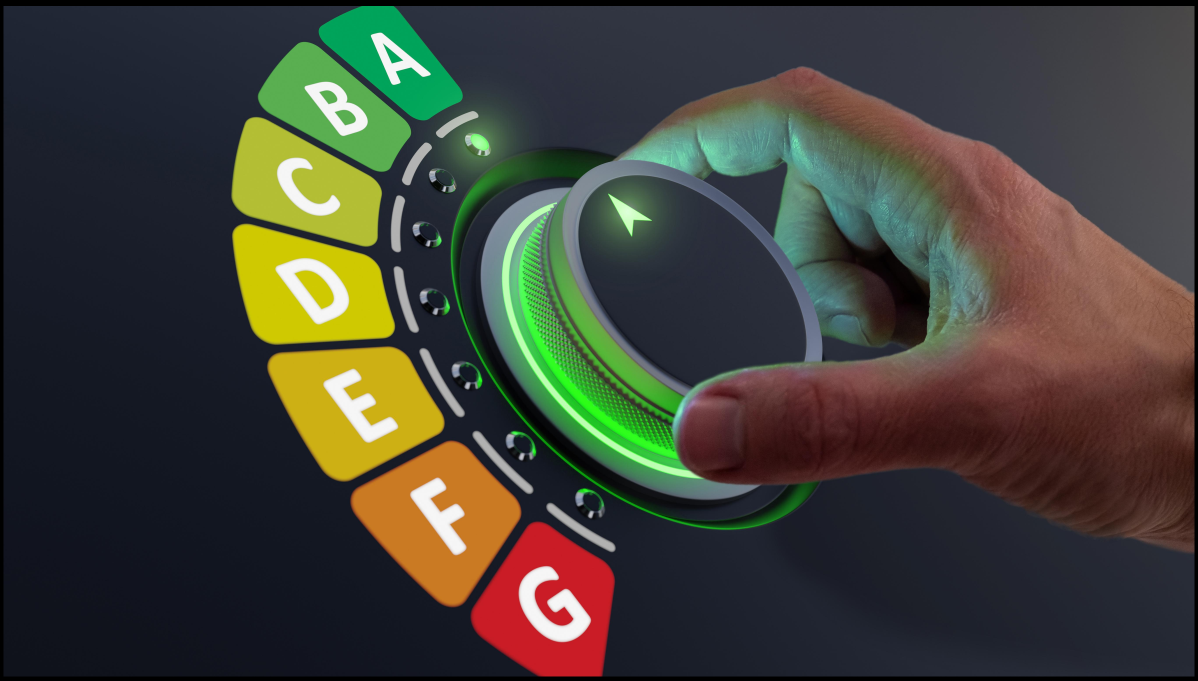
Energy Labels for Privacy-enhancing Technologies



Timothy T.R. Toonen

Layout: typeset by the author using LaTeX.
Cover illustration: Unknown artist

Energy Labels for Privacy-enhancing Technologies

Timothy T.R. Toonen
13056972

Bachelor thesis
Credits: 18 EC

Bachelor *Kunstmatige Intelligentie*

University of Amsterdam
Faculty of Science
Science Park 900
1098 XH Amsterdam

Supervisor
dr. Ana M. Oprescu, Maja H. Kirkeby

Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 900
1098 XH Amsterdam

Semester 2, 2023-2024

July 19, 2024

0.1 Abstract

In an era where digital technologies permeate every aspect of our lives, the challenges of ensuring privacy and maintaining energy efficiency have become increasingly critical. As personal data can be safeguarded using privacy enhancing tools (PETs), one thing that is often taken for granted is their power consumption. Thus this thesis proposes the integration of energy labels into PETs to foster the development of privacy-preserving and energy-efficient digital systems. Through Intel's Running Average Power Limit (RAPL) mechanism, we evaluated energy consumption of different software programs, utilizing machine learning algorithms to assign them an energy label. Naive Bayes, Neural Network, and Random Forest models are used in this survey to test their effectiveness in predicting the energy label. The present study established a base for future studies on software-based energy labeling even though it was characterized by difficulties in data collection specifically relating to datasets consisting solely of PETs.This framework can assist developers in creating more sustainable and secure digital technologies.

Contents

Chapter 1

Introduction

In an era where digital technologies invade every aspect of our daily lives, the demand for privacy-enhancing technologies(PETs) has surged. The demand is driven by increasing concerns over data privacy and security, as individuals and organizations seek protection from those who would attempt to access and misuse their data. PETs play a critical role in safeguarding personal information and maintaining the trust needed for the progression of the digital age.

Concurrently, the rapid expansion of digital infrastructure has brought attention to the energy consumption of the Information and Communication Technology(ICT) systems. As the world becomes more digitized, the carbon footprint of these systems has raised concerns due to it's substantial growth recent years. As we thrive for technological advancements, the need for energy-efficient software becomes critical to keep the growing sector sustainable and minimize it's environmental impact.

The convergence of these two domains, privacy and energy efficiency, is becoming increasingly important. As we develop and release PETs it is important these systems are not only effective for privacy protection but that they are also energy-efficient. Focusing on privacy protection and energy efficiency we can create digital solutions that are secure, sustainable, and aligned with ethical uses of technology. This thesis aims to integrate the concept of energy labels into PETs, aiming to incentive the creation of privacy-preserving yet energy-efficient digital systems.

1.1 PET's

Privacy-enhancing Technologies (PET's) can be described as technologies that enforce legal privacy principles in order to protect and enhance privacy of users of information technologies (IT) and/or data subjects (Fischer-Hbner & Berthold, 2017). These tools and techniques ensure user privacy in the digital age, with the exponential growth of online activities and data collection, privacy concerns have become paramount. PETs aim to mitigate these concerns by offering solutions that enhance the privacy and security of data, thus empowering users to have greater control over their personal information.

PETs are broadly classified into three categories. The first class comprises technologies that enforce the legal privacy principle of data minimization by minimizing or avoiding the collection and use of personal data. These PETs provide traditional privacy goals such as anonymity, unlinkability, unobservability, and pseudonymity. This minimazation of data can be achieved at both the communication and application levels, ensuring that personal data is not unnecessarily exposed.

The second class comprises technologies that enforce legal privacy requirements, such as informed consent, transparency, right-to-data subject access, purpose specification and purpose binding and security, in order to safeguard the lawful processing of personal data. These PETs ensure the lawful procession of personal data, even in situations where individuals must disclose sensitive information, for example on social networks. This class includes transparency-enhancing technologies(TETs), which are PETs that enforce or promote informed consent and transparency. These technologies allow users to understand and control how their data is used.

The last class combines the elements of the first two, integrating data minimization with legal privacy enforcement. These technologies anonymize communication and minimize data on the application level through anonymous credential protocols. Simultaneously they provide tools for privacy presentation, negotiation, and data tracking.

1.1.1 Importance of PET's

The digital age has brought with it the concept of knowledge as a primary economic resource forcing a paradigm shift in how we perceive and handle information. In present day's interconnected world, data has become an invaluable asset, being the driving force behind our continuous innovation.

The increase of personal data collection, -processing and -sharing, have led to a heightened risk of data breaches, surveillance and unauthorized access. Making PETs a vital part of modern digital infrastructure.

With its importance in our digital world the lack of PET uses is concerning. A study by Coopamootoo (Coopamootoo, 2020) attempts to show this to us by studying the individuals' privacy method in England, Germany and the US. The study shows that the majority of individuals use non-technological methods or simple PETs that are integrated in services. With the importance of our personal data this is not enough in this current day and age. According to Coopamootoo the cause is due to the lack of user knowledge and incentive for businesses to protect private data.

Following this research it shows the importance of developing more accessible and user-friendly software for individuals to defend themselves online. It is up to businesses to take up a proactive role in the creating efficient and transparent PET solutions that not only comply with legal privacy requirements but also empower users to protect their sensitive data effectively.

In summary, these technologies aim to preserve our privacy and keep our data safe from those who seek to exploit it. With the digitization of the world we are now at a point in time where the importance of PETs is at an all time high. Now we must look at these technologies more then ever.

1.2 Energy Efficiency in ICT

As the digital age progresses, the expansion of Information and Communication Technology(ICT) has been necessary for further innovation. However this rapid expansion brings with it noteworthy consequences for the environment, particularly concerning energy consumption and it's carbon footprint. The ICT sector has seen its energy demands soar due to data centers, telecommunications networks and a multitude of connected devices, resulting in increased scrutiny on it's sustainability.

Enerdata, an independent research company specializing in the analysis and forecasting of energy and climate issues, claims that the current ICT sector accounts for 6-9% of our global electricity consumption. With the sector not seeing a decrease in its expansion within the near future, it is important for us to take action now, as current projections estimate that ICT will account for 20% of our global electricity consumption by 2030.

Substantiating these claims, Lieven Eeckhout, professor at Ghent University in Belgium has also expressed his concerns, stating the ICT sector currently accounts for 2-4% of global greenhouse gas emissions. Just as with energy consumption, the gas emissions are also projected to expand up to 20% of our total global emissions by 2030.(Lieven Eeckhout, JAN. 10, 2024)

If left unchecked, the rising energy demands and greenhouse gas emissions could hinder global sustainability efforts trying to tackle climate change. Making it imperative to introduce energy-efficient solutions and technologies into the sector as soon as possible. That is why, with this thesis, we attempt to incentivize developers towards the creation of energy-efficient software.

1.3 Problem Statement

The EU energy labels addresses the issue mentioned above for other sectors, so why not for software? The introduction of the EU energy label has been influential in guiding consumers towards more energy-efficient products while still stimulating the industry and manufacturers to innovate with energy-saving technologies. The label's impact is proven to be significant, with 93% of consumers recognizing the label and 79% considering it when purchasing energy-efficient items (Special Eurobarometer 492, 2019)

With the impact of energy labels resulting in manufacturers striving to position their products into the highest energy-efficient categories to remain competitive, the introduction of labels for privacy-enhancing technologies(PETs) could similarly motivate innovation in this field as well. By providing clear information about the energy efficiency of these programs, the mandatory transparency that comes with them could motivate manufacturers to develop more energy-efficient solutions, creating more user-friendly and energy-saving technologies to preserve user data.

This thesis attempts to investigate the application of Maja Kirkeby's approach for energy consumption, to assign energy labels for software-based privacy-enhancing technologies (Kirkeby et al., 2021).

1.4 Research Question

With models where we are able to calculate the energy consumption of online systems the step towards creating accepted energy labels for these systems is not far of. If there exists a model that assigns energy labels towards them, a model could also be designed to label PETs. During this research we answer the following question:

How can machine learning algorithms be effectively utilized to automate the assignment of energy labels to privacy-enhancing technologies (PETs),and what factors affect their accuracy and reliability?

To answer this question we first answer these three questions listed below:

- How do we classify Energy Labels for software?
- What model works better for assigning Labels? Classification or Regression? And what features are the most important?
- Is it possible to label PET software using this model?

With the use of these three questions the goal is to eventually create a model that assigns energy labels to PETs and other software.

1.5 Contributions

Our research makes the following contributions:

- **Dataset of software with the same functionality**: This research compiles a comprehensive dataset of software programs that perform the same functions across multiple languages. These programs are all selected from *The Computer Language Benchmarks Game* and are available with all compiling methods, on Github(or elsewhere).
- **Energy labels for software**: The thesis attempts to create a framework for labeling software the same way appliances are labeled by the EU for their energy-efficiency. This system aims to create a method for the assignment of these labels.
- **Proposal of an Energy Labeling Model for PETs**: The research attempts to create a novel model for assigning energy labels specifically to privacy-enhancing technologies. In order to incentivize the creation of more energy-efficient PETs, by providing more transparency about their energy consumption.

1.6 Outline

n chapter 2 we describe background knowledge that assists in understanding the rest of the thesis. Chapter 3 discusses the project set-up we used, the design of the experiments and the data. The results are shown in chapter 4 and in chapter 5 we discuss these results. And lastly we conclude our research in chapter 6 also providing inspiration for future work.

Chapter 2

Background

In this chapter we shall discuss background information needed to understand the thesis. We do this by explaining concepts we further use in chapters 3, 4, 5 and 6.

2.1 Energy Measurement Methods

To assign energy labels to PET software we must first explain the possible methods to achieve our energy measurements. There are 2 main ways to measure energy consumption: Hardware-based and Software-based.

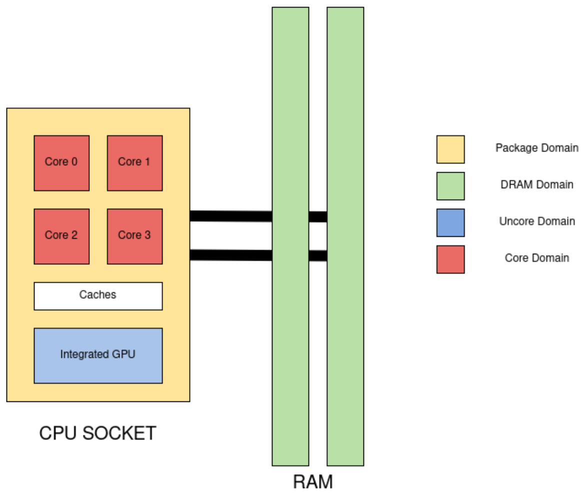
Hardware based refers to the use of physical devices and built-in hardware features to monitor and record power consumption, these methods provide highly accurate and real-time data on energy usage. As the accuracy of software-based approaches doesn't match the precision of hardware-based ones, this thesis opts for the latter. Popular hardware-based approaches include: Power Meters, On-board Energy Sensors and Running Average Power Limit(RAPL). This thesis continues further with RAPL due to it's accessibility to data and the data's high granularity.

2.1.1 Introduction to RAPL

RAPL is a built in feature for Intel and certain AMD processors that provides a mechanism for monitoring power consumption of various components within the system. RAPL allows users to programmatically obtain real-time data on the energy usage and power dissipation of the CPU package and its components, as well as the DRAM memory. Besides measuring and monitoring the power consumption

of systems, RAPL also provides the ability to limit power loss for certain components inside the processor, called power capping.
RAPL achieves this by combining automatic Dynamic Voltage-Frequency Scaling (DVFS) and clock throttling in order to keep the power consumption of the processor below a user-defined threshold. It employs an internal model of energy consumption, to estimate the average consumption over a given period, and uses DVFS to bring it as close as possible to the power cap and clock throttling to enforce it (Petoumenos et al., 2015).

Figure 2.1: RAPL domains as described by pyJoules



As seen in figure 2.1 RAPL supports multiple power domains. Each domain relays the power consumption of each domain. These domains also allow us to limit the power consumption of that domain over specified time windows, they monitor the performance impacts of limits and provide other useful information supported by the domain.(Khan et al., 2018). Figure 2.1 shows us the power hierarchy of the domains, the domains provided by RAPL to ensure accurate consumption readings are:

- **Package**: Package (PKG) domain measures the energy consumption of the entire socket. It includes the consumption of all the cores, integrated graphics and also the uncore components.
- **Power Plane 0**: Power Plane 0 (PP0) domain measures the energy consumption of all processor cores on the socket

- **Power Plane 1**: Power Plane 1 (PP1) domain measures the energy consumption of processor graphics (GPU) on the socket (desktop models only).
- **DRAM**: DRAM domain measures the energy consumption of random access memory (RAM) attached to the integrated memory controller.

Power Domain supported?					
Model	PKG	PP0	PP1	DRAM	PSys
Sandy Bridge	yes	yes	yes	No	No
Sandy Bridge-EP	yes	yes	No	yes	No
Haswell	yes	yes	yes	yes	No
Haswell-EP	yes	No	No	yes	No
Skylake	yes	yes	yes	yes	yes*

Table 2.1: Which power domains are supported by RAPL versions.

The first iteration of RAPL, available in the Sandy bridge model was released in 2010. After further advancement in CPU's by Intel their RAPL measurements also achieved a higher accuracy, dependant on it's version RAPL has supports certain domains as illustrated by 2.1 (Khan et al., 2018). While RAPL relays accurate power estimations we still encounter some challenges and limitations. We have collected a series of limitations to its use given to us by Zhang (Zhang & Hoffmann, 2014):

- **Low settling times**: Zhang realised that relative to overall runtime, RAPL achieved low settling times for their tests. These settling times indicated that RAPL may not be suitable to manage power in a system running many short-lived jobs.
- **High overshoot**: It was found that the maximum overshoot, especially for small power limits, can be significantly high. These overshoots do tend to be of short duration. How- ever, according to Zhang, even such short durations could be an issue in a large-scale distributed system if all nodes simultaneously overshoot their limits by significant amounts.
- **Dependent on application power**: RAPL's efficiency depends on both the running power and th application power. It is important for users to know that: RAPL is inefficient at low power limits but also encounters challenges at higher limits. (Zhang & Hoffmann, 2014) shows us at what levels these inefficiencies take place.

However, as explained at the start of this section RAPL provides many benefits with its accurate and accessible power measurements, making it the best option for this thesis.

2.2 Energy Labels

Energy labels are essential tools for guiding consumers towards more energy-efficient products while simultaneously encouraging manufacturers to innovate in the field of energy-saving technologies. As stated in the previous chapter the labels have been introduced into various sectors and have proven to be effective. This section hopes to introduce the concept and importance of these labels, eventually laying framework for the introduction of these labels in the coming chapter. More specifically, for the eventual introduction of energy labels for PET software.

2.2.1 Evolution of Modern Energy Labels

The energy labels have evolved significantly since it's inception, having to adapt and mitigate through technological advancements and consumer expectations, as we have moved more towards an energy-conscious society. With this evolution eventually forcing the EU to introduce new labels as of 2021.

Labeling began in the 1990s as a global effort to reduce the energy consumption to combat environmental impact. With it's first standardized introduction by the EU, these labels featured a simple rating system from A(most power-efficient) to G(least power-efficient). 20 years later these labels had evolved towards using classes above A(A+, A++, A+++) due to the rapid improvement in power efficiency, it was decided in 2017 to gradually return to a simpler scale with A as the best in order to be more understandable to consumers. Since 2021 the new labels have been introduced and for this thesis we will also be aiming towards the more recent scale from A to G. While globally the requirements and standardization for these labels vary we hope to take inspiration from the EU label and adapt it for software.

Chapter 3

Experimental Set-up

In this chapter we discuss how the experiments were set-up. We do this by first discussing the framework created for measuring the energy consumption to generate data and the method used to label our PETs. After that we explain how we clean our data. Lastly we discuss the models and metrics used for the experiments followed by how we finally evaluate them.

3.1 Consumption Measurement

To introduce energy labels we must first establish a method to measure energy consumption, in the previous chapter we have explained the concept of one such measurement technique, RAPL.

To measure energy consumption we incorporate the techniques developed by Maja Kirkeby and others, which is thoroughly documented on Github(Maja H Kirkeby & Santos, 2022). The repository provides a comprehensive framework for energy consumption measurements using RAPL. The material in the repository contains three parts:

- A setup for using Intel's RAPL.
- The statistical analysis for normal distributed data and for not normal distributed data.
- Video material explaining the entire repository and it's goals.

The Github repository contains multiple folders, with our focus on the "Energy-language-master" folder. Within this folder the setup and use of RAPL is further explained and demonstrated, allowing us to take inspiration and conduct our own

power measurements.
The Energy language folder contains multiple folders, one containing all the information RAPL needs to run his computations and three other folders: C, Haskell and Python. These three folders contain multiple algorithms written in its respective languages and a compile_all file so all measurements can be run sequentially, giving us multiple values. Kirkebey's implementation outputs a csv file when run correctly. The file contains information regarding the respective language folder when measured, giving us access to these three domains: PKG, CPU, and Time.

By following the framework provided to us we can accurately measure the power consumption of these various algorithms, providing us with data about the power consumption of different algorithms in various languages. Furthermore we are also able to run PET algorithms to measure their consumption as well.
For the implementation of your own algorithms (such as PETs) follow these steps:

- **Create folder**: Create a folder containing all the algorithms you want to measure, make sure all these algorithms are contained in their own folder.
- **Include Make**: Every folder containing an algorithm needs a Makefile, take inspiration from Makefiles in other folders. The Makefile is needed due to the compile file, explained further below.
- **Include compile_all**:The compile_all file runs every folder containing a Makefile. Using this, you can call measure on all of the algorithms you have in your folder and output a csv file containing the data.

To run RAPL an Intel processor is needed, for this we use a remote server to run the experiments on. This server is provided by UvA and contains an Intel(R) Xeon(R) CPU E5-1620 v4 @ 3.50GHz processor.

By following the methodology explained in the repository and above, we can accurately measure the power consumption of multiple algorithms sequentially providing us with data about these specific domains. This data is crucial for assigning energy labels and promoting the development of more power-efficient software.

3.2 Energy Label distribution

To assign energy labels we take inspiration from the current existing labels, attempting to mimic their methodology. As there are multiple energy labels, for example: the US, EU, Brazil, etc., the methodology varies and there is not a single standardized version recognized internationally. As we aim to mimic their methodology to assign labels, it is clear that there is no sector that is currently comparable to software. As the current sectors are all aimed towards appliances that are run for longer periods then the majority of software programs, their consumption is not easily comparable.

To achieve our own distribution of labels we look towards a study that attempts to explain the theory behind the assignment of them (Mahlia et al., 2002). This research provides us with framework to assign our own energy labels by defining thresholds that define boundaries between different energy efficiency classes. To achieve a balanced and statistically sound distribution, we opted for a normal distribution of the energy labels. To establish these thresholds we must: analyse the data, assess the distribution, determine the thresholds and then finally validate our chosen thresholds.

We first analyze the energy consumption data collected from the CLBG project using RAPL. This data contains the total energy consumed by each program during it's execution.
Then we plot the data and determine the mean and standard deviation.
Using these we calculate the thresholds for all the energy labels:

- A: below $\mu - 2\sigma$
- B: $\mu - 2\sigma$ to $\mu - \sigma$
- C: $\mu - \sigma$ to $\mu - 0.5\sigma$
- D: $\mu - 0.5\sigma$ to μ
- E: μ to $\mu + 0.5\sigma$
- F: $\mu + 0.5\sigma + \mu + \sigma$
- G: above $\mu + \sigma$

With these thresholds we aim to establish a clear and standardized process for assigning energy labels.

3.3 Data Collection and Preparation

In this research we faced a significant challenge: the absence or lack of datasets containing PETs. This lack of datasets necessitated a different approach to gathering the necessary data for our study. As stated in the first chapter we opt for datasets containing other algorithms to train the model.

To address this challenge, we turned to a well-established resource for performance benchmarking: the Computer Benchmark Game. This project provides a diverse collection of programs implemented in different languages, presenting us with a standardized way to compare programs and their consumption across different languages. As the Github uses the same source, and the game is popular across power consumption studies, such as (Couto et al., 2017), (Koedijk & Oprescu, 2022) and (St-Amour et al., 2012), we also selected this method to generate our data.

The Computer Language Benchmark Game (CLBG) project compares programming languages performance, in terms of energy- and memory consumption, by implementing solutions for multiple problems in different languages. The CLBG provides 13 computing problems in over 28 different languages providing us with a rich set of programs where we can gather extensive data on various aspects of energy consumption.

3.3.1 Collection

As explained above, we use the Github repository to run RAPL on multiple algorithms chosen from the CLBG project. As we follow the use of the repository we chose the same set of programs and languages:

- C:
  - binary-trees
  - bubblesort
  - chameneos-redux
  - fannkuch-redux
  - fasta
  - k-nucleotide
  - madelbrot
  - n-body
  - pidigits
  - regex-redux
  - reverse-complement
  - spectral-norm

- Haskell:
  - binary-trees
  - bubblesort
  - fasta
  - madelbrot
  - n-body
  - reverse-complement
  - spectral-norm
- Python:
  - binary-trees
  - fannkuch-redux
  - fasta
  - k-nucleotide
  - madelbrot
  - n-body
  - pidigits
  - regex-redux
  - reverse-complement
  - spectral-norm

These programs are all measured 50 times, providing us with enough data on programs containing these languages. When running RAPL, as stated in section 3.1, the program outputs a csv file containing three domains: PKG, CPU and Time. As these three domains are not enough to train a model we decided to add more features to the dataset to combat a model that overfits.

3.3.2 Preparation

As we need to add more features to the data we look at features that influence power consumption of algorithms, looking at what increases runtime or memory usage. Studies have researched the implementation of more energy-efficient programming techniques, showing that algorithms can be developed to consume less power when these techniques are implemented (Procaccianti et al., 2016) (Georgiou et al., 2019). As we look at the more important, and easy to distinguish features that impact the energy consumption, these are the features we select for our dataset:

- Code Language
- Compiler
- Time Complexity
- Space Complexity
- Amount of Nested Loops
- Amount of Simple Conditionals
- Amount of Complex Conditionals
- Amount of Recursive calls
- Consumption*
- Energy Label

*Consumption is the sum of the PKG and CPU. (Acar et al., 2016)

Due to the lack of trustworthy sources to uncover the Time and Space complexity they were removed, to compensate new features were added to the models, as both these complexities are high level features when discussing the computational load on devices. These features all hail from another study aiming to empirically derive meaningful threshold values for software metrics from benchmark data (Alves et al., 2010). These features are:

- Source Lines Of Code*
- Number of Functions
- Total Number of Parameters
- Cyclomatic Complexity

*Source LOC simply counts the number of lines in the code, excluding comments and blank lines.
Cyclomatic complexity measures the number of linearly independent paths through the program. It's calculated as $M = E - N + 2P$ where:

- E is the number of edges in the control flow graph.
- N is the number of nodes.
- P is the number of connected components.

For the further preparation of the data, we removed all null values and outliers. We also made sure all data-types correspond with the eventual needed types for calculations and training.

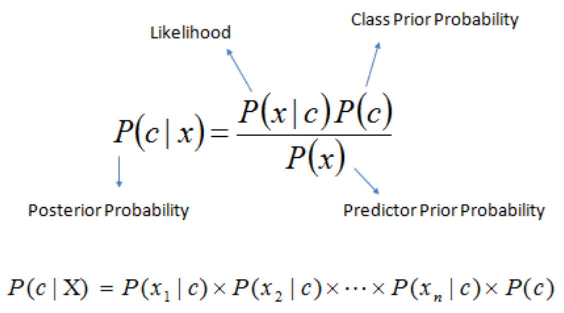
3.4 Model Design

In this section, we describe the design and implementation of the models we used to assign energy labels to PETs and other software. We employed three different models to analyze and predict the energy consumption: Random Forest Regressor, Neural Network(NN) Classification and Naive Bayes(NB) Classification.

3.4.1 NB Classification

When looking at the problem statement it resembles a classification task as you have to assign discrete energy labels to PETs. When selecting a low energy-consuming classifier we first opt for the Naive Bayes classifier. This classifier is a probabilistic classifier based on Bayes' theorem 3.1 with strong(naive) independence assumptions between features. Despite it's simplicity, Naive Bayes can be very effective for classification tasks. Due to it's simplicity it is fast to train, yet also effective with small datasets and can handle high-dimensional data.

Figure 3.1: Naive Bayes formula as described by saedsayad



$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

3.4.2 NN Classification

Continuing after our previous model we introduce another classifier with higher capabilities, but also a more computationally heavy model, Neural Network Classification. Neural networks are a set of algorithms loosely modeled after the human brain, designed to recognize patterns. The idea is to combine simple neurons, organized into multiple layers, to solve complex problems. Neural networks are able to capture complex, non-linear relationships between features and the target variable (Roßbach, 2018), making them a good fit for classification tasks where such relationships may exist. Their flexibility and adaptablility allow them to be adjusted to various types and data making them very powerfull. Furthermore, neural networks are well-suited for learning from large datasets making it scalable.

3.4.3 Random Forrest

A Random Forest is a meta estimator that fits a number of decision tree regressors on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting (Scikitlearn).

Random Forests belong to the family of decision tree algorithms. Decision Trees represent classification or regression models in a tree like structure. Random Forest works by making a multitude of decision trees with a small subset of the data, in parallel. Each individual tree in the Random Forest then gives its prediction as a vote. Then the model chooses based on an average (in case of regression) or the most votes(in case of classification).

As Random Forests can also capture non-linear relationships between features and the target variable (Roßbach, 2018), just like NNs, making it a good fit for our data, where energy consumption may not always have a simple linear relationship with the input features. They also provide a certain level of robustness to overfitting as they generalize well to unseen data. Due to there being multiple decision trees that get ensambled at the end the Random Forest handles these situations better then normal decision trees, which are more prone to overfitting (Great Learning). As Random Forests also provide feature importance, which we need to completely answer our second sub-question, it sounds like an ideal model for our project.

For this project we opt for the Random Forest classification and regression, below are listed why both are chosen and compared:

Classification:

- Discrete Label Prediction: Classification is useful when assigning categorical energy labels (e.g., A, B, C) directly based on the input features.
- Ease of Interpretation: Results are easier to interpret when discrete labels are needed.

Regression:

- Continues output: As energy consumption is a continues variable, making regression a more appropriate role than classification. The forest predicts the precise energy consumption which it maps into energy labels. If a model is needed where transparency is important this is a better approach, instead of predicting labels.
- Handling Missing Values: As Random Forest regression averages the trees it is able to handle missing values, which is beneficial for real-world datasets. In this case we removed all missing values when cleaning the data, but for a real-world model it offers a solution to handling more aggregate data.

We chose to include a regression model in our comparison due to its ability to predict not only the energy labels but also the exact energy consumption values. This provides transparency regarding the labeling of our PETs. Additionally, the regression model's robustness in handling missing values and providing continuous output makes it a valuable complement to the classification models. This comparative approach ensures that our energy labeling framework is robust, accurate, and comprehensive, capable of addressing various aspects of energy consumption prediction for PETs and other software.

3.5 Metrics

To measure the performance of all models, we used accuracy and F1-score for classification cases and R-squared (R^2) for regression. Accuracy is the ratio of correct predictions to the total number of predictions, providing an overall measure of performance. However, accuracy can be misleading on imbalanced data. For instance, if 90% of a dataset belongs to class A and 10% to class B, a model predicting all instances as class A would achieve 90% accuracy while failing to identify any class B instances. To ensure unbiased predictions, we used the F1-score, the harmonic mean of precision and recall. Precision is the ratio of true positive predictions to all positive predictions, indicating the correctness of predictions for a class, while recall is the ratio of true positive predictions to all actual instances of the class, showing how well the model identifies instances of that class. A high F1-score indicates reliable accuracy by balancing precision and recall.

For regression models, we used the R-squared (R^2) metric, which measures the proportion of variance in the dependent variable predictable from the independent variables, indicating how well the model fits the data. Additionally, one non-straightforward way to compare the results of a binary classification model with a regression model is to convert the regression continuous target values into binary by finding an optimal threshold(Acua). In this study, we categorize the target feature into energy labels while also measuring the R^2 of the continuous consumption output. This approach allows us to leverage the strengths of both classification and regression models, providing a comprehensive evaluation of model performance.

3.5.1 Model Evaluation

Besides these metrics explained above the eventual model evaluation compares the RAPL output of the PETs to the corresponding prediction outcomes resulting from the models.

Chapter 4

Results

In this chapter we show the results from our experiments. Our data, measurements, and execution of the project are available on a public Github repository[1]. The ordering of this chapter will be as followed, first we take a look at the results from the measurements and the resulting label distribution. After that we will show the results from the model experiments.

4.1 Energy Label Distribution

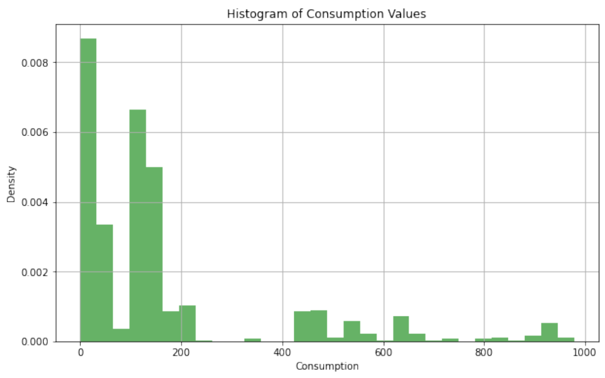
After applying the methodology described in the previous chapter, we first collected the data and afterwords we assigned energy labels to all software programs in our dataset. The distribution of these labels provides insight into the energy efficiency of the programs.

We plotted the data by consumption values, as seen in 4.1. This plot provides a visual representation of the distribution of energy consumption values across all software programs in our dataset. The histogram illustrates the density of consumption values, indicating that the majority of the programs have a lower energy consumption, with a few programs exhibiting significantly higher values.

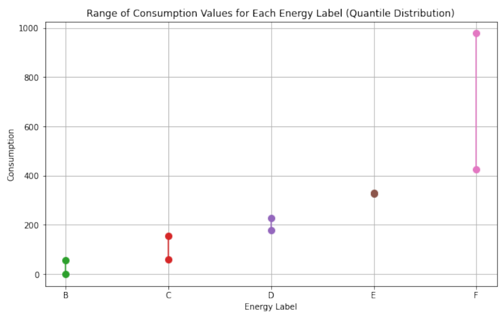
Afterwards we applied our methodology we assigned energy labels to all software programs in our dataset. In figure 4.2a it shows how our methodology assigns specific labels to specific consumption ranges. However it also shows how our method does not show any values for labels A or G. Now we refer to the second figure 4.2b where we chose a different approach towards distributing the bins, using pdcut. Using this method shows a better distribution over the energy labels, providing no

[1]https://github.com/Timothyttr/MsC-KI

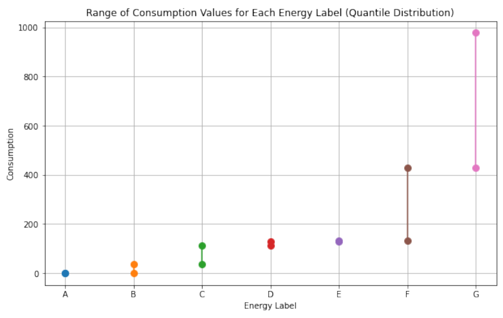
Figure 4.1: Consumption Data plotted by Density



empty labels as the previous method showed. Below is illustrated what happens when both these methods are applied to our clean data, showing the first approach loses even more labels while the second method only loses label G.



(a) Distribution of labels using our thresholds

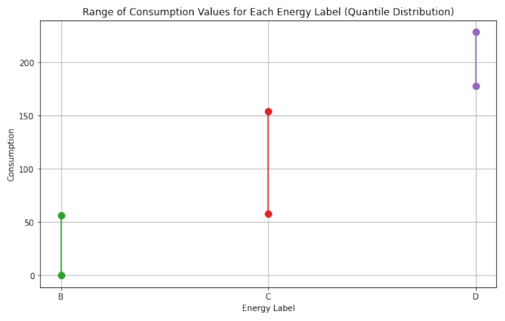


(b) Distribution of labels using pdcut

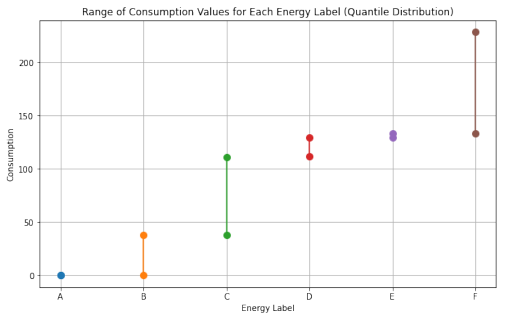
Figure 4.2: Distribution of Energy labels, on the merged dataset (without removing outliers)

4.2 Model performance

We now evaluate the four models for predicting the energy labels of PETs: Naive Bayes (NB) Classification, Neural Network (NN) Classification, Random Forest (RF) Classification and Random Forest (RF) Regression



(a) Distribution of labels using our thresholds, on cleaned data



(b) Distribution of labels using pdcut, on cleaned data

Figure 4.3: Distribution of Energy labels, on the clean dataset

Model	**Accuracy**	**f1-score**	**R^2-score**
Naive Bayes	0.512346	0.444665	
Neural Network	0.740741	0.690027	
RF Classification	0.740741	0.653515	
RF Regression	0.740741		0.903232

Table 4.1: Model Accuracy Comparison

4.2.1 Naive Bayes Classification

The NB classifier chosen for its simplicity and efficiency, scored the lowest (see 4.1). Providing an accuracy of 51.2%, not much more significant then a 50-50 guess. The f1-score also scored low at 44.4%, indicating that the model struggled with both precision and recall, resulting in unreliable predictions for energy labels.

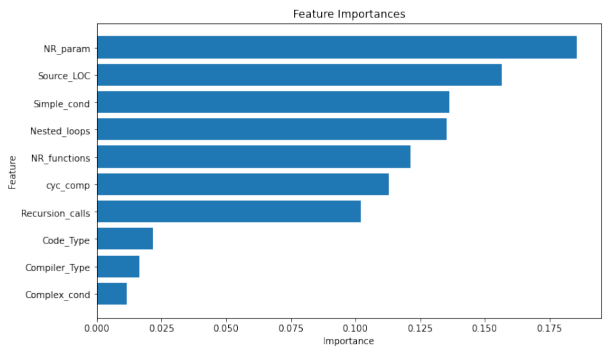
4.2.2 Neural Network Classification

The Neural Network performed, as expected, significantly better with an accuracy of 74.1% and an f1-score of 69.0%. The model performed better then the NB classifier but with an f1 score between 50-80% the performance is average.(Encord)

4.2.3 Random Forest Classification

Random Forest Classification also performed similar with an accuracy of 74.1% and an F1-score of 65.4%. While sharing the same accuracy, the lower f1-score may indicate it slightly lags behind in balancing precision and recall compared to the Neural Network. But with it being a RF we are also able to plot it's feature performance as shown in 4.4. Showing two of the most important features being the number of parameters and source lines of code, features added to compensate for the removal of complexity.

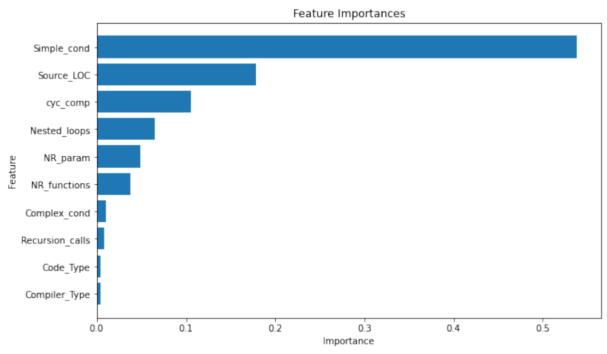
Figure 4.4: RF Classification feature Importance



4.2.4 Random Forest Regression

Instead of directly classifying the labels, the RF Regressor predicted the exact consumption values, which were then mapped to energy labels. This model achieved an R-squared (R^2) value of 90.3, indicating a good fit for predicting continuous values. When mapped to labels, it provided an accuracy of 74.1% competing with the two previously mentioned models. Just as the previous model we can also provide the feature importance of this one 4.5, showing the different models assign different importance to the features, as simple conditions is the most important feature in this case.

Figure 4.5: RF Regression feature Importance



Chapter 5

Discussion

In this chapter we delve into a comprehensive discussion of the findings from our study. We start by interpreting the results, exploring the implications of our model performance and discuss the strengths and limitations of our approach. Finally we will discuss some additional results and challenges we encountered.

5.1 Interpretation of Labeling Results

As briefly touched upon in the previous chapter section 4.1, our initial method for the distribution of energy labels did not achieve our goal. The method we wanted to use, inspired by the normal distribution, did not fit our data as we had hoped. By basing our threshold on the mean and standard deviation the first and last labels received 0 programs. This is why we opted for a more uniformly distribution, using pdcut to determine the bins used for label assignment, this seemed to fit our data more accurately then the predetermined thresholds. Resulting in new thresholds (given in joules):

- A: 0.000 to 0.15
- B: 0.15 to 37.36
- C: 37.37 to 110.84
- D: 110.84 to 129.23
- E: 129.23 to 132.99
- F: 133.00 to 428.69
- G: 428.69 and up

Later in this chapter more about the data is discussed.

5.2 Interpretation of Model Results

The results indicated that more complex models, such as Neural Networks and Random Forests, significantly outperform simpler models like the Naive Bayes, as was expected. It was hoped that the Naive Bayes model could achieve a higher accuracy, as it is a simple model and the computational load is relatively low compared to the other models.

The Neural Network classifier had the highest f1-score providing the same accuracy as the other models, but showed its superiority in balancing precision and recall effectively, when compared to the other classifiers. However, the regression model provided an impressive R^2 value, indicating it is a good fit for predicting continues values. The performance of the Neural Network classifier and Random Forest regressor show that these algorithms are both suited for the task of assigning energy labels. Even so, where the Neural Network is only capable of predicting the energy labels, the Random Forest regressor is capable of also predicting the actual power consumption of the programs. As we aim to provide transparent models the prediction of the energy consumption is crucial. Along with its transparency our regression algorithm also requires much less input preparation, as they can handle binary features, categorical features as well as numerical features and there is no need for feature normalization. Random Forests are quick to train and to optimize according to their hyperparameters. Thus, the computational cost and time of training a Random Forest are comparatively low when compared to the Neural Network Classifier(Roßbach, 2018).

5.3 (Data) Challenges

As stated before, RAPL needs an Intell processor to function. To be able to run this program, an SNE server was provided. The goal was to be able to collect data through the server to increase the dataset, while also being able to run PETs and gather a small dataset of their consumption as well, ultimately leading to a way to evaluate our model by comparing the PET consumption to the model predicted values. However using the sever was challenging, when attempting to run it would output csv's with half of the values missing, or in some cases it would simply time out my connection to the server. Due to these problems the data used was procured by an old computer possessing an Intell Core i5-243M CPU @ 2.40GHZ x 4 processor. Resulting in a different dataset then hoped, with the computer also only being able to run RAPL 30 times instead of 50. Due to it being an older computer and not being in a controlled environment, as I could not be present, the data cannot be deemed as completely sound. The consumption and time measurements when the program was run could be influenced by the age and speed of the computer.

Chapter 6

Conclusion

We attempted to create a model to assign energy labels for software-based Privacy-enhancing Technologies. We aimed to provide a method that could guide software developers in creating more energy-efficient software solutions. Using RAPL we collected data and using three machine learning algorithms we attempted to predict the necessary energy labels.

Before attempting to answer our research question, we first address the three sub-questions, starting with: **How do we classify Energy Labels?** For assigning labels, the consumption values were calculated and then divided into quantiles by following a uniform distribution to determine thresholds. However by generating more data it is possible the determination of thresholds could follow another distribution.

What model works better for assigning Labels? Classification or Regression? And what features are the most important? we ran multiple models attempting to find the best fit for this project. Multiple classification models were used with the best being a Neural Network model, while there was only one regression model testing, but it still achieved the same accuracy as the Neural Network. As the regression model provides more information about the total consumption it is deemed a better fit for this project, being able to accurately predict the consumption as well as being more energy-efficient. The feature importance differed according to which model was used but with both classification and regression these 5 were the most important:

- Number of parameters
- Number of Simple Conditionals
- Source Lines of Code

- Number of nested loops
- Number of functions

Is it possible to label PET software using this model? As we were unable to directly compare the prediction of the model with PET software due to technical issues this answer can not be answered in its entirety. The framework provided in this project should be capable of assigning energy labels for PETs, but as this could not be evaluated this statement can not be verified.

Finally we attempt to answer our main research question **How can machine learning algorithms be effectively utilized to automate the assignment of energy labels to privacy-enhancing technologies (PETs),and what factors affect their accuracy and reliability?** By the implementation of Random Forest regression it is possible to predict the energy consumption and labels of software, unfortunately we lack the tests and results to claim if energy consumption by PET software could be predicted with this model. However as this model relies on features that are related are important for every type of program, it should be feasible to assign energy labels for PETs using this model.

In conclusion while it was not possible to label PET software using the models developed in this study, the methodology and framework established hold significant potential. By addressing the identified limitations and continuing to refine the models and data collection methods, we can achieve the goal of promoting energy-efficient software development. This research contributes to the broader objective of sustainability in the ICT sector, aligning technological advancements with environmental responsibility.

6.1 Future work

For future projects on this subject it is recommended to make sure the data you have collected does come from a controlled environment. In a situation where you are able to run the measurements in this way, collect a larger dataset by running all the CLBG programs provided through RAPL. By running all these programs it could be possible that the code language and compilertype could become more important features due to the larger set of collected programs with different languages.

Another recommendation is to attempt to create a PET dataset which is similarly built to the CLBG one we use. As the lack of data has caused the project to center more around software then the original goal of Privacy-enhancing Technologies. Possibly, using the PET datasets instead of the CLBG programs future projects could base the dataset around the techniques used by PETs instead of the structure of the code, providing a more PET based model.

Bibliography

Acar, H., Alptekin, G. I., Gelas, J.-P., & Ghodous, P. (2016). The impact of source code in software on power consumption. *International Journal of Electronic Business Management*, *14*, 42–52.

Alves, T. L., Ypma, C., & Visser, J. (2010). Deriving metric thresholds from benchmark data. *2010 IEEE international conference on software maintenance*, 1–10.

Coopamootoo, K. P. (2020). Usage patterns of privacy-enhancing technologies. *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 1371–1390.

Couto, M., Pereira, R., Ribeiro, F., Rua, R., & Saraiva, J. (2017). Towards a green ranking for programming languages. *Proceedings of the 21st Brazilian Symposium on Programming Languages*, 1–8.

Fischer-Hbner, S., & Berthold, S. (2017). Privacy-enhancing technologies. In *Computer and information security handbook* (pp. 759–778). Elsevier.

Georgiou, S., Rizou, S., & Spinellis, D. (2019). Software development lifecycle for energy efficiency: Techniques and tools. *ACM Comput. Surv.*, *52*(4). https://doi.org/10.1145/3337773

Khan, K., Hirki, M., Niemi, T., Nurminen, J., & Ou, Z. (2018). Rapl in action: Experiences in using rapl for power measurements. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, *3*. https://doi.org/10.1145/3177754

Kirkeby, M. H., Gallagher, J. P., & Thomsen, B. (2021). An approach to estimating energy consumption of web-based it systems. *CERCIRAS Workshop-01: Abstracts*, 18.

Koedijk, L., & Oprescu, A. (2022). Finding significant differences in the energy consumption when comparing programming languages and programs. *2022 International Conference on ICT for Sustainability (ICT4S)*, 1–12. https://doi.org/10.1109/ICT4S55073.2022.00012

Mahlia, T., Masjuki, H., & Choudhury, I. (2002). Theory of energy efficiency standards and labels. *Energy Conversion and Management*, *43*(6), 743–761.

Maja H Kirkeby, J. P. F., & Santos, B. (2022). Energysoftware-cerciras. https://github.com/SustainableSoftware/EnergySoftware-CERCIRAS

Petoumenos, P., Mukhanov, L., Wang, Z., Leather, H., & Nikolopoulos, D. (2015). Power capping: What works, what does not. *21st IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, 525–534. https://doi.org/10.1109/ICPADS.2015.72

Procaccianti, G., Fernández, H., & Lago, P. (2016). Empirical evaluation of two best practices for energy-efficient software development. *Journal of Systems and Software*, *117*, 185–198. https://doi.org/https://doi.org/10.1016/j.jss.2016.02.035

Roßbach, P. (2018). Neural networks vs. random forests–does it always have to be deep learning. *Germany: Frankfurt School of Finance and Management.*

St-Amour, V., Tobin-Hochstadt, S., & Felleisen, M. (2012). Optimization coaching: Optimizers learn to communicate with programmers. *Proceedings of the ACM international conference on Object oriented programming systems languages and applications*, 163–178.

Zhang, H., & Hoffmann, H. (2014). A quantitative evaluation of the rapl power control system. https://api.semanticscholar.org/CorpusID:13950838