy labels:

- The EU has standardized energy labels for various products, such as household washing machines and washer-dryers [39], to provide consumers with valuable information that enables them to make an informed choice and eventually increase the market for more energy-efficient products.
- The EcoSoft project [40] proposes an eco-label for software sustainability. Their work is centered around

corporate sustainability and how companies can lessen the impact of new technologies on the environment. They propose an ecolabel for software sustainability, based on a set of relevant criteria.

- Raphael et al. [41] proposes a set of metrics to assess, compare, and report the efficiency and performance trade-off of different ML methods and models. They introduce a concept for visually communicating efficiency information to the public, inspired by the EU's energy label system.

Nonetheless, to the best of our knowledge, no study has so far analyzed the carbon footprint of training ML models from a large-scale repository mining perspective while also providing an interactive energy label that can be filled and generated for inputted models. Our study aims to fill this gap by reporting insights into the current state of carbon emission reporting in one of the largest ML communities and providing a convenient tool for generating the greenability sheet.

2. Methodology

In this section, we first define the study objective and the research questions that will guide the investigation. Next, we explain how to obtain the dataset for the analysis of the research questions. Figure 1 illustrates the outline of the data collection procedure along with the design of our investigation.

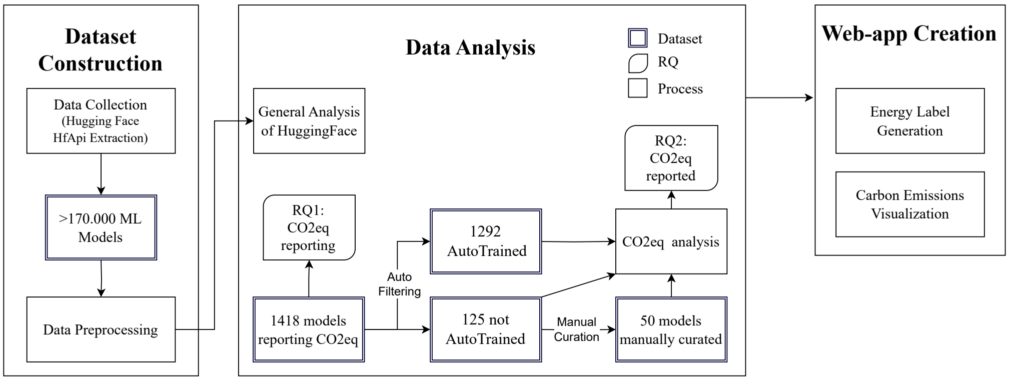


Figure 1: Study Design Diagram

2.1 Study Objective and Research Questions

We formulate our research goal according to the Goal Question Metric (GQM) guidelines [42] as follows: Analyze *ML models and corresponding datasets on the Hugging Face Hub* with the purpose of *developing* with respect to *a greenability sheet and energy label that quantifies carbon emissions during training* from the point of view of *ML practitioners, ML model developers, and end-users* in the context of *sustainable AI and ML applications*.

This goal breaks into three main Research Questions (RQ), which respectively have several sub-research questions. Firstly, we seek to understand the practices surrounding the measurement and reporting of training carbon emissions for ML models on Hugging Face:

> **RQ1**. *How do ML model creators measure and report training carbon emissions on Hugging Face Hub?*

- RQ1.1: How has the reporting of carbon emissions evolved over the years?
- RQ1.2: How has the reported carbon emissions evolved over the years?
- RQ1.3: What are the main characteristics of the models reporting carbon emissions?
- RQ1.4: How can we classify Hugging Face models based on their carbon emission reporting?

Subsequently, we conduct an in-depth analysis of various factors related to the carbon emissions of training ML models. We aim to identify and comprehend the diverse elements that influence the carbon emissions during the training process:

> **RQ2**. *What aspects impact the carbon emissions of training ML models on Hugging Face Hub?*

- RQ2.1: How are carbon emissions and model performance related?
- RQ2.2: How are carbon emissions related to model and dataset size?
- RQ2.3: What is the difference in carbon emissions between fine-tuned and pretrained tasks?
- RQ2.4: How do ML application domains affect carbon emissions?

Leveraging insights from RQ1 and RQ2, our next goal under RQ3 is to create an energy efficiency classification system for ML models. This system, visually conveyed via a greenability sheet, will be part of an interactive web application that generates energy labels and provides carbon emissions data visualization.

> **RQ3**. *How can we design and implement an energy efficiency classification system for ML models, represented through a greenability sheet and interactive web application?*

- RQ3.1: How can we classify models based on their carbon efficiency?
- RQ3.2: How can we design and implement a greenability sheet to accurately and effectively represent the energy efficiency of different ML models?
- RQ3.3: What is the data scientists perception on the energy label generation?

2.2 Dataset Construction

In our repository mining study [12], we start by gathering and preprocessing the data to answer the above RQs.

2.2.1 Data Collection

We developed a data collection pipeline to collect data from the Hugging Face models and their associated Model Cards. The pipeline uses the Hugging Face Hub API through the HfApi class, a Python wrapper for the API, which provides a *list_models* function. This function allows us to retrieve all the information of every model that has been uploaded to Hugging Face and to apply subset filtering. In this study, we collected data up to March 2023 (included). This the data extraction pipeline is rather slow due to the slow API calls from the Hugging Face's HfApi, taking more than 30 hours. To achieve a faster extraction, we parallelize the the API calls reducing the time spent to less than 8 hours. The further details of the data collection pipeline are available at the repository, allowing updating the data in the future if desired.

Once we have the models list retrieved, the pipeline automates the extraction of information, including the use of regular expressions to locate and collect evaluation metrics such as accuracy, F1, Rouge1, and so on. Each row of the dataframe generated by the pipeline is a model alongside many attributes and its CO_2e emissions metric if it is reported. Initially, we collect every model of Hugging Face regardless of whether carbon emission was reported or not. Afterwards, we consider the dataset split based on the models that reported CO_2e. The main attributes that we collect are the following:

- *co2_eq_emissions*: the reported training CO_2e emissions.
- *datasets_size*: the total size of the datasets used. For more than one dataset, the dataset sizes are summed up.

- *training_type*: if pretraining or fine-tuning training.
- *geographical_location*: the location of the training.
- *hardware_used*: the hardware used for the training.
- *accuracy*: the accuracy evaluation, if reported.
- *f1*: the F1 evaluation, if reported.
- *rouge1*: the Rouge1 evaluation, if reported.
- *rougeL*: the RougeL evaluation, if reported.
- *size*: size of the final model trained.
- *auto*: if the model is AutoTrained.
- *downloads*: number of downloads of the model.
- *likes*: number of likes of the model.
- *library_name*: library used by the model.
- *modelcard_text*: text of the model card of the model.
- *created_at*: date of the model creation on Hugging Face
- *tags*: the tags reported by the model (e.g., PyTorch, AutoTrain, BERT, ...)

Despite its ability to handle various data formats and filter out unnecessary information, the pipeline cannot fully address the issues of missing values and varying reporting formats due to the lack of a strict standard for information reporting on Hugging Face models.

For attributes like *datasets_size*, where manual data collection is required, we have added the information manually to the dataset. We acknowledge that this manual data collection process may not be entirely reproducible and could introduce some inconsistencies.

2.2.2 Data Preprocessing

After gathering all the data, we end up with a dataframe with all the models of Hugging Face, in particular 170,464 data entries. Only 1,417 of these models report the carbon emissions needed to train the model. Further, we clean this dataset to homogenize the diverse data formats, enabling easier analysis. We followed these steps in our data preprocessing: a) feature engineering for further analysis; b) variable harmonization, curation, and filtering (e.g., CO_2e units harmonization); c) one-hot encoding of tags; d) filtering of tags by deleting language and auxiliary tags (e.g., 'arxiv', 'doi', etc.) and tags that are included in less than 100 models; and e) creating a dictionary relating tags to their ML application domain.

We focus on feature engineering, variable standardization, and one-hot encoding of tags. We create variables such as *co2_reported*, *auto*, *year_month*, and *domain* for filtering, splitting datasets, and analyzing model behavior across ML application domains. The domain variable was extracted through a mapping on the model tags to their corresponding domains, including: Multimodal, Computer Vision, NLP, Audio, and Reinforcement Learning.

We standardize variables like CO_2e, which are reported in different units (e.g., kg or g) or the *created_at* variable by converting it to datetime format. Also, we filter out models that may have inaccurate CO_2e reporting (e.g., CO_2e = 0).

Lastly, we one-hot encode the tags variable, which we gathered as a list in the tags attribute for each model during data collection. We then filter the one-hot encoded tags, removing irrelevant ones such as language tags or auxiliary tags like 'license', 'dataset', or 'doi'.

2.2.3 Manually Curated Carbon Emissions Dataset and Final Datasets

After data preprocessing, our dataset contains over 170,000 models, with 1,417 reporting carbon emissions. Approximately 1,293 of the models reporting carbon emissions are AutoTrained. Many of the AutoTrained models are suspected to be of low quality, as they do not report any attribute nor text documentation along the emissions. Moreover, some attributes cannot be automatically retrieved if not reported in the model card. Therefore, we manually improve the metadata of around 150 non-AutoTrained models by completing missing attributes, including performance metrics, filtering out models without documentation or with suspicious reporting, and removing excessively repeated baseline models fine-tuned for different tasks. This results in a cleaner, manually curated carbon emission dataset of around 50 models, used alongside the complete Hugging Face dataset (170,464 models) for general context analysis and the carbon emission dataset (1,417 models) for addressing RQ1 and RQ2.

2.3 Data Analysis

Next, we explain the data analysis methodology for the reproducibility of the results section.

2.3.1 RQ1 Analysis

We use the complete carbon emission dataset.

We perform a correlation analysis on several factors. The selection of these factors is is based on the assumption that these could be intrinsically related to the resource usage, and consequently, the energy consumption of ML models. Dataset and model size are chosen because larger datasets and models generally require more computations, which in turn consume more energy. The performance metrics are included as we aim to understand if there is any trade-off between model efficiency and energy consumption, which can guide the development of more energy-efficient yet powerful models. Further, the types of tasks (like pretraining or fine-tuning) and application domains (like NLP or Computer Vision) are chosen because they represent different computational workloads and patterns that can impact energy consumption.

To answer how the reporting of carbon emission evolves (RQ1.1), we analyze the monthly percentage of models reporting carbon emissions. To conclude with statistical significance if a trend exists, we perform a t-test on the slope parameters of a linear regression fitted to this evolution, with the null hypothesis (H_0) that there is no linear trend. We check the linear regression assumptions (residuals normality and homoscedasticity, among others) to ensure the accuracy of the t-test. In light of multiple hypotheses testing throughout the study, we apply the Holm-Bonferroni correction [43] to adjust the p-values, ensuring accurate comparisons to our fixed significance level ($\alpha = 0.05$). This approach helps control the familywise error rate and reduces the risk of false positives.

To study the evolution of the carbon emissions (RQ1.2), we perform a similar analysis with the monthly trend of the median *co2_eq_emissions*. We use the median rather than the mean, as there are outlier models reporting extreme carbon emissions that distort the analysis, e.g., recent large language models (LLMs). Equivalently, we perform a t-test on the slope of a fitted linear regression.

To study the evolution of the ML application domain trends on Hugging Face (RQ1.3), we use the harmonized domain variable along with other descriptive attributes.

Finally, for RQ1.4 classification criteria on carbon emissions reported, we develop a classification system that evaluates models based on their carbon emission reporting practices. This classification aims to provide a clearer understanding of the Hugging Face carbon emission reporting, encouraging more transparent reporting.

2.3.2 RQ2 Analysis

We use the complete carbon emission dataset and the cleaned non-AutoTrained models dataset. First, we evaluate if there is a trade-off between carbon emissions and model performance (RQ2.1). We calculate Spearman's correlation coefficient between each performance metric in the dataset (accuracy, F1, Rouge1, and RougeL) and reported CO_2e emissions, for both AutoTrained and manually curated non-AutoTrained models. We consider Spearman's non-parametric correlation rather than Pearson's as the performance metrics are not normally distributed. A significant positive correlation would indicate that this trade-off exists.

Next, we examine whether larger models and datasets are associated with increased carbon emissions (RQ2.2). To investigate this, we perform a Pearson correlation test applying log-transformations on the variables. The log transformation is applied to ensure the normality of the variables so that Pearson can be applied, as the three variables exhibit power law distributions. We also considered the manually curated dataset, as the *datasets_size* variable was manually extracted.

To study if fine-tuning tasks are more carbon-efficient than pretraining tasks (RQ2.3), we perform a Mann-Whitney U test at significance level $\alpha = 0.05$. As the reported CO_2e does not follow a normal distribution (the Shapiro-Wilk test shows a p-value of ≈ 0, with heavy tails and kurtosis), we use a non-parametric test comparing the ranks of the data points in the two groups, rather than their means via a t-test.

To study if some ML application domains are more carbon-efficient than others (RQ2.4), we again perform a Mann-Whitney U test with significance level $\alpha = 0.05$.

2.3.3 RQ3 Analysis

For the classification criteria study and the greenability sheet modelling (RQ3.1), we have developed a carbon efficiency classification system. Carbon efficiency is the ability to minimize greenhouse gas emissions (e.g, CO2), per unit of output or service provided. In the context of ML models, we will refer as carbon efficiency to the model capacity of minimizing emissions generated during training and deployment processes. We have adopted the use of index values as proposed in [39] and used in [41]. Indexing allows us to put all attribute values in relation to reference values, effectively dropping the units of the attributes and enabling easier comparisons. The carbon efficiency classification system used in this study is based on the weighted mean of the quartiles of the model for each indexed attribute. We consider the following attributes:

- CO_2e emissions: the caused carbon footprint. As the carbon emissions reported in HuggingFace follows a power-law distribution we opt by classifying the models on the quartiles of the index of carbon emissions log transformed, so that more balanced results are obtained. Otherwise, most of the models are classified in as CO_2e efficient.

- Size efficiency: the ratio of model_size to CO_2e emissions, which favors models that achieve low emissions even with high model complexity.
- Dataset efficiency: the ratio of datasets_size to CO_2e emissions, which favors models that achieve low emissions even with large datasets.
- Downloads: a proxy for reusability, with more downloads suggesting greater efficiency through reuse. Similarly to carbon emissions, as it follows a power-law case we opt by a handcrafted quantile selection that divides the downloads category on levels with reasonable boundaries between classes (that is, without overlapping boundaries or repeated levels); otherwise, as most of the models do not have any downloads, the classification is flawed.
- Performance score: the harmonic mean of normalized performance metrics (accuracy, F1, Rouge-1, Rouge-L), penalizing cases where a single metric is extremely low.

The attributes are assigned weights based on their significance for carbon efficiency: CO_2e emissions (0.35), model size efficiency (0.15), datasets size efficiency (0.15), downloads (0.25), and performance score (0.2). `distilgpt2` serves as the reference model for index values due to its high download count among models that report carbon footprint. The index reference is structured so that lower index values indicate better efficiency for a given attribute. Consequently, for attributes that should be maximized (size efficiency, downloads, and performance), the index is calculated as $i = \text{ref}/\text{val}$; for attributes that should be minimized (CO_2e emissions), $i = \text{val}/\text{ref}$. This allows us to classify each of the carbon emissions-reporting models into five carbon efficiency labels, from E to A (from less carbon efficient to more carbon efficient).

To allow an accessible and easy-to-use greenability sheet generation (RQ3.2), we have developed an interactive form and dashboard through Streamlit, a flexible Python framework that lets developers create data-focused web applications. Hosted on Streamlit Cloud[2], this tool is designed to make the generation of energy labels straightforward and user-friendly.

Within the Streamlit application, users are provided a form to input their model parameters, which are then processed to produce an energy label. This label provides an at-a-glance evaluation of the model's carbon efficiency. In addition to this form, the Streamlit application offers an interactive visualization feature where users can view carbon emissions data associated with a range of ML models. While the default dataset consists of models retrieved from Hugging Face, the application allows users to add their own model to this dataset, thus promoting a comprehensive and expanding source of energy efficiency data. The handling of such extensive and dynamic data has been made possible through the use of Google Cloud Storage and the Google Sheets API, ensuring the data's accessibility and longevity.

For the validation of the greenability sheet (RQ3.3), we have asked diverse ML practitioners and researchers to participate in a survey designed to assess the perceived ease-of-use and usability of the tool. This survey [3], hosted on Google Forms and based on the Technology Acceptance Model (TAM), seeks to understand the users' experience with the tool, their thoughts on its strengths and weaknesses, and its overall usefulness in their tasks related to improving carbon efficiency. Through the responses gathered from this survey, we aim to continuously improve the tool, making it an ever more valuable and user-friendly resource for the machine learning community working towards more sustainable AI practices.

[2] `https://energy-label.streamlit.app/`

[3] The survey can be accessed at `https://forms.gle/YtD6NxsfJTBcn9G78`

3. Results

In this section, we present a general analysis of the extracted Hugging Face model dataset, and then the results per RQ.

3.1 General Analysis of Hugging Face Models

As a preliminary analysis, we report general characteristics of the ML models stored in the Hugging Face repository.

3.1.1 Is Hugging Face's popularity increasing?

From the number of models uploaded each month in Figure 2, we can observe that Hugging Face's popularity has been increasing exponentially in the past years. This trend is expected and aligns with what has been observed across multiple platforms, particularly in media and machine learning (ML) community conversations in the past several months.

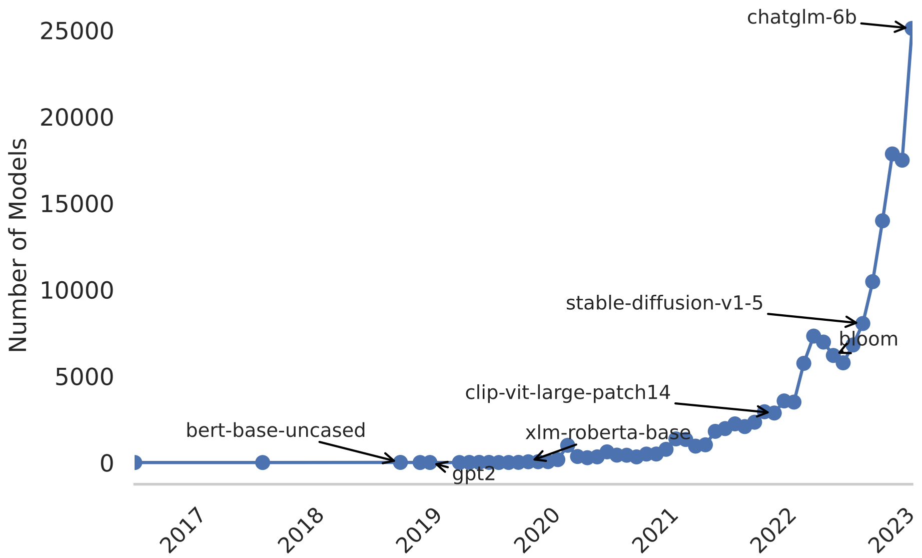


Figure 2: Evolution of total number of Hugging Face models added each month

3.1.2 What is the main ML application domain in Hugging Face?

Hugging Face models are attached with tags introduced by the user. These tags include information on several model characteristics, such as the task of the model, the library used, or other tags related to model configuration. In Figure 3, we can observe the most common tags in Hugging Face.

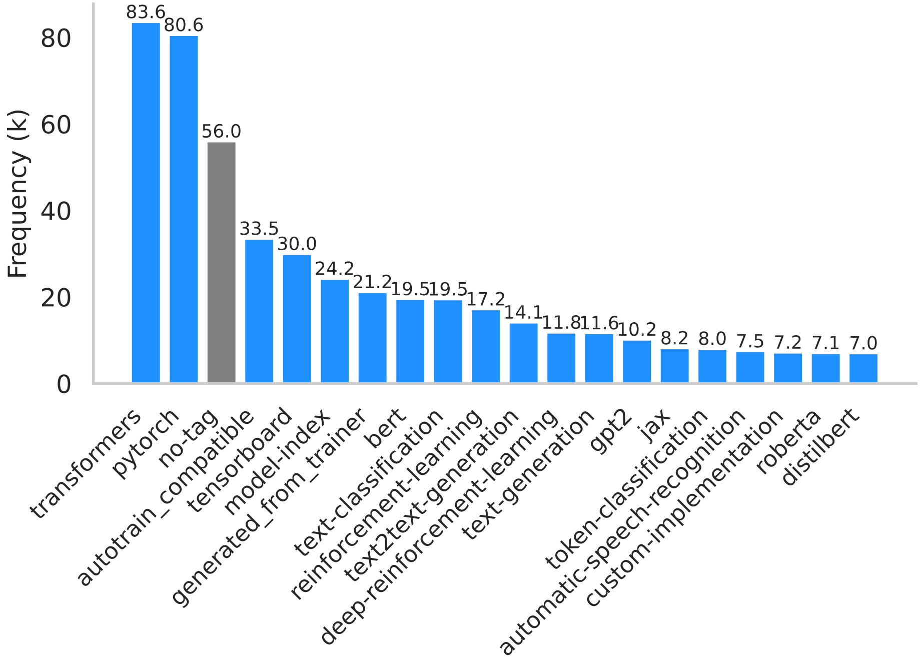


Figure 3: Most frequent Hugging Face model tags

The most popular tag is the *transformer* tag, along with *PyTorch*, the most used library in Hugging Face. The majority of the remaining tags are NLP-related, together with auxiliary tags (e.g., *autotrain_compatible*) or tags related to reinforcement learning. This NLP dominance can be further seen in Figure 4. NLP has been the dominant ML application domain since Hugging Face started. In fact, NLP was the only domain until the end of 2020, when multimodal or audio models started to appear. In the past months, the NLP dominance has been shrinking more than ever, with an increase in reinforcement learning models and other domains. Lastly, almost 30% of models have been published without any tag (no-tag), making their categorization impossible.

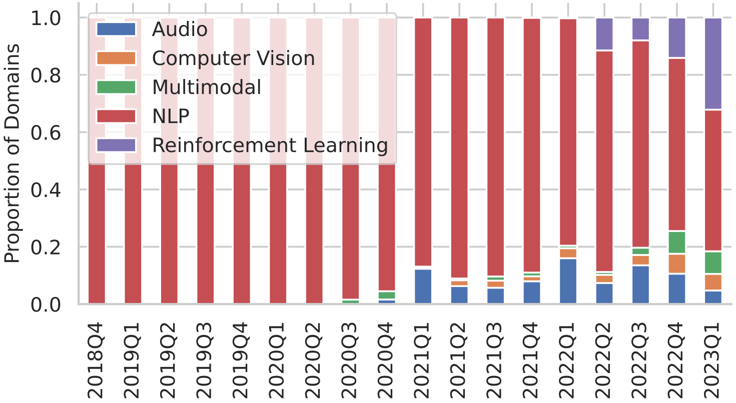


Figure 4: Application domain evolution of Hugging Face models

3.1.3 What is the main framework used in Hugging Face?

Figure 5 presents the evolution of the usage of the ML frameworks PyTorch, TensorFlow, Keras, and JAX. We chose these because they are among the most popular ML frameworks. As we can observe, the popularity of PyTorch has increased drastically, while TensorFlow's and JAX's popularity has decreased substantially. Keras shows a marginal popularity in comparison to the other frameworks, with scikit-learn playing no relevant role (fewer than 150 models in total).

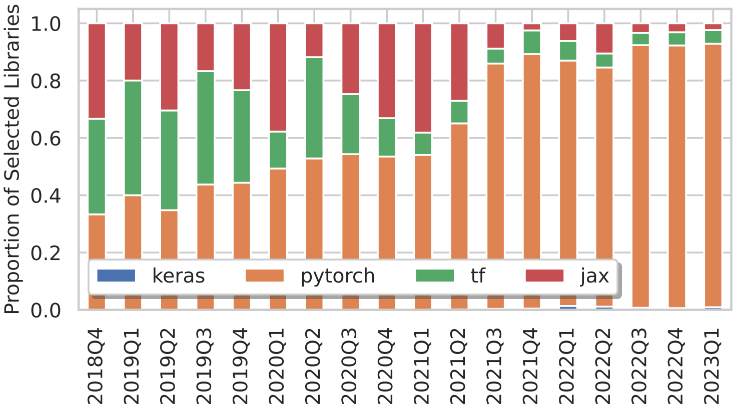


Figure 5: Evolution of the ML framework usage on Hugging Face

3.2 Carbon Emissions Reporting (RQ1)

Next, we present how model creators report carbon emissions on Hugging Face and what reporting techniques are used.

3.2.1 How has the reporting of carbon emissions evolved over the years?

> **Finding 1.1**. *Despite more models reporting carbon emissions, the percentage they represent over all models published in Hugging Face is not just marginal but stalled, pointing out lack of awareness of green AI by the community.*

As illustrated in Figure 2, the popularity of Hugging Face has increased significantly, with a growing number of models being published each month. Consequently, the number of models reporting their carbon emissions has also risen. However, to understand the overall interest in reporting carbon emissions or awareness of Green AI, we need to examine the evolution of the percentage of models that report their carbon emissions. Figure 6 presents the monthly aggregates for this.

Carbon emission reporting on Hugging Face started in mid-2021. However, this feature has unfortunately not experimented wide-spread adoption so far. The maximum percentage occurred in 10-2021, with 3.12% of the models reporting carbon emissions. The average percentage is 0.9%, and for the last month (03-2023), it is 0.62%. The median percentage is 0.92%. With the t-test result on the regression slopes (adjusted p-value = 0.57), we

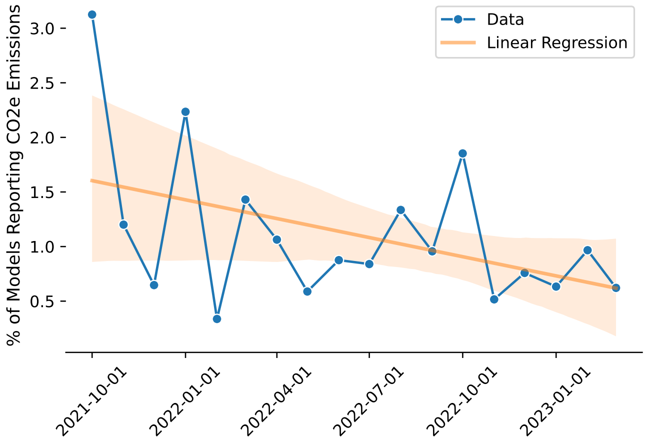


Figure 6: Evolution of the carbon emission reporting on Hugging Face

do not have enough evidence to consider a decreasing trend. However, the stall may indicate a lack of awareness or no perceived importance of Green AI concerns in the Hugging Face community.

3.2.2 How have the reported carbon emissions evolved over the years?

> **Finding 1.2**. *The carbon emissions reported on Hugging Face have slightly decreased in the past 2 years.*

In addition to analyzing the practice of carbon emission reporting, we also wanted to know if the reported carbon emissions increased, i.e., if recent models are more demanding. We can observe the evolution of the median carbon emissions aggregated by month in Figure 7. It should be noted that 10-2021 was not included in the figure, as it was the first month in which carbon emissions were published on Hugging Face, with an abnormally high median CO_2e (65.58g) compared to other months.

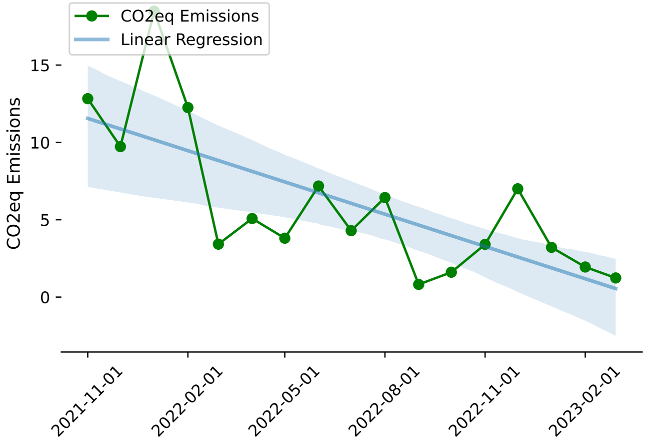


Figure 7: Evolution of the reported carbon emissions on Hugging Face

Contrary to expectations, the median reported CO_2e has been slightly decreasing over the past months. Excluding 10-2021, the maximum was reached in 01-2022 with 18.48g. For the whole timeframe, the median was 4.69g and the mean was 9.35g. The latter is roughly equivalent to the generated emissions by fully charging a single phone. In the last included month (03-2023), the reported emissions dropped to 1.24g. Based on the slope t-test (adjusted p-value of 0.011), we can conclude that there has been a statistically significant decrease in the reported carbon emissions over the last 2 years.

3.2.3 What are the main characteristics of the models reporting their carbon emissions?

> **Finding 1.3**. *While NLP dominates the carbon emissions reporting, computer vision shows the highest proportion of models within its domain reporting emissions in recent quarters. Other domains remain marginal.*

> **Finding 1.4**. *90% of the models that report carbon emissions in Hugging Face are AutoTrained models.*

As we saw in Figure 4, NLP has been the main domain in Hugging Face for the past years. As expected, this domination is replicated for those models reporting carbon emissions (≈ 85% of the carbon emissions models are NLP). Additionally, in Figure 8 we can visualize the evolution of the percentage of models in each domain reporting carbon emissions. Computer vision seems to be the domain showing the largest relative percentage of models reporting carbon emissions (e.g., for the last quarter, roughly 6% of computer vision models reported carbon emissions). Moreover, the other domains apart from NLP and computer vision seem to be rather marginal.

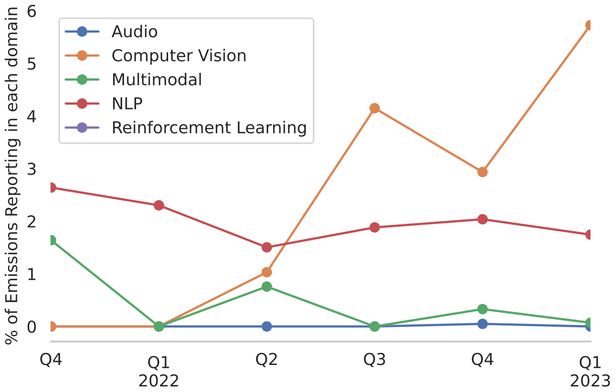
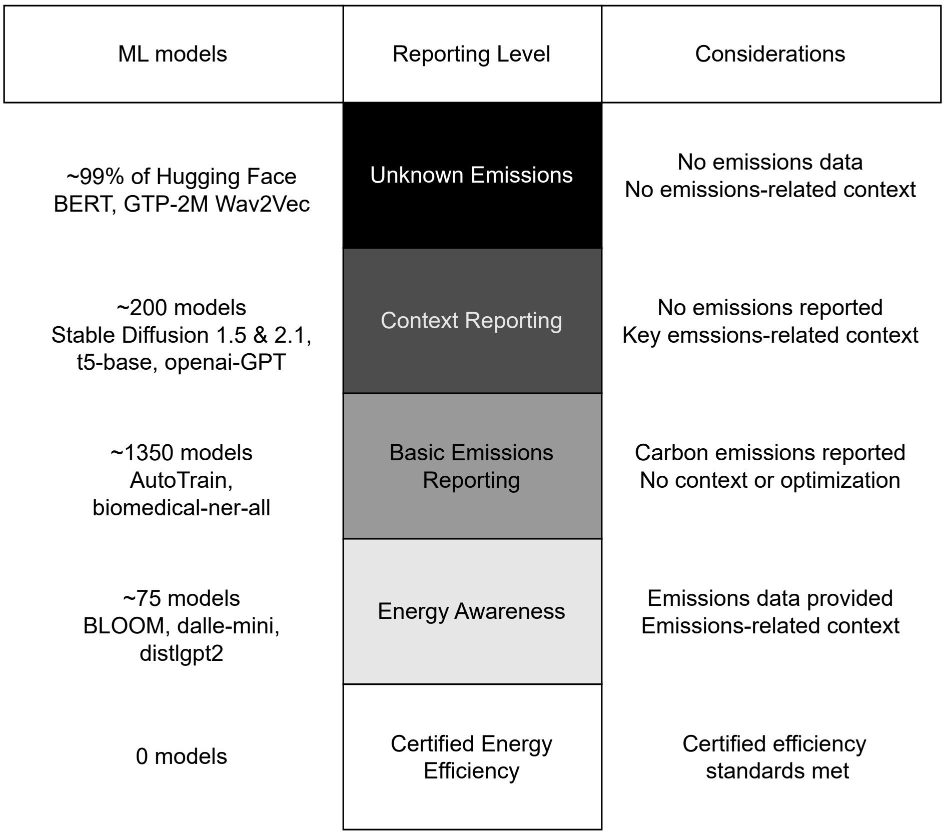


Figure 8: Evolution of the % of emissions reporting models in each domain

As a final remark regarding the characteristics of emissions models, over 90% of the models that report carbon emissions are AutoTrained, and only AutoTrained models provide performance metrics. AutoTrained models automatically report their performance metrics and carbon emissions upon publication. Consequently, many users might not have consciously reported this information, but rather did so as a result of the AutoTrain process.

3.2.4 How can we classify Hugging Face models based on their carbon emission reporting?

Using our classification scheme, our analysis reveals a significant disparity in the carbon emission reporting practices present on the platform. The results can be found in Figure 9.



ML models	Reporting Level	Considerations
~99% of Hugging Face BERT, GTP-2M Wav2Vec	Unknown Emissions	No emissions data No emissions-related context
~200 models Stable Diffusion 1.5 & 2.1, t5-base, openai-GPT	Context Reporting	No emissions reported Key emssions-related context
~1350 models AutoTrain, biomedical-ner-all	Basic Emissions Reporting	Carbon emissions reported No context or optimization
~75 models BLOOM, dalle-mini, distlgpt2	Energy Awareness	Emissions data provided Emissions-related context
0 models	Certified Energy Efficiency	Certified efficiency standards met

Figure 9: Carbon emission reporting classification

1. Unknown Emissions (99% of models): The vast majority of Hugging Face, including well-known models such as `BERT`, `GPT-2`, and `Wav2Vec`, do not report any emissions-related information (neither emissions nor context), which hides their environmental impact and inhibits the development of carbon-efficient practices.

2. Context Reporting (≈200 models): This category includes models that do not provide emission data but report some contextual attributes related to carbon emissions, e.g., the hardware used, the dataset size, or training location. Some notable models in this group are `Stable Diffusion 1.5 & 2.1`, `t5-base`, and `openai-GPT`. While explicit emission data is preferable, the provided training context is a step forward towards estimating energy consumption.

3. Basic Emission Reporting (≈1,350 models): Models in this category report carbon emission data without context or optimization efforts. This group includes most of the AutoTrained models, `biomedical-ner-all`, and others. While providing carbon emission data is an improvement over not reporting it, the lack of context and optimization limits the usefulness.

4. Energy Awareness (≈75 models): These models report both carbon emissions and context, indicating an increased awareness of carbon efficiency concerns. Examples are `BLOOM`, `dalle-mini`, and `distilgpt2`. Sharing both carbon emission data and context contributes to a better understanding of the relationship between carbon emissions and model performance.

5. Certified Energy Efficiency (0 models): Unfortunately, no models on Hugging Face currently fall into this category, which requires optimized carbon emissions and adherence to established energy efficiency categorizations such as the ones proposed in [41] or [40].

3.3 Correlations Between Carbon Emissions and Other Attributes (RQ2)

RQ2 addresses the relationships and correlations we can find between carbon emissions and other attributes in the dataset.

3.3.1 How are carbon emissions and model performance related?

> **Finding 2.1**. *We could not find a correlation between model performance and carbon emissions for Hugging Face models reporting carbon emissions.*

Since only AutoTrained models report performance metrics for the carbon emissions models, we only considered these models in this analysis. Using Spearman correlation, we obtain correlations close to 0 with p-values > 0.05 for every performance metric. Thus, it is reasonable to assume that this trade-off does not exist for AutoTrained models on Hugging Face. This result may be due to the nature of the metrics reported by the AutoTrain method, as they might represent non-reliable results. To further investigate this, we consider the manually curated dataset ($n = 48$). However, with p-values > 0.05 and correlations between -0.3 and 0.3, we again conclude that there is no evidence in our sample to support that carbon emissions and model performance metrics are related.

3.3.2 How are carbon emissions related to model and dataset size?

> **Finding 2.2**. *Larger models and larger datasets imply an increase in carbon emissions during training.*

Based on the correlation test between model size and carbon emissions with the curated dataset, we can conclude that there exists a positive correlation. Applying Pearson's correlation with the log-transformed variables to ensure normality, we obtain adjusted p-values < 0.05 and correlations ≈ 0.56. These results are visualized in Figure 10.

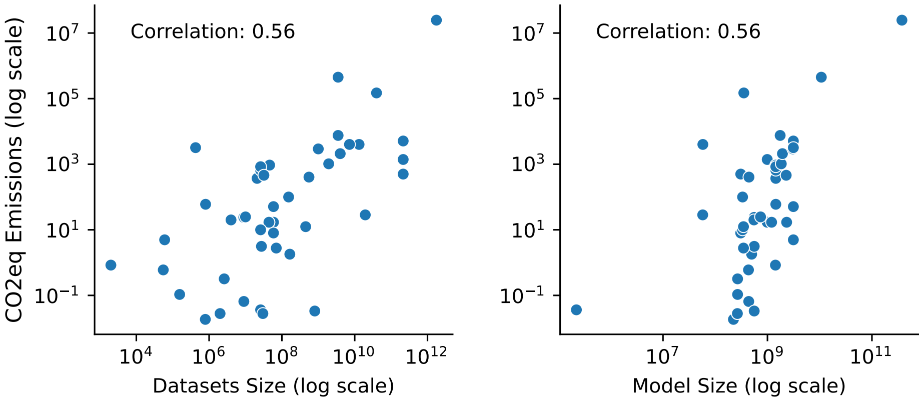


Figure 10: Correlation between carbon emissions and dataset / model size

3.3.3 What are the carbon emission differences between fine-tuned and pretrained tasks?

> **Finding 2.3**. *Although fine-tuning tasks appear to consume less than full pretraining tasks, we cannot conclude that the difference is statistically significant.*

In Figure 11, we can observe the carbon emissions divided by pretraining, fine-tuning, and pretraining + fine-tuning tasks. Pretraining involves training a model on a large dataset to learn general patterns and features, while fine-tuning refines the model on a specific task or domain using a smaller, targeted dataset. We can notice a difference between fine-tuning and pretraining tasks, where fine-tuning consumes on average 200% less than pretraining. Nonetheless, based on the Mann-Whitney U test (adjusted p-value = 0.29), we do not have enough evidence to conclude that this difference is, in fact, significant considering $\alpha = 0.05$. Despite the visual indication of a difference in the plot, the adjusted p-value suggests that the observed difference might be due to chance.

We can also observe that tasks combining pretraining and fine-tuning consume tasks involving only one of these processes (e.g., pretraining median is 432.67g vs. 228.92kg on pretraining + fine-tuning). This is expected since integrating both training stages requires additional computational resources.

3.3.4 How do ML application domains affect carbon emissions?

> **Finding 2.4**. *There is not enough evidence to consider that ML application domains affect carbon emissions.*

As we saw in RQ1.3, there is only a marginal number of audio and multimodal models that report carbon emissions. Thus, we exclude these ML application domains and only consider NLP and computer vision. According to Figure 11, there is a slight difference in carbon emissions between NLP and vision models: the latter seem to consume less. The Mann-Whitney U test confirms (p-value = 1.14e-15) that there is a statistically significant difference in carbon emissions between the computer vision and NLP domains. However, this difference may

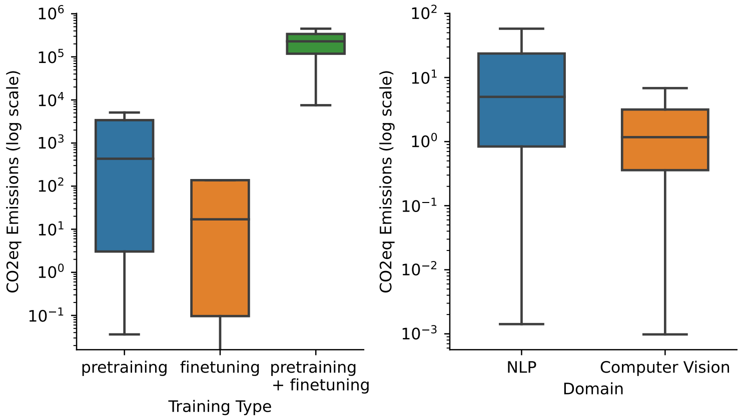


Figure 11: Carbon emissions by ML application domain and training type

be due to an increased model or dataset size for NLP models. After fitting a linear regression with all the available predictors (*is_nlp*, *is_cv*, *model_size*, excluding *datasets_size* as is only available for manual dataset), we can conclude that the intrinsic domain variable does not influence CO_2e (p-value0.23>0.05 for *is_nlp*). In fact, NLP and model size are positively correlated (Spearman's correlation coefficient = 0.41, p-value ≈ 0), which is the reason for the confounding effect.

3.4 Greenability Sheet Modelling & Implementation (RQ3)

3.4.1 How can we classify models based on their carbon efficiency?

Regarding RQ3.1, using the developed classification system, we categorize the models based on their carbon efficiency (see Figure 12).

We found the following distribution of Hugging Face models across carbon efficiency labels:

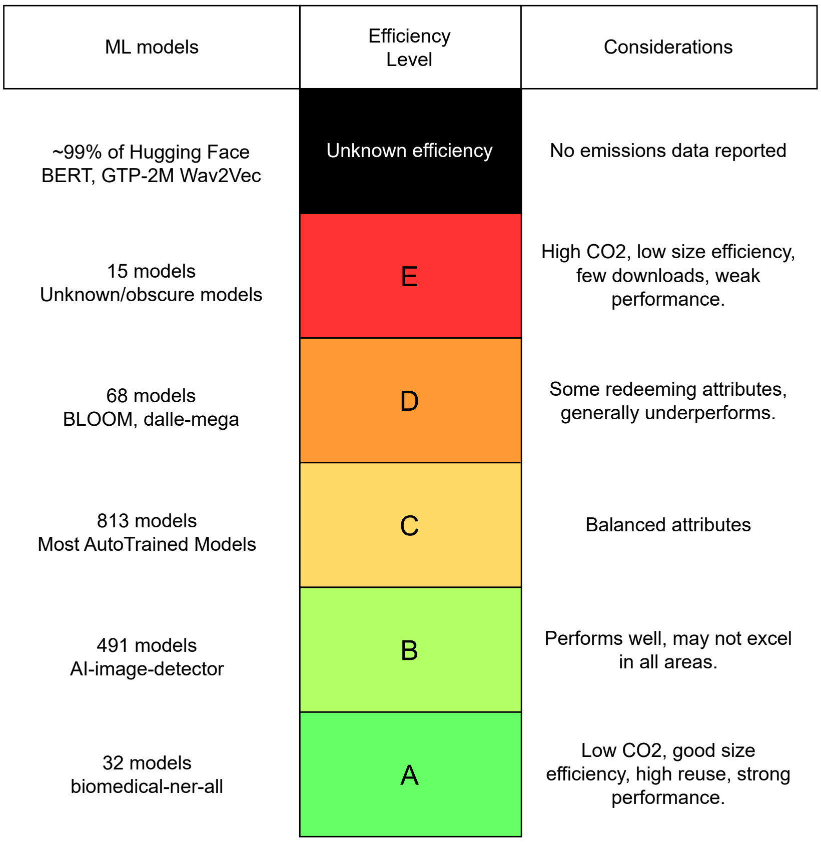
1. E Label (15 models): These models have high CO_2e emissions, relatively low model size, few downloads, and overall poor performance. Popular models are absent from this category due to low download counts.

2. D Label (68 models): Similar to E Label, but with at least one attribute outperforming the others, e.g., `BLOOM` has high CO_2e (4.01kg), poor size efficiency, but much more downloads (47,449) than average E models. `distilpgt2`, our reference model, also fails into this category

3. C Label (813 models): These models exhibit a balance between the attributes. Most of the AutoTrained models, which exhibit the median behaviour, fall here.

4. B Label (491 models): Examples include `AI-image-detector`, which has 1K downloads, rather low emissions (7.94g) and good performance (0.94 accuracy), reflecting better carbon efficiency.

5. A Label (32 models): These models demonstrate low CO_2e, high download counts, proper model size efficiency and good performance. Examples include: `biomedical-ner-all` with high reusability (15.5k downloads), low emissions (0.028g) and high complexity considering the low emissions; `BERT-Banking77` with high reusability (5,043 downloads), low emissions (0.033g) and good accuracy (0.926).



ML models	Efficiency Level	Considerations
~99% of Hugging Face BERT, GTP-2M Wav2Vec	Unknown efficiency	No emissions data reported
15 models Unknown/obscure models	E	High CO2, low size efficiency, few downloads, weak performance.
68 models BLOOM, dalle-mega	D	Some redeeming attributes, generally underperforms.
813 models Most AutoTrained Models	C	Balanced attributes
491 models AI-image-detector	B	Performs well, may not excel in all areas.
32 models biomedical-ner-all	A	Low CO2, good size efficiency, high reuse, strong performance.

Figure 12: Carbon efficiency classification

This classification system based on weighted means provides valuable insights into the carbon efficiency profiles of Hugging Face models. By analyzing models across multiple attributes, we can better understand the relationships between carbon efficiency, size, reusability, and performance. This information can guide ML practitioners and researchers in selecting models that not only meet their specific requirements, but also contribute to more sustainable ML practices.

3.4.2 Requirements Analysis

Next we define the requirements of the application that will introduce the energy efficiency classification and further emissions insights. A requirements analysis is an essential phase in the system development life cycle. It assists in understanding the needs of the users and translating these needs into specific functionalities and qualities that the software should exhibit. This analysis ensures that the software, once developed, is fit for its intended purpose. It sets clear expectations and prevents misunderstandings between developers and users, leading to a successful product.

This analysis is divided into functional requirements and non-functional requirements. Functional requirements are directly related to the system's functionalities and determine what the system is supposed to do. Non-functional requirements, on the other hand, describe how the system should behave; they set the standards for performance, reliability, usability, etc.

3.4.2.1 Functional Requirements

- **Model Introduction Functionality**: The system should provide an interface for users to introduce a new machine learning model. This functionality must allow users to input all necessary parameters, including model characteristics, performance metrics, and context-related information.
- **Energy Label Generation**: The system should generate an energy label for any model - new or existing in the system. The label generation process must consider all inputted parameters and calculate an accurate energy efficiency rating. In case any parameter is not introduced due to lack of information from the user, the label should still be generated with the given information.
- **Carbon Emissions Data Visualization**: The system must offer a comprehensive and interactive data visualization interface. This interface should allow users to view carbon emissions data related to various ML models and should support filtering based on different parameters. The visualizations should give the user a general sense on the evolution of carbon emissions and the boundaries on the energy efficiency rating calculation. Moreover, basic statistics should be provided related to the filtered configuration.

3.4.2.2 Non-functional Requirements The following non-functional requirements, which need to be consistently implemented throughout the application to maintain its quality, are sorted in accordance with the Volere requirements taxonomy [44]. We also include requirements related to data and ML:

Look and Feel

	Appearance
Description	The system should offer a clean, intuitive, and user-friendly interface. The design should inspire familiarity by borrowing elements from conventional energy labels.
Justification	An intuitive and familiar design can reduce the learning curve and increase user engagement with the system.
Condition of Satisfaction	Users can independently navigate and use the system effectively after an initial walkthrough or guide of no more than 4 minutes. This assumes a clear and intuitive layout.

Usability and Humanity

	Ease of Use
Description	The system should be easy to use, even for non-technical users.
Justification	As the tool might be used by a diverse range of users, ensuring ease of use is crucial to facilitate user engagement and satisfaction.
Condition of Satisfaction	Users can input model parameters, generate energy labels, and use the data visualization interface with minimal effort, with tasks achievable in no more than 3-4 straightforward steps, providing a clear and accessible guidance available if needed.

Performance

	Reliability and Availability
Description	The system should be reliable and available for use at all times.
Justification	Unreliable software can frustrate users and undermine the system's credibility.
Condition of Satisfaction	Users should be able to access and use the system 99% of the time, ensuring high availability while also accommodating scheduled maintenance and unforeseen circumstances.

	Capacity, Speed, and Latency
Description	The system should be able to handle a large number of models and generate energy labels within a reasonable timeframe.
Justification	High capacity, speed, and low latency can improve user experience significantly.
Condition of Satisfaction	The system can handle a large volume of data, specifically up to approximately 350,000 models, and return results within a maximum latency of 3 seconds through Streamlit's cloud service.

	Robustness
Description	The system should be robust and able to handle unexpected inputs or errors gracefully.
Justification	Robustness ensures that the system can function effectively in various scenarios without crashing or failing.
Condition of Satisfaction	The system should not crash or behave unexpectedly due to unforeseen user inputs or other unpredictable circumstances.

Maintainability

	Adaptability
Description	The system should be adaptable, allowing for easy updates and enhancements over time.
Justification	Adaptability ensures the system can evolve with the changing requirements and expectations of users.
Condition of Satisfaction	Updates and enhancements to the system can be implemented with minimal effort and without causing significant disruption to existing functionalities.

Data and Machine Learning

	Data Quality
Description	The system should be able to handle different qualities of data, compensating for missing or incomplete data when generating energy labels.
Justification	Data quality varies across different models and datasets. The system should be flexible enough to accommodate this variability.
Condition of Satisfaction	The system can generate accurate and reliable energy labels, even with incomplete or missing data.

3.4.3 System's Design

The system design and architecture section aims to illustrate the conceptual and physical structure of the application (RQ3.2), providing an overview of the different components that constitute the system and how they interact. This understanding is crucial for ensuring the app's efficient implementation, maintenance, and enhancement. It provides stakeholders with a clear image of how the application will function and how data will flow through the system.

3.4.3.1 General Scheme Our application is a web-based tool designed following the principles of a three-tier architecture. This architectural model allows the separation of concerns, dividing the system into the UI Layer, the Application Logic Layer (also known as the Business Logic Layer), and the Data Layer:

- **UI Layer**: This is the front-end layer of the application, with which the user interacts. It includes the user interface and the functionalities that allow users to introduce a new model into the system, generate energy labels for new and existing models, and visualize carbon emissions data.
- **Application Logic Layer**: This layer, also known as the Business Logic Layer, processes user requests, applies business rules, and manages data flow between the Presentation and Data Layers. It encompasses the energy label generation process and the carbon emissions data visualization logic.
- **Data Layer**: The Data Layer manages the storage and retrieval of application data. For this system, it involves the interaction with Google Cloud Storage and Google Sheets API for storing, accessing, and manipulating data related to machine learning models and their energy efficiency metrics.

Following, on Figure 13 we have the general scheme diagram for the application. The diagrams shows the relationships between components abstracted from the code that all together encompass the web-app.

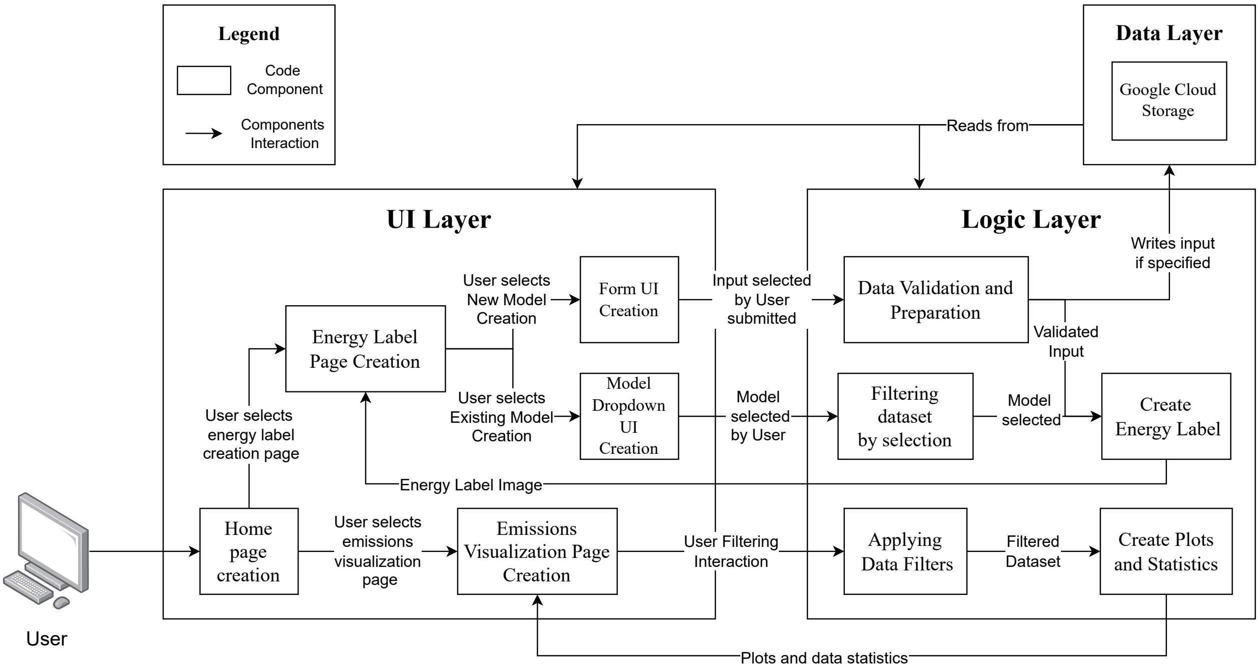

Legend
Code Component
Components Interaction
Data Layer
Google Cloud Storage
Reads from
UI Layer
Logic Layer
Writes input if specified
User selects New Model Creation
Form UI Creation
Input selected by User submitted
Data Validation and Preparation
Energy Label Page Creation
Validated Input
User selects Existing Model Creation
Model Dropdown UI Creation
Model selected by User
Filtering dataset by selection
Model selected
Create Energy Label
User selects energy label creation page
Energy Label Image
Home page creation
User selects emissions visualization page
Emissions Visualization Page Creation
User Filtering Interaction
Applying Data Filters
Filtered Dataset
Create Plots and Statistics
User
Plots and data statistics

Figure 13: Application General Scheme

3.4.3.2 System's Components

- **Model Introduction & Energy Label Creation Component**: When a user introduces a new machine learning model, the data is captured in a form. This includes information on the model's CO2 emissions, file size, dataset size, number of downloads, and performance metrics. Other contextual fields for capturing the name of the model, dataset used, computing environment, and geographical location are also included in the form.

  The form data is processed in the Application Logic Layer. Here, validation checks are run, and the data is prepared for storage. The processed data is then sent to Google Cloud Storage through the Google Sheets API. If the user selects the option to include the model in the main dataset, the system takes the additional step of updating the main dataset with the new model's information.

  Once the model data is stored, the system immediately retrieves the data for energy label generation. The Application Logic Layer processes the stored model data to calculate the energy efficiency, which is then transformed into a familiar visual grading system. This label, providing a breakdown of the ratings for individual parameters, is then presented to the user, thus allowing them to gain a detailed understanding of their model's energy efficiency.

- **Carbon Emissions Data Visualization Component**: This component fetches the necessary data from the Google Cloud Storage, applies the user's filter parameters, generates the corresponding visualizations, and presents them on the data visualization page.

The system design provides a streamlined, efficient process for model introduction and energy label generation. It maintains a scalable, modular, and maintainable architecture while ensuring reliable service to the end-users. The design also accommodates dynamic data handling, ensuring users receive accurate and up-to-date energy efficiency metrics for ML models.

3.4.3.3 Scalability The application's design supports scalability. With the application logic separated from the data layer, additional resources can be deployed to either layer independently to accommodate load. Furthermore, the use of Google Cloud Storage and Google Sheets API ensures the application can handle substantial amounts of data efficiently, providing a path for future scaling.

3.4.3.4 Maintainability and Ease of Use Maintainability has been a key consideration in the design of the application. The three-tier architecture allows for independent updates or modifications to each layer without affecting the others, simplifying maintenance tasks. Additionally, the use of popular, well-documented technologies like Google Cloud Storage and Google Sheets API ensures easier updates and bug fixes.

The application has also been designed with a focus on user experience. The UI is designed to be intuitive and user-friendly, allowing users to easily introduce new models and generate energy labels. The visualization of carbon emissions data is also designed to be straightforward and easily interpretable. This ease of use promotes user engagement and facilitates the wider adoption of the tool.

3.4.4 Survey Feedback on the Energy Efficiency Evaluation Tool

Following we have the feedback we received on the survey regarding our web application (RQ3.3). In this initial stage, we have collected 15 responses from the tool's users. Although the number of responses is small, their feedback provides valuable initial insights into the user experience and the tool's potential areas for improvement. This is just preliminary and we plan to continue collecting responses to obtain a more comprehensive understanding of the user experience and areas of improvement[4].

3.4.4.1 Participants Context The participants in the preliminary study encompass a range of occupations, with a significant number being students (60%), followed by researchers (20%), software engineers (13%), and a data scientist (7%). When queried about their familiarity with the concept and practices of Green AI, while a majority indicated some degree of familiarity (with 6 agreeing and 1 strongly agreeing), there is a notable portion of participants who expressed a neutral stance or unfamiliarity with Green AI practices (with 5 being neutral, 2 disagreeing, and 1 strongly disagreeing). This paints a mixed picture of the participants' existing knowledge and understanding of Green AI, which is a crucial context to consider when interpreting their feedback on the tool. The results can be seen in Figure 14 and Figure 15

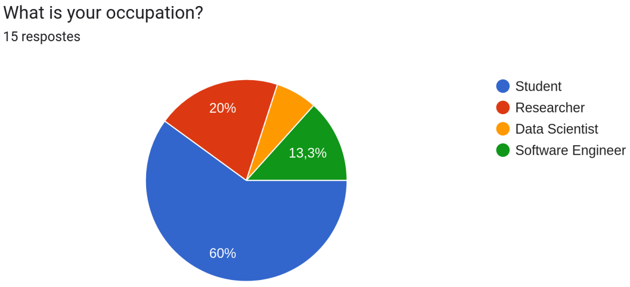


Figure 14: Occupation Survey Results

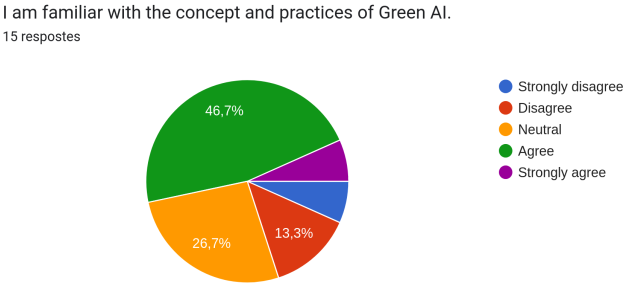


Figure 15: Green AI Familiarity Survey Results

3.4.4.2 Perceived Ease-of-Use and Usefulness On the aspect of ease of use, the responses show a very positive reception, with most users agreeing or strongly agreeing that the tool is easy to navigate, well-structured, and straightforward to use. The responses to these questions would be beneficially visualized as a bar chart, showing the distribution of responses. The results can be further seen in Figure 16

In terms of perceived usefulness, the tool received generally positive feedback. A majority of the users found the tool's elements informative (14 out of 15), valuable (13 out of 15), and useful (11 out of 15) for tasks related to assessing carbon efficiency. However, some users remained neutral, indicating potential areas for improvement in conveying the tool's utility. The results can be further seen in Figure 17

3.4.4.3 User Experience and Recommendation The overall user experience was deemed positive by the majority of the users (14 out of 15). This suggests that the tool's design, functionality, and overall concept resonated well with the users. Moreover, encouragingly, every user would recommend the tool to others working on Green AI. This suggests that the tool has value for its intended audience and has the potential to contribute positively to the field. The complete results are in Figures 18 and 19

[4] The survey can be accessed at `https://forms.gle/YtD6NxsfJTBcn9G78`

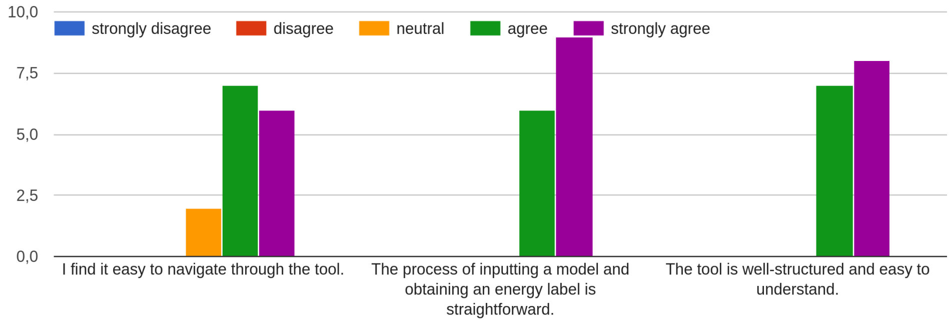


Figure 16: Ease of Use Survey Results

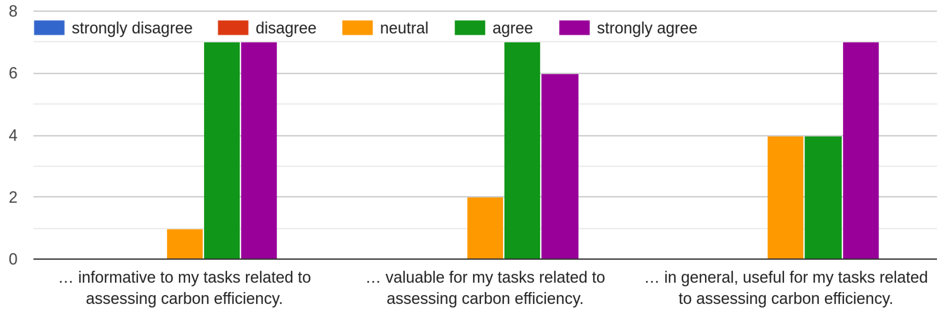


Figure 17: Usefulness Survey Results

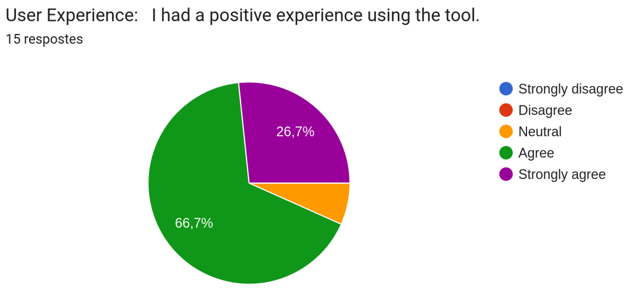


Figure 18: User Experience Survey Results

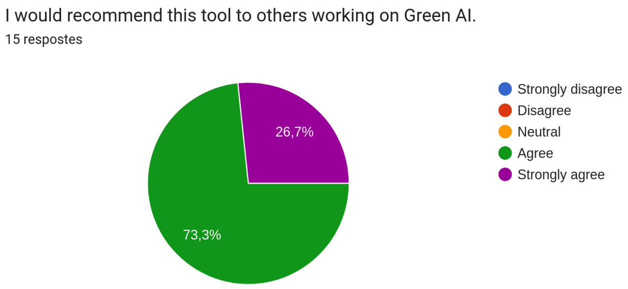


Figure 19: Recommendation Survey Results

3.4.4.4 Strengths and Weaknesses

User-identified strengths of the tool include its potential for raising awareness about the energy consumption of ML models, the ease of entering model parameters, and its overall user-friendly design. As for areas of improvement, some users expressed confusion about how certain parameters, like the number of downloads, contribute to the energy efficiency label. They also raised concerns about the parameters' contribution appearing to be narrow at times. Moreover, a user expressed the desire for more context about how the energy efficiency rating is obtained.

In response to these concerns, we plan to improve the tool's transparency by providing more details about the computation of energy efficiency labels and the roles of different parameters. Furthermore, we will investigate the mentioned issues and consider adjusting the influence of the parameters on the energy label to better reflect their significance.

3.4.4.5 Conclusion and Future Work Overall, the preliminary analysis of the survey responses provides promising feedback for our Green AI energy efficiency evaluation tool. We plan to continue collecting responses to obtain a more comprehensive understanding of the user experience and areas of improvement. Given the feedback, future work will focus on enhancing the tool's transparency, further investigating the influence of various parameters on the energy efficiency labels, and continuously improving its user experience.

4. Implications

The findings and the web-app of this project have several important implications for the ML research community and industry practitioners. By identifying the current state of carbon emission reporting for ML models on Hugging Face and developing the energy label generation, this research highlights areas of improvement and offers recommendations for promoting carbon-efficient ML development.

4.1 Hugging Face Insights Implications

4.1.1 Raising Awareness, Encouraging Transparency, and Standardizing Reporting Practices

Our analysis in reveals that despite the growing popularity of Hugging Face, the proportion of models reporting carbon emissions has stalled, suggesting a gap in ML sustainability awareness. Moreover, there is a clear lack of standardization in carbon emission reporting on Hugging Face, with ML practitioners reporting carbon emissions and emission context attributes without proper guidelines, leading to missing attributes (such as location, hardware, or model and datasets sizes). This lack of awareness and standardization in reporting practices may also be contributing to the absence of models meeting existing carbon efficiency classifications (e.g., [41]), as we saw in RQ1.5 with the carbon reporting classification.

Emphasizing the importance of reporting energy consumption can help promote standardized reporting guidelines and best practices for sustainable ML development. The AI community should actively promote the importance of energy-efficient model development, transparent energy reporting practices, and energy efficiency certification [45]. Establishing standardized reporting guidelines can help improve the consistency and quality of energy data and context reported.

To address these issues in carbon emissions reporting on Hugging Face, we propose the following initial guidelines:

- Carbon emissions (CO_2e): It should be reported consistently across all models to enable comparisons (gCO_2e). Some tools to report carbon emissions are `CodeCarbon` [20], `Carbontracker` [22] or `Eco2AI` [23].
- Energy consumption metrics: Report power consumption in kilowatt-hours (kWh) during training and inference, e.g., via tools such as `NVML` [46] or `RAPL` [47] and training time. This metric depends more on software design decisions and is more precise to judge efficiency than carbon emissions alone, which depends on carbon intensity. Further proposals on the estimation of energy consumption can be seen in [30, 18].
- Key energy-related context information: Include accurately attributes such as: hardware, location, energy source, model size, dataset size, and performance metrics. These factors can significantly impact the carbon footprint of a model and can be useful to evaluate the trade-off on carbon emissions. This information should be included in the Model Card `README` on Hugging Face for easier information retrieval.
- Energy optimization techniques: Encourage reporting of any energy optimization techniques used during the model training and deployment process in the Model Card text. This information can help others in adopting similar practices to improve energy efficiency.

These guidelines could be translated into the following extended Hugging Face metadata proposed in [26]:

```
co2_eq_emissions:
    emissions: number (in grams of CO2e)
    power_consumption: number (in kWh)
    source: source of the information, e.g., code carbon, from AutoTrain, mlco2 calculator, etc.
    training_type: pre−training or fine−tuninghttps://es.overleaf.com/project/63f7772d9b98d443fe9b1954
    geographical_location: as granular as possible.
    hardware_used: how much compute and what kind, e.g., 8 v100 GPUs
    training_time: training duration (in seconds)
    optimization_techniques: any energy optimization techniques used during the model training and deployment process
    energy_label:
        − energy_label_source: certification label name and source
          energy_label_classification: energy classification
model_info:
    model_file_size: size of the model resulting file
    number_of_parameters: number of parameters of the model
    datasets_size: size of the dataset used
    performance_metrics:
        − metric: e.g., accuracy
          value: value of the metric, e.g., 0.92
        − metric: e.g., f1
          value: value of the metric
```

The app form on the creation of the energy label follows a really similar metadata proposal.

4.1.2 Developing Energy-Efficient Models and Enhancing Carbon Efficiency Classification

Our proposal for classifying the carbon efficiency of ML models is based on these practices:

- Minimize CO_2e emissions: Focus on reducing the environmental impact during model training and deployment.
- Encourage reusability: Share and promote models that can be easily adapted for various tasks, increasing their download count and overall efficiency.
- Find the proper trade-off on model size and dataset size: Our findings show that larger models and datasets lead to increased carbon emissions during training.
- Maintain strong performance: Ensure models have balanced performance across key evaluation metrics. Our research shows that model performance may not be directly related to carbon emissions, as we could not find enough evidence to relate performance with carbon emissions, which means it may be possible to develop efficient models without sacrificing too much performance.

By adopting the above best practices and enhancing the carbon efficiency classification, researchers and developers can create models that meet specific requirement while contributing to environmentally sustainable ML practices.

4.2 Implications of the Greenability Sheet Web-App

Our web application, through its user-friendly interface and transparent methodology, carries several potential implications for both the developer community and the wider landscape of AI.

4.2.1 Democratizing Energy Efficiency

The web-app bridges the gap between technical and non-technical stakeholders by providing an accessible platform to understand and measure the carbon efficiency of ML models. This democratization of energy efficiency analysis has the potential to invite broader discussions and engagement around sustainability in AI.

4.2.2 Promoting Transparency

By facilitating the generation and easy access to energy labels for ML models, our web-app promotes transparency within the AI community. As more models are assigned energy labels, this platform serves as a mirror to the community, reflecting the current landscape of carbon emissions in AI. This transparency could imply for developers and researchers to acknowledge energy efficiency as a crucial factor when designing new models.

4.2.3 Feedback Loop for Model Developers

Developers can leverage the app as a feedback mechanism during the model development process. They can gauge the potential energy efficiency of their model in its earlier developmental stages and adjust their strategies accordingly before finishing their work.

4.2.4 Potential for Standardization

As an initial step, our web-app introduces a potential direction for the standardization of measuring and reporting the energy efficiency of AI models. While we do not claim to offer a complete solution, we hope that our initiative could inspire further developments in this direction, contributing to the broader conversation about standardization in AI energy efficiency reporting.

5. Threats to Validity

Following empirical standards on repository mining [12], we reported the applied good practices throughout the project, e.g., explaining why Hugging Face mining is appropriate for our research goal, describing data preprocessing, and providing a detailed data collection and preprocessing pipeline available at the repository. Below, we discuss several potential validity threats and associated mitigation actions.

5.1 Hugging Face Analysis Threats to Validity

Construct Validity: The primary threat to construct validity is the reliance on self-reported carbon emission data from Hugging Face, which may not accurately reflect the actual carbon emissions due to variations in measurement methodologies or inconsistencies in reporting. Moreover, the manual curation of the carbon emission dataset could introduce bias in the selection and filtering of models. To address the self-reporting issue, mitigation actions were applied during the Data Preprocessing stage, which involved a harmonization process to standardize all model reports. Moreover, Hugging Face has implemented a carbon emission reporting proposal to address the self-reporting issue and AutoTrained models automatically report carbon emissions, which adds a layer of transparency and reduces the risk associated with self-reported data.

Internal Validity: While we controlled for certain variables such as model size and datasets size, other factors like training setup, hardware, and data preprocessing might also influence the carbon emissions of a model. Additionally, the lack of informative model cards for many AutoTrained models might limit our ability to accurately assess the quality and representativeness of these models in our study.

External Validity: Our findings based on Hugging Face models, a NLP predominant repository, may not be fully generalizable to other ML application domains like computer vision. Moreover, while our analysis covered a substantial sample of over 1,400 models (emissions-reporting models), it did not capture the entire range of models available in Hugging Face. Also, evolving trends in ML and energy efficiency could also affect the applicability of our findings in future contexts. These limitations can be mitigated through the data pipeline in the repository, allowing for data updates when desired.

Conclusion Validity: The lack of standardized reporting practices for carbon emission data might have affected our ability to accurately compare models and draw conclusions. We made assumptions (e.g assuming test evaluation when not specified) based on the available information, which may not always be correct or complete. To improve conclusion validity, future research should seek to establish more standardized reporting and data collection practices for carbon emissions.

5.2 Energy Label Web-app

Construct Validity: Our tool relies heavily on user-reported data when assessing the energy efficiency of a new model. This means that the accuracy of the generated labels largely depends on the reliability and comprehensiveness of the user-provided data, which can vary. To mitigate this, we provide clear user guidance and instructions for data submission to help ensure accuracy and consistency. On the other hand, the tool's energy efficiency grading system, while robust, may not capture all the nuances of energy consumption. Factors like the efficiency of the hardware used or specifics of the training techniques employed, which might significantly impact energy usage, are not considered in the grading system. Future work could plan on the validation and refinement of the classification system based on the latest research findings to minimize this threat.

Internal Validity: We assume that the users who submit new models are providing accurate and honest data. However, there's no definitive way to verify this. Users could inadvertently or intentionally provide incorrect information, leading to misclassified models. To address this, we implemented strong error handling and user input validation to minimize the impact of incorrect or misleading user submissions. Moreover, the performance of the web-app is subject to the capabilities of the Streamlit framework and the Google Sheets API. Issues or limitations with these platforms may impact the app's functionality and reliability.

External Validity: The primary dataset on which classification is defined and visualizations made is the Hugging Face dataset. As such, it may not be entirely applicable or accurate for models developed or hosted on other platforms or designed for tasks outside of those commonly found on Hugging Face. Also, the use of Google Sheets to store datasets externally might limit its scalability when dealing with larger datasets or increased user traffic, which could hinder its use in a broader or more intensive context. To counter this, the code has been designed to be abstract enough to facilitate a switch to other external databases in case its necessary in the future.

Conclusion Validity: The tool assigns energy efficiency labels based on a specific set of criteria. While these criteria are rigorously defined, the interpretation of these labels can introduce a margin of error. Users might over-simplify or misinterpret the meaning of these labels, leading to potential inaccuracies in the conclusions drawn from the tool's output. Lastly, the application's visualizations are based on the current state of models on Hugging Face, as well as any additional models added by users. Given the fast-paced nature of the AI field, these visualizations may quickly become outdated and may not represent the most current trends or practices in model development and energy efficiency.

6. Conclusions

This project addresses the critical issue of carbon emissions and sustainability in the field of Machine Learning (ML), with a particular focus on the Hugging Face model repository. By analyzing carbon efficiency and reporting practices, we have provided invaluable insights and recommendations for ML practitioners and researchers.

We began by assessing the sustainability awareness within the ML community. Despite the increasing popularity of Hugging Face, we found a concerning trend - the proportion of models that report their carbon emissions has stalled and even slightly decreased. This lack of reporting and the absence of standardized practices emphasize the need for establishing guidelines and promoting an energy efficiency certification system.

To tackle this issue, we examined correlations between carbon emissions and other relevant factors within Hugging Face ML models. This analysis yielded insights that can guide the development of more carbon-efficient models. Moreover, we proposed a carbon efficiency classification system, providing a practical tool for ML practitioners and researchers to make environmentally conscious choices in their model selection and development. All our findings have been compiled in the following paper [48], which has been accepted at the 2023 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM).

However, recognizing the complexity of the matter and the need for accessible solutions, we implemented a web application (publicly available on `https://energy-label.streamlit.app/`) that deploys the findings of our research. This app, built using the Streamlit framework, allows users to evaluate the carbon efficiency of ML models by generating energy labels based on their input parameters. Users can introduce new models into the system, and it also enables them to view and understand carbon emissions visually through insightful charts and graphs.

The web application allows users to generate energy labels for new models and view labels for existing models within our dataset. Our system is designed with flexibility in mind, accommodating various levels of data availability and giving users the choice to add their models to the overall dataset. On the data visualization page, users can filter and analyze data according to their needs, providing a tool for understanding the current state of energy efficiency across models and identifying areas for potential improvements. All the above functionalities are fed by the +1,400 Hugging Face ML models dataset hosted on Google Sheets. The code of the data analysis and web application is publicly available on `https://github.com/GAISSA-UPC/ML-EnergyLabel`.

Through this project, we encourage the ML community to look after awareness, transparency, standardized reporting practices, and strive for the development of energy-efficient models. The insights and tools presented here could be valuable additions to the research literature of Green AI.

References

[1] UPC. Universitat Politècnica de Catalunya. English — gaissa.upc.edu. `https://gaissa.upc.edu/en`.

[2] Coral Calero, Mª Ángeles Moraga, and Mario Piattini. Introduction to software sustainability. *Software Sustainability*, pages 1–15, 2021.

[3] Anders S. G. Andrae and Tomas Edler. On global electricity usage of communication technology: Trends to 2030. *Challenges*, 6(1):117–157, 2015.

[4] B. Penzenstadler, V. Bauer, C. Calero, and X. Franch. Sustainability in software engineering: a systematic literature review. In *16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012)*. IET, 2012.

[5] Special Eurobarometer. 501: Attitudes of european citizens towards the environment, 2019.

[6] International Organization for Standardization. *Guidance on social responsibility*. na, 2010.

[7] Silverio Martínez-Fernández, Justus Bogner, Xavier Franch, Marc Oriol, Julien Siebert, Adam Trendowicz, Anna Maria Vollmer, and Stefan Wagner. Software Engineering for AI-Based Systems: A Survey. *ACM Transactions on Software Engineering and Methodology*, 31(2):1–59, April 2022.

[8] Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. Green ai. *Communications of the ACM*, 63(12):54–63, 2020.

[9] Dimitrios Stamoulis, Ermao Cai, Da-Cheng Juan, and Diana Marculescu. Hyperpower: Power-and memory-constrained hyper-parameter optimization for neural networks. In *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 19–24. IEEE, 2018.

[10] Hugging Face Inc. Hugging Face Hub Documentation. `https://huggingface.co/docs/hub/index`, 2023.

[11] Shashank Mohan Jain. Hugging Face. In *Introduction to Transformers for NLP*, pages 51–67. Apress, Berkeley, CA, 2022.

[12] Preetha Chatterjee, Daniel German, Tushar Sharma, and Melina Vidoni. Repository Mining Standard. `https://acmsigsoft.github.io/EmpiricalStandards/docs/?standard=RepositoryMining`, 2023.

[13] Christoph Becker, Ruzanna Chitchyan, Leticia Duboc, Steve Easterbrook, Birgit Penzenstadler, Norbert Seyff, and Colin C. Venters. Sustainability Design and Software: The Karlskrona Manifesto. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, pages 467–476, Florence, Italy, May 2015. IEEE.

[14] Roberto Verdecchia, Patricia Lago, Christof Ebert, and Carol De Vries. Green IT and Green Software. *IEEE Software*, 38(6):7–15, November 2021.

[15] Stefanos Georgiou, Maria Kechagia, Tushar Sharma, Federica Sarro, and Ying Zou. Green AI: Do deep learning frameworks have different costs? In *Proceedings of the 44th International Conference on Software Engineering*, pages 1082–1094, Pittsburgh Pennsylvania, May 2022. ACM.

[16] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and Policy Considerations for Deep Learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, 2019. Association for Computational Linguistics.

[17] Luís Cruz. All you need to know about Energy Metrics in Software Engineering, September 2021.

[18] Luís Cruz. Tools to Measure Software Energy Consumption from your Computer, July 2022.

[19] International Organization For Standardization. ISO 14064-1:2018 Greenhouse gases: Specification with guidance at the organization level for quantification and reporting of greenhouse gas emissions and removals, 2018.

[20] Victor Schmidt, Benoit Courty, and Amine Saboni. Codecarbon. `https://github.com/mlco2/codecarbon`, 2023.

[21] Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*, 2019.

[22] Lasse F Wolff Anthony, Benjamin Kanding, and Raghavendra Selvan. Carbontracker: Tracking and predicting the carbon footprint of training deep learning models. *arXiv preprint arXiv:2007.03051*, 2020.

[23] Semen Budennyy, Vladimir Lazarev, Nikita Zakharenko, Alexey Korovin, Olga Plosskaya, Denis Dimitrov, Vladimir Arkhipkin, Ivan Oseledets, Ivan Barsola, Ilya Egorov, et al. Eco2ai: carbon emissions tracking of machine learning models as the first step towards sustainable AI. *arXiv preprint arXiv:2208.00406*, 2022.

[24] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[25] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 220–229, 2019.

[26] Hugging Face Inc. Displaying carbon emissions for your model. `https://huggingface.co/docs/hub/model-cards-co2`, 2023.

[27] Hugging Face Inc. Autotrain: Create powerful ai models without code. `https://huggingface.co/autotrain`, 2023.

[28] Streamlit. A faster way to build and share data apps — streamlit.io. `https://streamlit.io/`.

[29] Roberto Verdecchia, June Sallou, and Luís Cruz. A systematic review of green ai. *arXiv preprint arXiv:2301.11047*, 2023.

[30] Eva García-Martín, Crefeda Faviola Rodrigues, Graham Riley, and Håkan Grahn. Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, 134:75–88, December 2019.

[31] David Patterson, Joseph Gonzalez, Urs Holzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R. So, Maud Texier, and Jeff Dean. The Carbon Footprint of Machine Learning Training Will Plateau, Then Shrink. *Computer*, 55(7):18–28, July 2022.

[32] Tim Yarally, Luís Cruz, Daniel Feitosa, June Sallou, and Arie van Deursen. Uncovering energy-efficient practices in deep learning training: Preliminary steps towards green ai, 2023.

[33] Yinlena Xu, Silverio Martínez-Fernández, Matias Martinez, and Xavier Franch. Energy Efficiency of Training Neural Network Architectures: An Empirical Study. In *Proceedings of the 56th Hawaii International Conference on System Sciences*, January 2023.

[34] Roberto Verdecchia, Luis Cruz, June Sallou, Michelle Lin, James Wickenden, and Estelle Hotellier. Data-Centric Green AI An Exploratory Empirical Study. In *2022 International Conference on ICT for Sustainability (ICT4S)*, pages 35–45, Plovdiv, Bulgaria, June 2022. IEEE.

[35] Alexander E.I Brownlee, Jason Adair, Saemundur O. Haraldsson, and John Jabbo. Exploring the Accuracy – Energy Trade-off in Machine Learning. In *2021 IEEE/ACM International Workshop on Genetic Improvement (GI)*, pages 11–18, Madrid, Spain, May 2021. IEEE.

[36] Nesrine Bannour, Sahar Ghannay, Aurélie Névéol, and Anne-Laure Ligozat. Evaluating the carbon footprint of NLP methods: a survey and analysis of existing tools. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 11–21, Virtual, November 2021. Association for Computational Linguistics.

[37] Roger Creus, Silverio Martínez-Fernández, and Xavier Franch. Which Design Decisions in AI-enabled Mobile Applications Contribute to Greener AI?, 2021.

[38] Wenxin Jiang, Nicholas Synovic, Matt Hyatt, Taylor R. Schorlemmer, Rohan Sethi, Yung-Hsiang Lu, George K. Thiruvathukal, and James C. Davis. An empirical study of pre-trained model reuse in the hugging face deep learning model registry, 2023.

[39] Commission delegated regulation (eu) 2019/2014 of 11 march 2019 supplementing regulation (eu) 2017/1369 of the european parliament and of the council with regard to energy labelling of household washing machines and household washer-dryers.

[40] Rébecca Deneckère and Gregoria Rubio. Ecosoft: Proposition of an eco-label for software sustainability. In *Advanced Information Systems Engineering Workshops: CAiSE 2020 International Workshops, Grenoble, France, June 8–12, 2020, Proceedings 32*, pages 121–132. Springer, 2020.

[41] Raphael Fischer, Matthias Jakobs, Sascha Mücke, and Katharina Morik. A unified framework for assessing energy efficiency of machine learning. In *Machine Learning and Principles and Practice of Knowledge Discovery in Databases: International Workshops of ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part I*, pages 39–54. Springer, 2023.

[42] Victor R Basili1 Gianluigi Caldiera and H Dieter Rombach. The goal question metric approach. *Encyclopedia of software engineering*, pages 528–532, 1994.

[43] Hervé Abdi. Holm's sequential bonferroni procedure. *Encyclopedia of research design*, 1(8):1–8, 2010.

[44] Volere Requirements Specification Template – Volere Requirements — volere.org. `https://www.volere.org/templates/volere-requirements-specification-template/`.

[45] Alcides Fonseca, Rick Kazman, and Patricia Lago. A manifesto for energy-aware software. *IEEE Software*, 36(6):79–82, 2019.

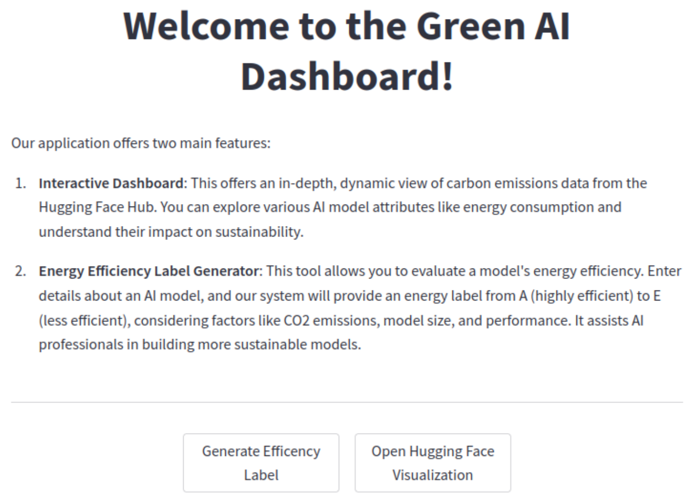
[46] NVIDIA Management Library (NVML) — developer.nvidia.com. https://developer.nvidia.com/nvidia-management-library-nvml.

[47] Running Average Power Limit Energy Reporting CVE-2020-8694,... — intel.com. https://www.intel.com/content/www/us/en/developer/articles/technical/software-security-guidance/advisory-guidance/running-average-power-limit-energy-reporting.html.

[48] Joel Castaño, Silverio Martínez-Fernández, Xavier Franch, and Justus Bogner. Exploring the Carbon Footprint of Hugging Face's ML models: A Repository Mining Study. In *Accepted in the ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2023.

A. Web-App User Manual

Our tool provides an intuitive and user-friendly interface for the assessment of machine learning models' carbon efficiency. The landing page of the tool (Figure 20) serves as the gateway to its two primary functions - the generation of energy labels and the visualization of carbon emissions data. The two buttons on this page offer users the choice between these two operations. On the left we also have a sidebar that lets you navigate through the tool's website.



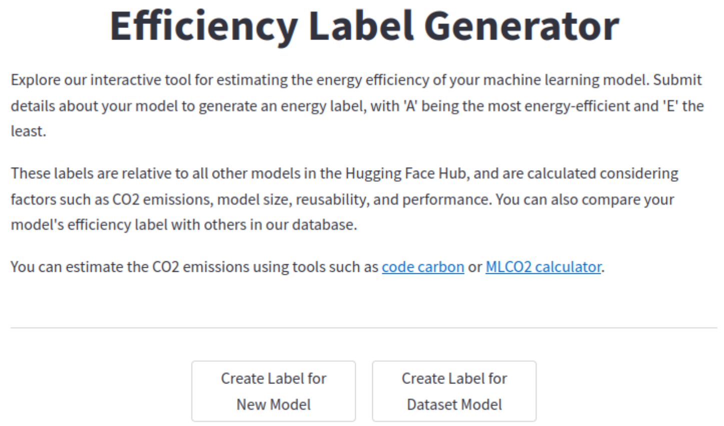
Welcome to the Green AI Dashboard!

Our application offers two main features:

1. **Interactive Dashboard**: This offers an in-depth, dynamic view of carbon emissions data from the Hugging Face Hub. You can explore various AI model attributes like energy consumption and understand their impact on sustainability.
2. **Energy Efficiency Label Generator**: This tool allows you to evaluate a model's energy efficiency. Enter details about an AI model, and our system will provide an energy label from A (highly efficient) to E (less efficient), considering factors like CO2 emissions, model size, and performance. It assists AI professionals in building more sustainable models.

Figure 20: Home landing page

Upon selecting the energy efficiency label generation option, users are directed to a new page (Figure 21) which offers a concise explanation of the process behind the energy efficiency computation. Two more buttons allow users to choose whether they want to generate an energy label for a new model or view the energy label of an existing model in the dataset.



Efficiency Label Generator

Explore our interactive tool for estimating the energy efficiency of your machine learning model. Submit details about your model to generate an energy label, with 'A' being the most energy-efficient and 'E' the least.

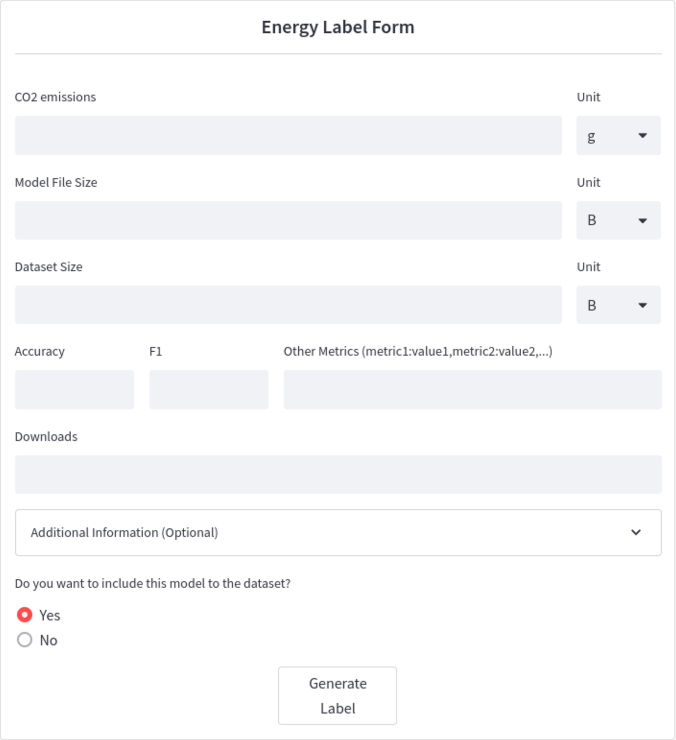
These labels are relative to all other models in the Hugging Face Hub, and are calculated considering factors such as CO2 emissions, model size, reusability, and performance. You can also compare your model's efficiency label with others in our database.

You can estimate the CO2 emissions using tools such as code carbon or MLCO2 calculator.

Figure 21: Efficiency Label Generation Page

In the case of generating an energy label for a new model, users are presented with a form (Figure 22). This form contains various fields to capture information about the model, including CO2 emissions, model file size, dataset size, the number of downloads, and performance metrics such as accuracy and F1 score. Additional contextual fields capture the name of the model, the dataset used, the computing environment (i.e., GPUs/CPUs), and geographical location. Importantly, none of these fields are mandatory. If certain information is unavailable, the system can compute the energy label without it, demonstrating the tool's adaptability to different levels of data availability.

Alternatively, for existing models in the dataset, the user can generate an energy label through a simple dropdown menu (Figure 23) followed by clicking the 'generate' button.



Energy Label Form

CO2 emissions — Unit: g

Model File Size — Unit: B

Dataset Size — Unit: B

Accuracy — F1 — Other Metrics (metric1:value1,metric2:value2,...)

Downloads

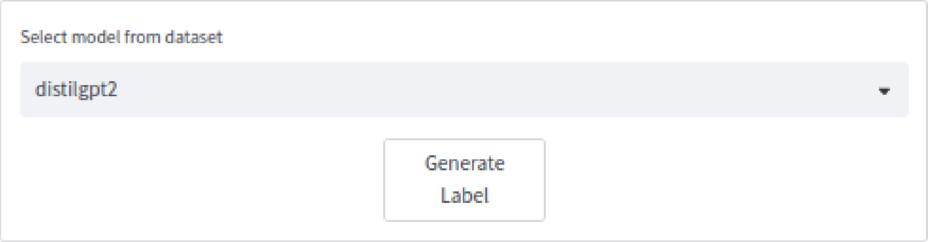
Additional Information (Optional)

Do you want to include this model to the dataset?

● Yes

○ No

Figure 22: Form on label creation



Select model from dataset: distilgpt2

Figure 23: Existing Model Label Generation

Upon submission of the form or selection from the dropdown menu, an energy label is generated (Figure 24). Designed in line with conventional energy labels, it offers a familiar visual grading system from A (highly energy efficient) to E (low energy efficiency). The label also provides a breakdown of the ratings for individual parameters, allowing users to gain a detailed understanding of their model's energy efficiency.

In this example, the reference model selected for index calculation is `distilgpt2`. As we can see, even when certain parameters aren't available for this model — in this case, the performance metrics — the carbon efficiency classification still rates the model based on the existing attributes. Here, because the carbon emissions are exceptionally high (1,492kg!) and the model size and dataset size are relatively small compared to these disproportionate emissions, the model receives a low rating with a 'D' label. However, with over a million downloads, the model's reusability is exceedingly high. This significantly contributes to the model's overall rating and prevents it from being labeled as 'E'.

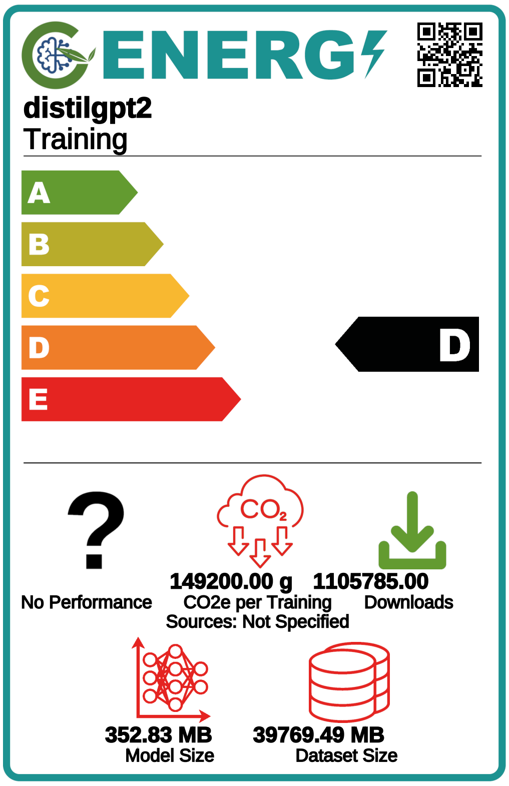


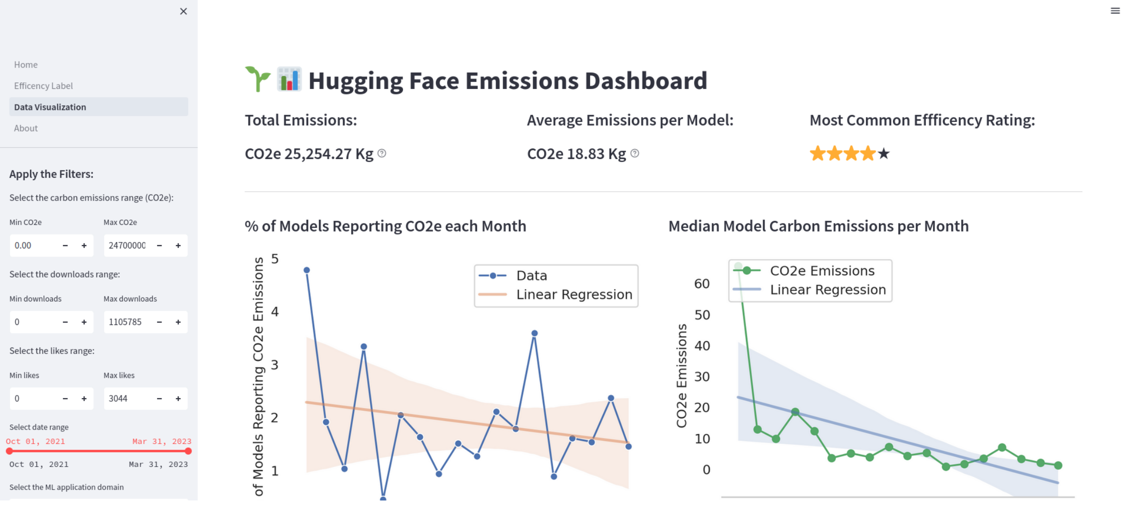
Figure 24: Energy Label Example

The data visualization page (Figure 25) provides an overview of the carbon emissions associated with various machine learning models. This function plays a crucial role in facilitating an understanding of the current state of energy efficiency across models and aids users in identifying potential areas for improvement in their own models.

This page offers a wealth of statistical data, providing the average and total carbon emissions under the current filtering configuration, alongside the most common energy efficiency rating. This data allows users to gain a quick, comprehensive understanding of the models' performance in terms of energy efficiency.

Furthermore, we enrich these statistical data with real-world connections by converting the average and total carbon emissions into equivalents of daily-life scenarios. For instance, we might present the total carbon emissions in terms of equivalent carbon emissions of a flight from Paris to London. This approach helps users to grasp the impact of carbon emissions intuitively and in a tangible way.

Users can also apply multiple filters to refine the view, such as domain, carbon emissions range, date, and library used, enabling a targeted analysis of the data. The page also showcases a series of plots illustrating the evolution of carbon emissions, the distribution of efficiency labels given the current filter configuration, and plots showing the trade-off boundaries on the final energy label based on the attributes that constitute the compound rating (Figure 26).



Apply the Filters:

Select the carbon emissions range (CO2e):

Min CO2e: 0.00

Max CO2e: 24700000

Select the downloads range:

Min downloads: 0

Max downloads: 1105785

Select the likes range:

Min likes: 0

Max likes: 3044

Select date range: Oct 01, 2021 – Mar 31, 2023

Oct 01, 2021 Mar 31, 2023

Select the ML application domain

Hugging Face Emissions Dashboard

Total Emissions: CO2e 25,254.27 Kg

Average Emissions per Model: CO2e 18.83 Kg

Most Common Effficency Rating: ★★★★★

% of Models Reporting CO2e each Month

Median Model Carbon Emissions per Month

Figure 25: Carbon Emissions Visualizations Page

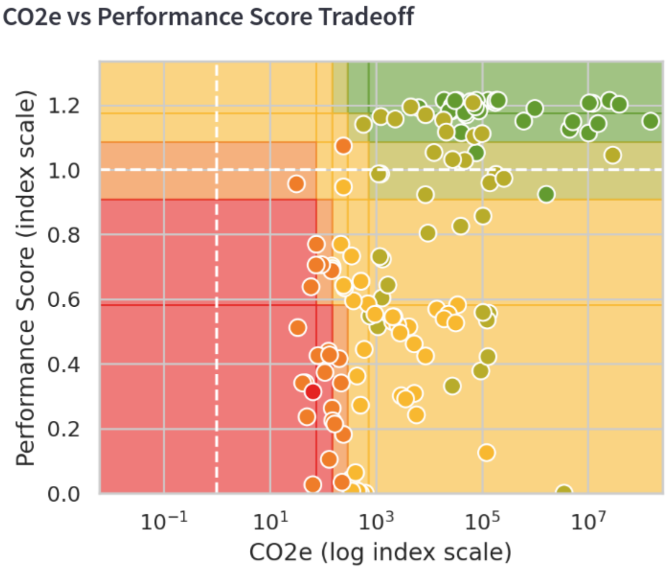

CO2e vs Performance Score Tradeoff

Figure 26: Tradeoff Plot Example