


Streamlining AI: Techniques for Efficient Machine Learning Model Selection

Daniel Schönle 
IDACUS Insitute
Furtwangen University
Furtwangen, Germany
schonledanielhfu@gmail.com

Christoph Reich
IDACUS Insitute
Furtwangen University
Furtwangen, Germany
christoph.reich@hs-furtwangen.de

Djaffar Ould Abdeslam
Institut IRIMAS
Université de Haute Alsace
Mulhouse, France
djafar.ould-abdeslam@uha.fr

Abstract—As machine learning (ML) systems become more ubiquitous, their resource requirements and associated financial burdens increase, highlighting the need to optimise energy consumption and costs to meet stakeholder expectations. While quality metrics for predictive ML models are well established, efficiency metrics are less commonly addressed. We present a comprehensive framework for evaluating efficiency metrics that facilitates the comparison of different types of efficiency. A novel efficiency metric approach (Compact Efficiency Metric) is proposed that considers resource usage, computational effort, and runtime in addition to prediction quality. Implementations for specific focus areas have been developed, such as the Quality-Focused Compact Efficiency Metric (QCO). This work also introduces a Pareto-based methodology for selecting ML models with an emphasis on efficiency. The QCO metric has undergone rigorous testing to validate its applicability, plausibility, and ability to adjust for variations in dataset size and hosting environment performance. This QCO metric has been applied to two datasets and calculated for a wide range of ML models. In particular, our metric identified an efficient solution when determining the optimal sequence length for transformer-based models. The results from Pareto-based selection were congruent with those derived from the QCO metric, providing a viable approach for pre-selecting preferred solutions. The framework enables stakeholders to make informed decisions concerning the utilisation and design of ML models, thereby ensuring environmental responsibility and cost-effectiveness.

Index Terms—*machine learning; nlp; efficiency; metric; software performance; automl.*

I. INTRODUCTION

In past decades, the imperative for computational efficiency was determined by the constraints of then-available technology, compelling the optimal use of limited hardware resources. With advancements in computing capabilities, particularly in machine learning (ML), focus has gradually shifted towards augmenting the quality of predictions, often at the expense of operational efficiency. The advent of large language models (LLMs) has been pivotal, marking a transformative phase in computational tasks and signalling their growing dominance in human-computer interactions. This shift coincides with the integration of ML in various environments,

including edge computing, where resource limitations necessitate a critical reconsideration of computational approaches from perspectives of green computing and sustainability. Utilising resource-intensive techniques, such as transformer-based embeddings or LLM, introduces financial and environmental challenges, emphasising the necessity for enhanced computational efficiency. In response to these challenges, the oblique objective of this publication is to improve the efficiency of ML operations. By introducing a set of comprehensive metrics, this work aims to measure the efficiency of ML models, thereby facilitating their sustainable and economically viable deployment across different platforms [2].

The focus of ML research has been on improving model quality, for which a number of metrics are available. Research on effective ML lacks standardised and comprehensive efficiency metrics. To ensure reproducibility and facilitate comparison of results, best practices in ML research typically include detailed descriptions of experiments, including datasets, pre-processing steps, machine learning techniques, hyperparameters, and hardware setups [3]. The lack of dataset-agnostic procedures and the absence of 'golden standard' datasets pose challenges in achieving accurate reproducibility and fair comparisons in natural language processing (NLP) [4]. Furthermore, assessing the impact of innovations in ML process steps, such as improved pre-processing, on prediction quality is complicated by the influence of other ML steps.

The delicate balance between complexity and outcome is often overlooked in research efforts to utilise all available resources to reduce time-to-solution. Evaluating complexity in time and space is subordinated to finding the best model or process. Numrich stated [5]: "Increasing productivity by minimising the total time-to solution is a somewhat ill-defined statement of the problem. We propose an alternative statement: at each moment in time, use the resources available optimally to accomplish a mission within imposed constraints." In the context of ML research, it is important to establish metrics that address efficiency concerns alongside prediction quality.

We propose to fill the existing gap by introducing novel metrics for measuring the efficiency of machine learning models.

Note: This paper is a revised and extended version of [1].

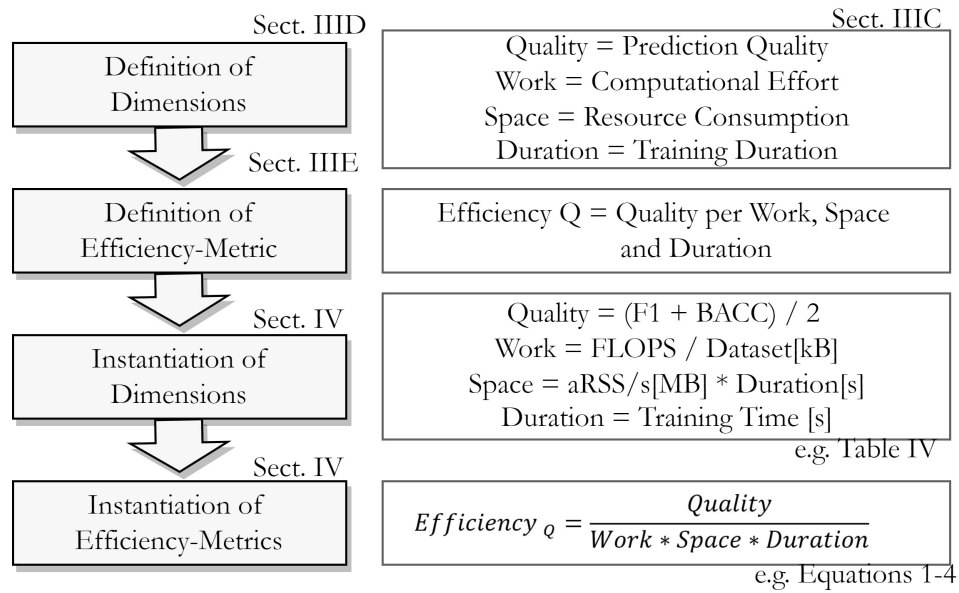


Fig. 1. Development of Quality Focused Efficiency Metric.

By incorporating resource consumption, computational effort, and runtime considerations into our efficiency metrics, we aim to provide a holistic perspective on the actual efficiency of ML models. We demonstrate the process of defining a quality-focused efficiency metric (Figure 1) and present the Quality CCompact (QCO) Efficiency Metric, detailed in Equations (4) and (5).

We acknowledge the significance of dataset-agnostic evaluation and propose solutions to overcome this challenge. Furthermore, we demonstrate the benefits of our metric for evaluating hyperparameter tuning. To facilitate the measurements, we have developed a tool to monitor the ML computing processes. Additionally, we introduce a Pareto-based approach to simplify selecting the most suitable ML setup. The aim is to enable researchers and practitioners to make informed decisions that prioritize prediction quality and efficiency, thus advancing the field of machine learning towards sustainable, green, and economically feasible solutions.

A. Summary of Contributions

The contributions of this research are as follows:

Efficiency Definition Following Physical Units. Defines machine learning efficiency using a framework analogous to physical units, providing a standardized approach to quantify and compare machine learning efficiency.

Definition of Efficiency Dimensions: Establishes key metrics such as quality, speed, and resource utilization.

Foundational Efficiency Metric: Introduces a basic efficiency model.

Focused Efficiency Metrics: Extends the Efficiency Metric to address specific aspects like computational cost or quality, allowing for targeted optimizations.

Definition of Instantiation Process: Applies the efficiency metrics on use cases, including selecting appropriate

measurements, data transformation techniques, and determining validity ranges.

Exemplary Instantiation for QCO: Demonstrates how to implement an efficiency metric specifically focusing on quality.

Capturing Tool Observe: Develops a user-friendly tool to automatically capture data needed to evaluate the efficiency of machine learning algorithms, simplifying performance analysis.

Evaluation of Efficiency Metric: Discusses methods to assess the validity of efficiency metrics by comparing them to expert and Pareto solutions.

B. Comparison of Approaches

Table I compares traditional efficiency approaches and the solutions provided by the Compact Efficiency Metric, with particular emphasis on the advantages of the former. The comparison demonstrates the enhanced flexibility and applicability of the Compact Efficiency Metric across various computational contexts. It addresses the limitations of traditional methods by providing a structured, measurable framework for evaluating machine learning efficiency.

C. Outline

The structure of the paper is as follows: Section II (State of the Art) covers the research on efficiency types, metrics in ML, and the Pareto approach. In Section III, the efficiency metric is presented by elaborating on its objectives, followed by the theoretical foundations of *efficiency dimensions* and concepts, and finally, the definition of the efficiency metrics. In the subsequent Section IV, the metric for quality-focused efficiency is defined, adhering to a specified protocol. The score equations for QCO_F are presented, accompanied by a brief explanation of its usage. The Evaluation Section V uses

TABLE I
COMPARISON OF APPROACHES

Approach	Downside	Compact Efficiency Metric
Computational Cost	Complexity based on theoretical calculation	Empirical measurements
Computational Cost for Deep Learning	Model parameter based theoretical calculation	Empirical measurements
Resource Efficiency	Objective is a two-sided optimization of software and hardware	Resource consumption is emphasized, aiming to balance multiple efficiency dimensions
Efficiency Comparison	Restricted to comparing methods, not scoring	Provides comparable scores with defined validity ranges
Efficiency Metrics	Optimized for high-performance computing components with specific measurements	Measurements applicable on standard hardware and software
Pareto Efficiency	Multi-dimension optimization without dimension weights	Allows for weights to be assigned to each efficiency dimension

two experiments to assess the performance of the metric. The results obtained are discussed in detail in Section VI, leading to the presentation of the conclusion (Section VII).

II. STATE OF THE ART

This section provides an overview of the current state of the art and serves as a background chapter. Approaches that address computational cost as a method, with the aim of predicting both computational and financial costs, are considered first. Next, resource efficiency approaches are reviewed, aiming to find algorithms that operate cost-effectively. Additionally, generic approaches to efficiency metrics are examined. Finally, the concept of Pareto efficiency is introduced.

A. Computational Cost

Computational Cost or efficiency is based on the computational effort. Most statistical ML algorithms can be addressed, and their time complexity or space requirements can be calculated. For example, the time complexity of gradient descent is $O(ndk)$, where d is the number of features and n is the number of rows. In the context of transformer-based approaches, the number of operations for multi-head attention can be calculated as $n^2d + nd^2$, where n is the sequence length and d is the depth [6]. Translating these statistical calculations into real training times is challenging due to numerous optimisations of modern CPUs and GPUs that change the type of computation and the number of operations [7][8][9]. The approach defines work and duration dimensions based on actual measurements.

Computational Cost for Deep Learning is specific to deep learning due to its reliance on complex neural network architectures, which complicates the direct computation of complexity. Several approaches attempt to predict complexity, such as the model proposed by Li et al. [10], which introduces two classes of prediction models for distributed SGD. The use of profiling information in this approach is similar to the presented method, but it has limited validity for deep learning optimised with distributed SGD.

Resource Efficiency is essential for deep learning, where hardware requirements differ from statistical ML and constantly evolve. The research aims to adapt deep learning to

specific hardware. Yang et al. [11] developed a method to bridge this gap, focusing on computing the model locally near the sensor. In HPC, research such as Performance Metrics based on computational action (Numrich [5]) optimises the use of hardware. Resource efficiency focuses primarily on the optimal hardware usage of specific algorithms, ignoring algorithm complexity or runtime. The efficiency definition addresses this aspect to provide comprehensive statements about the entire ML task.

B. Efficiency

Efficiency Comparison plays a role in evaluating novel approaches. For instance, Thomson et al. [12] present an optimisation for machine learning-based compilers that focuses on process speedup while overlooking the impact on resource consumption. Fischer et al. [13] propose a framework for evaluating the energy efficiency of ML without considering prediction performance. Kumar, Goyal & Varma [14] develop ML with a small footprint and compare efficiency based on model size, prediction quality, prediction time and prediction energy. Discussions of the novel approach primarily revolve around individual measurements, lacking an overall efficiency comparison. In contrast, Huang et al. [15] discuss the selection of an object detection architecture in terms of efficiency, defining it as a speed/memory/accuracy trade-off and evaluating it through two-dimensional trade-off curves. The proposed efficiency metric would provide a balanced and meaningful score for evaluating [14] and [15].

Efficiency metrics were 'invented' for HPC research, which deals with highly scaled hardware systems and highly specialised applications, making efficiency statements easier to derive and crucial. The difficulties have been recognised and discussed from an early stage [16] - to philosophical considerations [17]. Numrich of Cray Research developed an approach based on physical laws [18] [5], which inspired the proposed metric based on dimensions reflecting components of a physical law.

C. Pareto efficiency

In the field of machine learning, Pareto efficiency has emerged as a key criterion for evaluating and optimising algorithms under the constraints of multiple competing objectives.

Pareto efficiency delineates an optimal state where no single objective can be improved without simultaneously degrading another, thus embodying the essence of trade-offs inherent in decision-making processes.

Pareto efficiency is a critical factor in multi-objective optimization (MOO) [19] [20], where ML models are optimized across various dimensions, including accuracy, computational complexity, and energy consumption [21] [22]. This requires a departure from traditional single-objective optimization paradigms towards more nuanced approaches that can navigate the complex trade-offs among competing objectives. Several algorithms have been proposed to address MOO in ML. These include evolutionary algorithms [23] [24], swarm intelligence [25], and gradient-based techniques [26], each offering different mechanisms for approximating the Pareto front, which is the set of all Pareto efficient solutions.

Pareto efficiency has become a critical criterion for evaluating and optimising algorithms in the evolving ML landscape, where multiple competing objectives must be considered [27]. Pareto efficiency is a concept from economics that describes an optimal state in which it is impossible to improve one objective without degrading another, thus embodying the essence of trade-offs inherent in decision-making processes [28]. This chapter explains the applications of Pareto efficiency in machine learning, including its significance, evaluation methods, and implications for algorithm design and evaluation.

Recent advancements have led to the integration of Pareto efficiency with deep learning, particularly in areas such as neural architecture search (NAS) [29]. The objective is to discover optimal network architectures that balance performance with resource constraints. Studies have employed Pareto-based approaches to navigate the search space efficiently, identifying architectures that offer optimal trade-offs between accuracy and computational cost [30].

Furthermore, Pareto efficiency can be applied to optimisation and model evaluation and selection [31]. In this context, Pareto fronts serve as a valuable tool for evaluating the performance of various models, allowing practitioners to make informed decisions based on a comprehensive understanding of the trade-offs involved. This approach has proven advantageous in fields where performance is multifaceted and cannot be captured by a single metric. For instance, in recommender systems, accuracy, diversity, and novelty may all be of concern [32].

The investigation of Pareto efficiency in machine learning also poses fundamental questions regarding the nature of optimality and the objectives of optimization itself. It urges the community to reassess current practices and create new theoretical frameworks and algorithms to more accurately depict and navigate the intricate trade-off landscapes typical of real-world issues. In conclusion, integrating Pareto efficiency into machine learning research and practice represents a significant paradigm shift, promoting a more comprehensive and nuanced approach to optimization. As the field progresses, further exploration of Pareto-based methods is expected to provide significant insights into the design, evaluation, and application

of machine learning systems, ultimately advancing the frontier of what is achievable in the domain.

III. EFFICIENCY METRIC PROPOSAL

This proposal encompasses two integral parts: the development of abstract efficiency metrics and the definition of the quality-focused efficiency metric. The objectives, limitations, and use cases of efficiency in machine learning (ML) are introduced, and basic efficiency types based on trade-off relationships are established. Drawing inspiration from the laws of physics, efficiency is defined using *efficiency dimensions* for quality, work, space, load, and duration of the ML procedure, with each dimension comprising measurements of the ML process. Two types of metrics are presented: the *efficiency vector*, which provides insight into the raw strengths and weaknesses of the ML process in terms of efficiency, and the *focused efficiency scores*, which are designed for ease of interpretation. A defined procedure is employed to adjust dimensional weights and perform sophisticated measurement smoothing to enhance the significance of scores. For example, the metric equation for quality-focused efficiency is outlined, and a metric definition protocol is proposed to achieve metric validity. This protocol is applied to define the quality-focused efficiency metric, including the definition of the equation for score calculation, selecting appropriate measurements, smoothing measurement values, and elaborating dimension weights.

TABLE II
LIMITATIONS OF QUALITY COMPACT (QCO) METRIC

Aspect	Category or Requirement	Evaluated	Compensation
Dataset	Dataset Independent	Labelled Text Samples	Dataset Size ¹
ML-Task	Complete or partial Pipeline	Text Classification Pipeline, NLP-Tasks ⁴	No ²
ML-Techniques	Technique Independent	Statistical and Deep Learning Techniques	No ³
System Environment	Access to Measurements of Duration, Calculation Steps, Resource Consumption of ML-Task	Virtualised Host	No ³
Host-Setup	Host-Setup Technique Independent	Non-HPC, Non-GPU	Compute Speed ¹

(1) Efficiency remains consistent regardless of value.

(2) Scores from different tasks are not comparable.

(3) Provides comparable efficiency scores per ML technique.

(4) untested

A. Objectives & Limitations

The methodology aims to ensure applicability across various machine learning techniques. It is designed to function on standard hosting setups, ensuring that measurements reflect balanced efficiency irrespective of differing host or system

environments. Table II lists the requirements and the evaluated selection for each relevant aspect. The 'Compensation' column provides an overview of the metric compensation for variability, including dataset size and host setup performance. Measurements are conducted during the training phase of the classification model. Valid ranges for measurements must be aligned to the expected values. The validity ranges of the QCO Instance for the proof of concept are listed in Table III. Smoothing techniques are employed to standardise the validity of the data to ensure these measurements fall within comparable ranges.

TABLE III
VALIDITY RANGE FOR QCO INSTANCE

Aspect	Validity
Dataset	Size <256 MB
Training Duration	<48h
Calculation Amount	63P FLOPS, 800M Minor Page Faults
Resource Consumption	RAM <128GB

The application of the ML process is contingent upon the establishment of objectives and constraints. The following use cases are subject to specific efficiency requirements.

- 1) Effects of changes in the ML process: The effects of different techniques, such as pre-processing techniques, need to be measured [33].
- 2) Select ML technique by efficiency: Identify the ML technique that achieves high classification quality while minimising the use of computational resources [11].
- 3) ML technique for limited resources or private data: Select a Whitebox ML technique suitable for local model training [34].
- 4) Parameter optimisation: Effectiveness as a cost function in the optimisation of hyperparameters or setups [35].
- 5) Performance comparison: Compare the performance of an ML technique on different host setups to evaluate ML efficiency [36].
- 6) Predicting computational costs: Predicting the cost by predicting computational effectiveness of an ML technique in a production setup [37].

B. Dimensions

The efficiency approach may vary between applications but relies on similar elements. All concepts consider the trade-off between model performance and resource consumption during training or inference. The efficiency metrics focus on the following key components:

Accuracy or Performance. The efficiency of a machine learning model depends on its accuracy or performance. Several standard metrics can be employed to assess this, including accuracy, precision, recall, F1 score and area under the ROC curve (AUROC), depending on the task at hand.

Resource Utilisation. The efficiency should be evaluated regarding the resources consumed during the training or inference process. These resources include computational resources

such as CPU, GPU, or memory usage. An efficient model should aim to minimise the utilisation of resources.

Relative Resource Utilisation. The load imposed on the host by the machine learning process provides a means of measuring the relative utilisation of hardware resources. A higher load indicates a more efficient use of available resources, as fewer resources are left unused.

Computational Effort. The efficiency of ML processes is contingent upon the complexity of the ML process itself, as well as the amount of computation required to train the model or to compute a result for inference. In order to enhance the efficiency of ML processes, it is necessary to minimise the computational effort.

Training Duration. The definition of efficiency encompasses the time required to train the machine learning model. Faster training times may be advantageous, particularly in instances where models must be trained frequently or where time constraints exist.

Inference Latency. In the context of models deployed in real-time or interactive applications, the time taken to make predictions or perform inference is critical. Low inference latency or fast response times can be important efficiency metrics in such cases.

In the context of cost-effectiveness in machine learning research, different dimensions or base units are considered. The need to define base units, such as distance and power, which can be used to define efficiency, has been discussed by Numrich [38]. The CO-Metric uses abstract dimensions that allow for customisation through flexible adaptation. The proposed efficiency metric uses the following *efficiency dimensions*, with a description of valid measurements:

Quality. (Or Performance) The machine learning model should achieve the desired level of accuracy as a performance indicator for addressing the given task or problem. This accuracy can be measured using appropriate evaluation metrics tailored to the specific task, including accuracy, precision, recall, F1-Score, or AUROC. The scores should compensate for any unbalanced datasets [6].

Work. (Or Computational Effort, Computational Complexity) The number of computational operations, such as matrix multiplications, gradient computations, data transformations, and the usage of computational cache (e.g., CPU L1-Cache). The theoretical amount of work can be calculated by applying the theory of computational complexity. The actual workload differs due to optimisation at the software and hardware level [7]–[9]. The measurement shall count generated and processed compute steps; optionally data transfers through memory and network. Computational steps can be counted directly (floating point operations or instructions) [6] or indirectly by measuring side-effects of computation, e.g., memory management activity.

Load. (Or Relative Resource Consumption) The relative host usage metric reflects the *degree* to which all available host resources are utilised. This encompasses the relative

usage of compute units (CPU and GPU cores), as well as relative memory usage. Additionally, it encompasses information pertaining to load-related memory management events, such as major page faults.

Space. (Or Absolute Resource Consumption, Space Complexity) The *amount* of data resources, such as memory and storage, required by the machine learning process. Memory space consumption is quantified by resource usage on the host system. This includes main memory usage and allocation, such as virtual memory allocation, resident set size, working set size or stack size.

Duration. (Or Time Requirements) Time-related measurements encompass metrics such as training duration and inference latency. These metrics quantify the time required to complete machine learning procedures and the duration spent on processing units.

Other non-dimension specific measures incorporate dataset characteristics, including the number of samples and the dataset size. Sample attributes like the number of sentences, words, and linguistic text properties are also considered for specialised metrics.

C. Efficiency

Efficiency (cost-effectiveness) is defined as the achievement of a high level of performance or accuracy while optimising the utilisation of resources and minimising associated costs. The objective is to balance the model's effectiveness (performance) and the costs or resources required to achieve that effectiveness. The CO-Metric approach encompasses three concepts of cost-effectiveness:

Solution Efficiency. Efficiency is defined as the balance between solution achievement and cost. Solutions are focused on quality, and costs include the efforts done and resources consumed. Every aspect is provided by one or multiple efficiency dimensions. Solution efficiency with a quality focus describes the computational effort used to achieve prediction quality. This reflects the efficiency of the model, i.e., the algorithm and its implementation. Efficiency increases by doing less work in less time and achieving higher prediction quality. Other focuses include achieving low latency of ML inference.

Resource Efficiency. Efficiency is defined as the degree to which resources are used. Resource efficiency is the capability of the ML procedure to utilise all available resources. It is increased by adapting to the host setup using more existing resources. This is important for designing hardware for specific ML techniques and adapting ML algorithms to specific hardware [39].

Synthetic Efficiency. Efficiency can be employed as a metric for the assessment of specific performance attributes. This may include the analysis of text quality indicators, as exemplified by the work of [40], or comparing performance outcomes [41].

Efficiency rules are defined based on the efficiency objectives:

1 Solution Efficiency

1.1 The more quality is achieved in less time, work and effort, the higher the ML quality efficiency.

1.2 The less time it takes to achieve more quality, the higher the ML-Speed-Efficiency.

1.3 The less work required for more quality, the higher the ML-Work-Efficiency.

2 Resource efficiency

2.1 The more load is used for more quality, less duration and less work; the higher the ML resource efficiency.

3 Synthetic efficiency

3.1 The less computational work is necessary per data chunk, the higher the ML model efficiency.

In addition to the efficiency objectives, there are two opposed requirements for handling ML efficiency results: interpretability and applicability. The greater the amount of information a metric provides, the greater the need for interpretation. This approach provides metrics at two levels of complexity. (i) Efficiency is determined as a single scalar by the at-a-glance metric (compact metric score) while supporting weights for each dimension. (ii) The efficiency vector metric represents uninterpreted values per dimension.

D. Compact Efficiency Metrics

Machine learning (ML) efficiency exhibits variability contingent upon the specific application. Metrics have been proposed for particular purposes and categorised according to their level of complexity. The group of compact metrics employs a subset of dimensions that contribute to calculating an efficiency score, which is determined with respect to the dominant dimension. The compact efficiency metric (CO) is specified in the Definition 1.

Definition 1: It exists a compact efficiency score CO of a ML procedure M for focused dimensions F with a focus weight α and unfocused dimensions U , defined in Equation (1); based on efficiency dimensions (D) quality q , work w , space s , load l and duration d with specific dimension weights β , detailed by Equation (2).

Quality Focused COmpact Efficiency Metric (QCO). A compact metric to reflect quality-focused efficiency. A score describes the best solution with a predefined high relevance of the quality dimension and low relevance of the work and duration dimensions. Relevant dimensions: Quality, Work, Space, Duration. Dominant dimension: Quality.

The QCO -score for an ML process M is derived from Equations (1) and (2) for the Quality-Focus, as stated by Equation (3). The quality dimension is represented by q , which measures the quality or performance of the machine learning model. w represents the dimension of computational effort, which quantifies the computational operations or effort required for the machine learning tasks. The resource consumption dimension s quantifies the system resources required during the execution of the model. d represents the dimension of duration, which measures the time or duration required to train the model. The weight per dimension β is employed to adjust the importance of dimensions, while α represents

$$[F]CO(M) = (F \times \alpha) \times U \quad (1)$$

$$[F]CO(M) = (q_M \times \beta_q) \times (w_M \times \beta_w) \times (s_M \times \beta_s) \times (l_M \times \beta_l) \times (d_M \times \beta_d) \times \psi \quad (2)$$

where $D = \{r \in \mathbb{R} \mid r > 1\}$ and $\{q, w, s, l, d \in D\}$

$F \subseteq D$ and $U = D \setminus F$

$\psi = \text{Score-Compensation}$

the additional weight of the focus dimension, both derived from expert knowledge of the use case. The compensation factor ψ is introduced to optimise the readability of the score, where $1 > \psi \geq 0.1$. The dominance of quality q is reflected in the numerator, so efficiency is defined as the quotient of quality divided by work w , space s and duration d (terms in the denominator). It is assumed that the dimensions will become increasingly important in proportion to their current size. Consequently, the weights β of the dimensions and the focus weight α are treated as exponents, with the respective dimension as the base.

$$QCO(M) = \frac{q^{\alpha \cdot \beta_q}}{(w^{\beta_w} + s^{\beta_s} + d^{\beta_d})} * \psi \quad (3)$$

Resource Focused Compact Efficiency Metric (RCO). Compact metric to reflect resource-oriented efficiency. A score describes the best solution with a predefined high relevance of the relative load usage and a low relevance of the quality dimension. Relevant dimensions: Load, Quality, Work, Duration. Dominant dimension: Load.

Inference Focused Compact Efficiency Metric (ICO). Compact metric to reflect resource-oriented efficiency. A score describes the best solution with a predefined high relevance of duration, low relevance of the quality dimension and lowest relevance of work. Relevant dimensions: Quality, Work, Duration. Dominant dimension: Duration.

Algorithmic Focused Compact Efficiency Metric (ACO). Compact metric to reflect resource-oriented efficiency. A score describes the best solution with predefined high relevance of work and duration, low relevance of duration, quality, and dataset dimension. Relevant dimensions: Quality, Work, Duration, Dataset. Dominant dimension: Work.

E. Efficiency Vector Metric (EV)

The CO metrics condense efficiency information into a score. To provide information on the dimension-specific performance, the EV metric reveals the dimension scores of the CO metric. The EV is available per CO as a vector, to describe the efficiency in the vector space of the specific CO. For QCO the QEV is represented by a vector in a *Quality-Work-Space-Duration* space.

TABLE IV
INSTANTIATION PROTOCOL.

Step	Objective
1	Select Efficiency Metric
2	Define Validity Requirements
3	Setup and Conduct Experiment
4	Define Dimensions
5	Analyse measurements
6	Assign measurements to Dimensions
7	Define Validity Ranges
8	Normalisation of Measurement-Values
9	Determine Dimensional Weights
10	Define Score compensation factor
11	Define Score Equation

F. Pareto

In the field of machine learning, it is important to select the most efficient configurations to streamline usage. This can be achieved by considering efficiency metrics and making preliminary selections. The selection process should consider the requirements of the machine learning workflow, which reflect diverse priorities such as speed, resource utilization, and outcome quality. Using a Pareto-based approach to select the most efficient configurations aligns well with the given criteria. This methodology identifies configurations that offer an optimal trade-off among competing objectives, ensuring that the selected setups are not only efficient but also tailored to the specific demands of the machine learning process.

IV. COMPACT METRIC INSTANTIATION

To operationalize the proposed efficiency metric, the abstract definitions must be instantiated. The instantiation is akin to the object instantiation of a class in programming. Attributes like the efficiency dimensions are set, and the metric can then be applied to measurements to receive scores. These attributes are defined based on empirical evidence, valid within specified ranges and tailored for particular use cases. This paper focuses on a text classification use case aimed at optimizing classification quality for two distinct datasets. The instantiation stages for a CO-Metric are listed in Table IV. For reasons of brevity, the instantiation is limited to the QCO metric.

A. Use Case

In this demonstration of the instantiation process Use Case 1 was selected, the aim is to identify the most efficient machine learning method for a set of text classification tasks, prioritising efficiency - a measure of quality per unit of work, space and time - over pure classification quality. The Quality COmpact (QCO) score is employed to assess this efficiency. The use case involves two different text classification tasks: spam classification, using the dataset of Almeida et al. [42], and sentiment analysis of movie reviews, based on the dataset of Maas et al. [43]. Vectorisation methods include traditional TF-IDF and advanced word embeddings using DistilBERT. Classifiers selected include Support Vector Machines, Naïve Bayes, Gradient Descent, Random Forest and a transformer-based method using a fine-tuned DistilBERT model for both vectorisation and classification tasks.

TABLE V
VALIDITY REQUIREMENTS

Aspect	Count	Variables	Optional
Dataset	≥ 2	Size, Sample Count	Sample Length, Language
Vectorization	≥ 2	Algorithm	Dictionary Size, Model Size
Classifier	≥ 4	Algorithm, Classifier Tech.	Hyperparameters
Host-Setup	≥ 2	Hardware Conf., Operating System	Software Version

B. The QCO-Metric Instance

The dimensions and the QCO score are instantiated according to the protocol given in Table IV. The required validity for different datasets and ML procedures results in the empirical variance requirements presented in Table V. To gain comparison validity among host-setups, four different computing environments were set up (No. 1-4 as shown in Table VI).

TABLE VI
HOST-SETUPS

No.	Type	CPU-Model	Clock	Threads	RAM
1	Virtualised	AMD Ryzen 7 5800U	1.9	8	16
2	BareMetal	Intel Core i5-6200U	2.3	4	8
3	BareMetal	Intel Core i7-7700	3.6	16	32
4	Virtualised	Intel Xeon Gold 6230	2.1	4	8
5	Virtualised	AMD EPYC 7742	2.2	16	16

[Clock in GHz, RAM in GB.]
OS: Linux, Language: Python3,
Libraries: Scikit-learn [44], DistilBERT [45], torch [46], pandas [47].

C. Observe Tool

To facilitate a reusable instrument for gauging efficiency, an observation tool was developed. This tool is designed to monitor the performance metrics of a Linux process. Initially,

it places the target process in a suspended state, thereby preparing the environment for the commencement of measurements. Subsequently, it activates three distinct measurement tools (Figure 2) and resumes the operation of the process under scrutiny. Upon the completion of the process, the observation tool automatically terminates the measurement utilities.

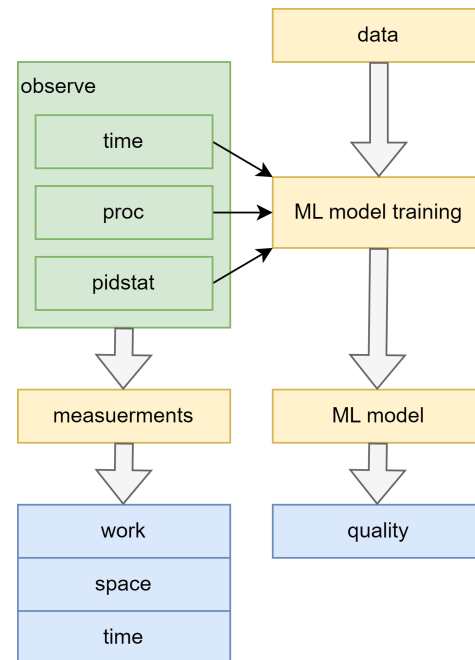


Fig. 2. Observe Tool

The measurements were provided by a set of Linux tools:

- `time`. Basic process measurement (CPU, Memory).
- `pidstat`. Advanced process measurement (CPU, Memory, IO-Usage).
- `perf`. Performance counter capture. (CPU, Memory).

The acquired dataset necessitates transformation for analytical purposes. The utility `time` generates output in a textual format, which requires conversion into a comma-separated values (CSV) format to facilitate subsequent data processing and analysis. Similarly, the `perf` utility collects performance metrics in a multi-table format, necessitating transformation into CSV format to enable efficient data manipulation and interpretation. Furthermore, the `pidstat` utility produces continuous output, which must be processed to extract minimal, maximal, and mean values over the specified duration. This transformation is essential for summarising performance characteristics and facilitating a comprehensive analysis of system behaviour under varying conditions.

Quality scores were computed separately from the ML procedure. Measurements were grouped for resource domain, e.g., memory consumption or computational work on CPU. The groups were filtered by correlation, the heatmap (Figure 3) shows Pearson Correlation Coefficients for selected measurements. `perf` was not supported on all host setups due to missing performance counters and conflicts with power

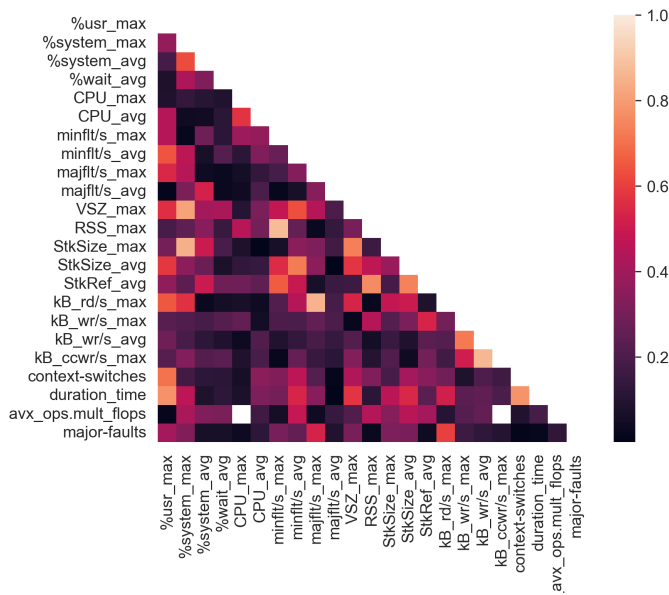


Fig. 3. Pearson Correlation Coefficients of Empirical Measurement Values.

TABLE VII
MEASUREMENTS

TYPE	DIM	IMP	TRANS	DEP	RNG
F1-Score	Quality	10	None		0-1
Bal.-Acc.	Quality	10	None		0-1
FLOPS	Work[CPU]	10	Log 63P	Data	0-63P
MinorPF	Work[CPU]	5	Log 800M	Data	0-800M
RSS (avg)	Space[Mem]	10	Log 128G	Time	0-128G
CPU Time [ns]	Duration	10	Log 172T		0-172T
Data Size	-	10	Log 256M		0-256M

TYPE=Measurement; DIM=Dimension; IMP=Weight; TRANS=Data transformation; DEP=Dependency; RNG=Range of validity

saving methods. Two sets of measurements have to be set up, which results in two QCO flavours: *F*loating Point Operation (QCO_F) based and *M*inor Page *F*ault (QCO_P) based. The selected measurements are listed in Table VII.

Range definition is essential for normalisation. The valid ranges for this QCO-Instance are listed in Table III. Normalisation is necessary due to the use of different units and data types in calculations. Monotonic data transformations on dimension values result in a range between 0 and 2, based on the maximum values per dimension. Consequently, valid ranges are not related to measurement ranges. The definition of valid measurement ranges, as shown in Table III, facilitates data transformations on measurement values. After transformation, values are placed on a closed scale with a slightly decreased distribution (Figure 4).

The dimension-equations are defined by interpreting the dependencies of the measurements (Table VII). Especially the dependency on duration and data size has been considered.

The dimension weight is used to adjust the importance of the focused metrics. The importance of quality is based on domain knowledge: Quality is about two times more important than

work, space, and duration which delivers $\beta_Q = 6$. Readability compensation ψ is set to 10.

QCO is defined for each set of measures, resulting in Equations (4) and (5).

TABLE VIII
QCO DIMENSION INSTANCES

Dimension	QCO_F	QCO_P (*)
Quality	(F1 + BACC) / 2	(F1 + BACC) / 2
Work	FLOPS / Dataset[kB]	Minor PF / Dataset[kB]
Space	aRSS[s[MB]] * Duration[s]	aRSS[MB] * Duration[s]
Duration	Time on CPU [ns]	Time on CPU [ns]

(*) FLOPS-Measurement was not available on all hosts.

D. QCO Metric Usage

- 1) Select QCO type according to available measurements. If CPU-Performance-Counters are available QCOF, otherwise QCOP. Respect expected validity ranges (Table III).
- 2) Perform training on a dataset subset while capturing measurements according to Table VIII.
- 3) Calculate efficiency by Equations (4) and (5).

E. QCO Score Calculation

The Quality-Focused Score is calculated for FLOPS-based-score as QCO_F (4) and QCO_P for Page-Fault-based score (5).

F. Pareto Implementation

The implementation of the Pareto principle has been used to address three critical dimensions: quality, space, and workload efficiency. To improve the utility and manageability of the Pareto set, a compression technique has been employed. This approach involves removing points within the set that exhibit a high degree of similarity, with a predefined similarity threshold set at 50% of the value range. This methodological adjustment ensures a streamlined and representative Pareto set, making it easier to identify optimal solutions across the specified dimensions.

V. EVALUATION

The applicability, plausibility and balance of the proposed metric is assessed in a comprehensive evaluation.

A. Experiments

In Experiment 1, preliminary studies were conducted to design the text classification pipeline. This included the selection and configuration of appropriate measurement tools, as well as the implementation of hyperparameter adjustments to set up text classification algorithms.

In Experiment 2, Use Case 1 was adapted to binary classification tasks, which were performed by different vectorization and classifier technologies. Two datasets are selected for Experiment 2, both with moderate text length; SMS Spam Classification (25.000 samples) [42] and Movie Survey Classification (7.805 samples) [43]. The experiments were

$$QCO_F(M) = \frac{\left(\left(\frac{F1+BACC}{2}\right)^6\right)}{\left(\log_{63P} FLOPS/DS[kB] + \log_{128G} RSS[MB] * \log_{864M} D[s] + \log_{172T} TOC[ns]\right)} * 10 \quad (4)$$

$$QCO_F(M) = \frac{\left(\left(\frac{F1+BACC}{2}\right)^6\right)}{\left(\log_{800M} MPF/DS[kB] + \log_{128G} RSS[MB] * \log_{864M} D[s] + \log_{172T} TOC[ns]\right)} * 10 \quad (5)$$

where $F1$ = F1-Score, $BACC$ = Balanced Accuracy Score,
 $FLOPS$ = Floating Point Ops., MPF = Minor Page Faults,
 DS = Dataset-Size, RSS = Resident Set Size,
 D = Duration, TOC = Time on CPU

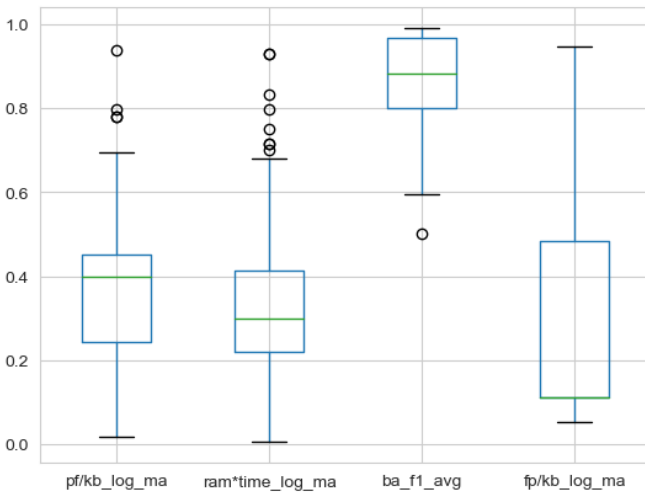


Fig. 4. Spread and Skewness per Dimension after logarithmic smoothing.

run on host 1 (Table VI) in two virtual hosts with different virtualisation technologies. The results of experiment 2 are shown in Table IX. To compare the QCOF and QCOP metrics in Experiment 2, two sets of QCO had to be created as some FLOPS measurements were not available (QCO_1 & QCO_2^*).

In Experiment 3, the metric was further evaluated by applying it to an optimisation problem similar to Use Case 4. The objective was hyper-parameter optimisation with efficiency as the cost function. The ML process involved fine-tuning a transformer model (DistilBERT [45]), word embedding and text classification. The experiment aimed to find the most efficient value for the Maximum Sequence Length (MSL) for the SMS spam detection task [42], which was run on host 5 (Table VI).

B. Applicability

Experiment 2 shows surprising results that can be explained by runtime conditions such as schedulers, competing pro-

cesses and caching techniques. The experiment is not designed to make general statements about specific combinations of vectorization or classification methods. Consequently, the following statements apply only to this experiment, which does not preclude testing the usefulness of the efficiency metric. The word embedding method is, on average, superior to the TFIDF in terms of quality, but there are classifiers (NB, GD) that can compensate for the quality disadvantage and, in some cases, achieve the highest efficiency. This is due to the low workload. The transformer method requires significantly more work. It achieves high quality, but also takes the longest time. The Random Forest (RF) classifier has a low efficiency because it requires a lot of computation and time to achieve good quality. The Support Vector Machine (only linear kernel) classifier benefits most from the word embeddings and, therefore, achieves good efficiency. When comparing the combinations in terms of the time-to-work ratio (WO-Focus), the worst ratio (1.28) is found for IMDB/TFIDF/SVM and the best for SMS/TFIDF/NB with 0.39. This leads to the conclusion that the measurement of time does not reflect the amount of work.

In Experiment 3, both QCO_F and QCO_P were successfully computed (see Table X). The most efficient MSL configuration consisted of 512 tokens, resulting in a high classification quality and moderate duration. On the other hand, the configuration with 126 tokens showed an increased workload and duration. The fastest result was obtained with an MSL of 256 tokens.

C. Plausibility

QCO was successfully generated for all ML methods in Experiment 2. A comparative assessment of QCO based on expert rankings is used for evaluation. Domain experts ranked the dimensions, listed in Table IX Column Rank-EXP. Comparing expert and QCO rankings, a minimal deviation from the expert rank was observed for high-quality ML methods, but the deviation increased with decreasing quality. This variance can be attributed to the expert's specific weighting of quality

TABLE IX
QCO EVALUATION RESULTS

DAT	VECT	CLF	DUR	EVMetric					QCO Metrics		Rankings				
				QUA	TIME	SPA	WO _P	WO _F	QCO _P	QCO _F	EXP ₁	QCO _{1P}	EXP ₂	QCO _{2P}	QCO _{2F}
SMS	TFIDF	NB	00:00:34	0.968	0.407	0.260	1.043	0.691	1.260	1.726	1	1	1	1	1
SMS	TFIDF	GD	00:02:36	0.972	0.583	0.371	1.043	0.631	1.190	1.678	2	2	2	2	2
IMDB	TFIDF	GD	00:00:19	0.885	0.339	0.222	0.616	0.610	1.145	1.153	3	3	3	3	3
SMS	DIST-T	DIST-T	00:01:34	0.982	0.525	0.384	1.249		1.099		6	4			
SMS	BERT	SVM	00:11:29	0.972	0.755	0.510	1.143	1.465	1.018	0.852	4	5	5	4	4
IMDB	DIST-T	DIST-T	02:38:32	0.982	1.058	0.795	1.058		0.970		5	6			
SMS	BERT	GD	02:04:00	0.978	1.030	0.696	1.140	1.637	0.956	0.752	8	7	4	5	6
IMDB	DISTIL	DISTIL	11:31:52	0.978	1.228	0.923	1.136		0.848		7	8			
IMDB	TFIDF	NB	00:01:18	0.845	0.504	0.330	0.618	0.581	0.766	0.797	9	9	6	6	5
SMS	BERT	NB	01:58:30	0.932	1.024	0.692	1.141	1.638	0.715	0.563	11	10	8	7	8
SMS	DISTIL	DISTIL	01:39:58	0.983	1.005	0.750	1.845		0.697		12	11			
SMS	BERT	RF	01:09:50	0.916	0.963	0.651	1.140	1.639	0.660	0.516	10	12	7	8	9
IMDB	TFIDF	RF	00:00:58	0.809	0.469	0.309	0.617	0.646	0.604	0.586	15	13	9	9	7
SMS	TFIDF	RF	00:03:02	0.787	0.601	0.376	1.043	0.931	0.335	0.363	16	14	12	10	10
IMDB	TFIDF	SVM	00:20:20	0.714	0.821	0.554	0.638	0.812	0.223	0.194	13	15	11	11	11
SMS	TFIDF	SVM	00:00:35	0.652	0.409	0.264	1.048	1.092	0.117	0.113	14	16	10	12	12

Columns: Dataset, Vectorizer, Classifier, Duration, Quality, Time, Space, WO_F = Work (FLOPS), WO_P = Work (Minor Page Faults), QCO Metrics, Rankings by Domain EX Perts, or QCO,

Abbrev.: SMS = SMS Spam Dataset [42], IMDB = IMDB Dataset [43], BERT = BERT word embedding, DIST-T = finetuned DistilBERT word embedding (PyTorch) & classification, DISTIL = finetuned DistilBERT word embedding (TensorFlow + keras) & classification, GD = Gradient Descent, SVM = Support Vector Machine, NB = Naïve Bayes, RF = Random Forest

TABLE X
EFFICIENCY OF DISTILBERT

SL	Measurements			Dimensions			Scores	
	Duration	F1	Q	W	S	T	QCO _F	QCO _P
128	09:08:50	0.76	0.64	3.02	0.45	0.78	0.159	0.164
256	00:27:02	0.78	0.64	2.86	0.45	0.71	0.171	0.172
512	00:50:13	0.78	0.65	2.94	0.47	0.72	0.183	0.174

Text Classification Efficiency with DistilBERT with different maximum Sequence Length (SL). Smoothed Dimensions: Quality, Work, Space and Time. Efficiency Scores Quality-Focused based on FLOPS (QCO_F) and Minor Page Faults (QCO_P)

relevance, which is particularly evident in the DistilBERT setups.

D. Balance & Compensation

QCO achieved a balance of aspects through compensation (Table III). The results of Experiment 2 showed no anomalies for different datasets; even ML processes with large datasets achieved high efficiency. Moreover, significant differences in speed and computational complexity were observed for comparable efficiency, suggesting a balance in these aspects. Due to the small number of hosts available for evaluation, the balance on host setups could not be verified.

E. Pareto Application

Pareto sets were successfully established for both datasets, as delineated in Table XI, optimizing for maximization of quality while minimizing work and space requirements. The initial composition of these sets included two distinct points for the IMDB and seven for the SMS Spam datasets. Subsequent refinement involved the elimination of points exhibiting significant similarity, with the threshold set at 50% of

the value range, leading to a reduction in the SMS Spam Dataset's Pareto set (Figure 6), while the IMDB Dataset's set remained unchanged (Figure 5). The optimal Pareto front was characterized by the most favourable values within the set, namely the highest quality alongside the minimum for both work and space. The distance to this Pareto front was computed to evaluate the proximity of solutions to the optimal trade-off.

Analysis of the results for the IMDB Dataset [43] revealed that the fastest solution was achieved through TFIDF Vectorization combined with Gradient Descent, whereas the best quality solution was attributed to a fine-tuned DistilBERT model. For the SMS Spam Dataset [42], three top-quality performers emerged: TFIDF combined with Naïve Bayes, BERT with Gradient Descent, and a fine-tuned DistilBERT. A balanced solution, offering a compromise between quality and computational efficiency, was identified as TFIDF combined with Random-Forest. These findings highlight the efficacy of employing a Pareto-based approach to identify optimal solutions that cater to varying optimization dimensions across different datasets.

VI. DISCUSSION

This study proposes an efficiency metrics framework for machine learning techniques that addresses different aspects of cost-effectiveness, resource utilisation and model performance. The approach is intended to be adaptable and applicable to a variety of ML techniques and host setups.

The objectives of the efficiency metrics framework have been defined with the intention of addressing different real-world scenarios and use cases. The proposed efficiency metrics provide information that can be used to identify the optimal ML technique and hyperparameters, select ML techniques for

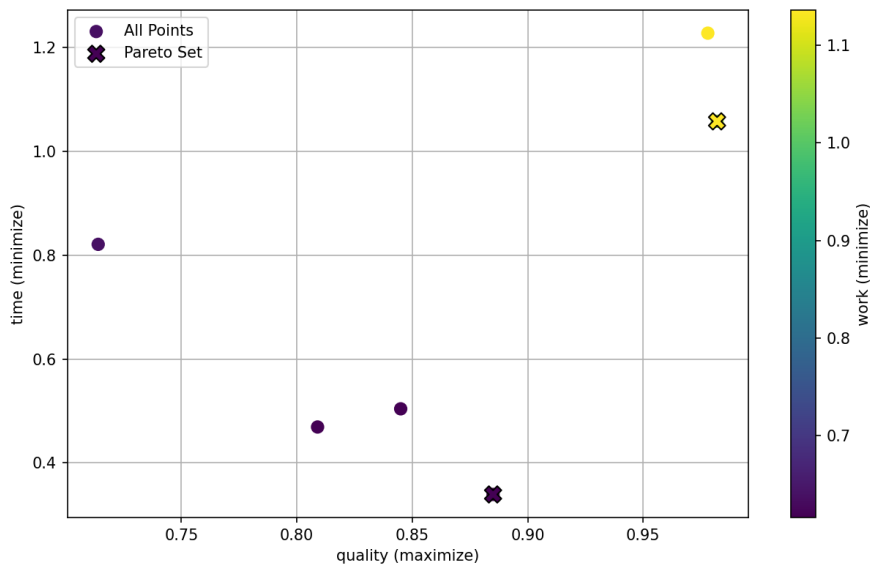


Fig. 5. Pareto results IMDB Dataset [43]

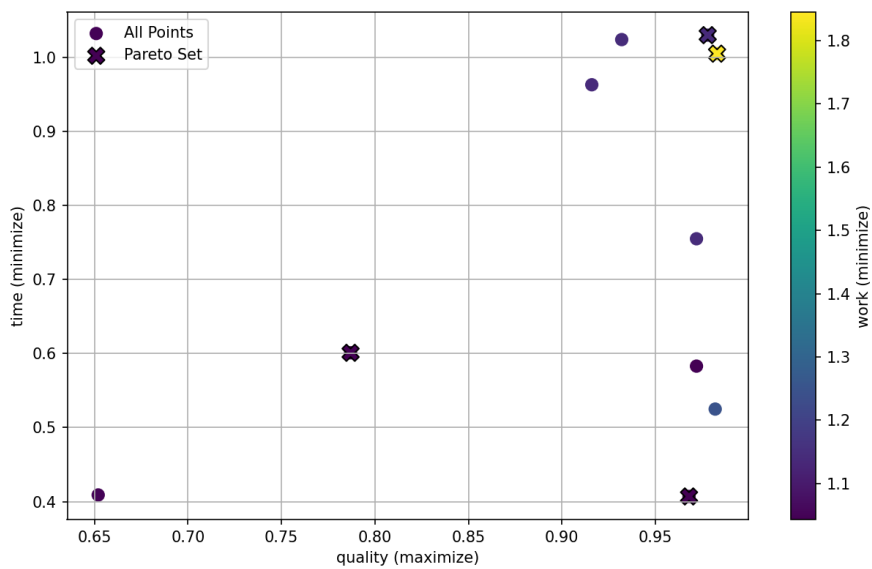


Fig. 6. Pareto results for SMS Spam Dataset [42]

TABLE XI
PARETO SETS

DAT	VECT	CLF	DUR	QUA	Time	SPA	Work	QCOP	Distance
SMS	TFIDF	NB	00:00:34	0.97	0.41	0.26	1.04	1.26	0.02
IMDB	TFIDF	GD	00:00:19	0.89	0.34	0.22	0.62	1.15	0.10
IMDB	DISTIL	DISTIL	02:38:32	0.98	1.06	0.80	1.06	0.97	0.84
SMS	BERT	GD	02:04:00	0.98	1.03	0.70	1.14	0.96	0.63
SMS	DISTIL	DISTIL	01:39:58	0.98	1.01	0.75	1.85	0.70	1.00
SMS	TFIDF	RF	00:03:02	0.79	0.60	0.38	1.04	0.34	0.28

Columns: Dataset, Vectorizer, Classifier, Quality, Time, Space, Work, *QCOP* Metric, Distance to Pareto Front.

Abbrev: SMS = SMS Spam Dataset [42], IDB = IMDB Dataset [43], BERT = BERT word embedding, DIST = finetuned DistilBERT word embedding (PyTorch) & classification, DISTIL = finetuned DistilBERT word embedding (TensorFlow + keras) & classification, GD = Gradient Descent, SVM = Support Vector Machine, NB = Naive Bayes, RF = Random Forest

limited resources or private data, compare classification performance across different host setups, and estimate computational cost.

The metrics framework introduces several dimensions that collectively capture the efficiency of machine learning techniques in achieving the aforementioned goals. These dimensions include quality, work, load, space and duration, each of which contributes to the overall efficiency score. The dimensions are intended to assess different aspects of machine learning performance and resource use, allowing for a comprehensive evaluation.

A key advantage of the proposed framework is its adaptability to different ML techniques and tasks. The dimensions and metrics can be adjusted based on specific use cases and requirements, ensuring relevance and accuracy in different contexts. This adaptability makes the metric framework suitable for a wide range of applications, from small-scale experiments to large-scale production systems.

The efficiency metrics introduced in the framework, such as QCO, FCO, ICO and ACO, provide different perspectives on efficiency. These compact metrics provide a clear, at-a-glance view of efficiency, making it easier for researchers and practitioners to evaluate and compare different ML techniques. In addition, the Efficiency Vector (*EV*) metric provides detailed information about the performance of ML techniques on individual dimensions, providing insights for further analysis and improvement.

The process of instantiating the efficiency metrics necessitates empirical investigation to ensure that the metric definitions are concrete and applicable to specific ML experiments. The validity of metric instantiation is emphasised, and the size of the experiment plays an important role in achieving reliable results. By conducting experiments on different datasets and host setups, the metric instantiation gains credibility and comparability.

The development of the observational tool provides a pragmatic approach to incorporating efficiency measurements into the machine learning (ML) toolchain. The implementation of an out-of-the-box efficiency measurement tool represents a significant contribution in several ways. First, it provides a tangible benefit to practitioners and researchers in the field of machine learning by facilitating the direct assessment of

computational efficiency. This tool enables users to benchmark and optimise the performance of ML algorithms and systems, thereby improving the overall efficiency of the ML development lifecycle. Secondly, by providing a standardised method for measuring efficiency, it contributes to the reproducibility and comparability of experimental results, which are fundamental to the scientific rigour of machine learning research. Finally, the tool's accessibility and ease of integration into existing ML toolchains encourages its adoption, thereby broadening its impact on the field through improved performance evaluation practices.

In summary, the efficiency metrics framework developed in this study presents a robust approach to quantifying and comparing the efficacy of machine learning methods in terms of both performance and resource consumption. Its flexible and comprehensive nature makes it an invaluable resource for the machine learning community, aiding in the strategic decision-making process regarding the deployment of machine learning models and the allocation of computational resources. The continued refinement and application of this framework are poised to significantly influence the efficiency and sustainability of future machine learning endeavours.

VII. CONCLUSION AND FUTURE WORK

The successful computation and evaluation of efficiency scores represent a step forward in improving the effectiveness of machine learning research. By incorporating sophisticated dimensions that reflect measurement interdependencies - such as FLOPS relative to data volume or memory usage relative to duration - we could effectively balance the metric to account for variations in dataset size and host setup. A lack of sufficient samples challenged the evaluation of the QCO dimensions, which limited the robustness of our findings. Nevertheless, the FLOPS-based evaluation demonstrated consistency, and our innovative use of a minor page fault measure to extend the work dimension met with limited success. For future efforts, exploring efficiency dimensions that integrate ML-specific factors, such as model size, would be prudent to deepen our understanding of efficiency dynamics in different machine learning contexts. It is pertinent to highlight that the validation methods used for the proposed efficiency metric rely heavily on expert consensus. While this approach provides insightful

feedback, the need for improved statistical validation is evident to strengthen the metric's credibility and structural integrity. Expert insight and the relevance of quality within specific applications particularly shape the effectiveness of machine learning methods. However, if quality is considered the sole criterion for development, it is important to recognise the potential for a significant escalation in complexity. Achieving a balance between different dimensions of efficiency is essential to ensure a pragmatic and reasoned strategy for optimising machine learning workflows.

A. Future

For further research, several key areas have been identified that require in-depth exploration to improve the understanding and implementation of machine learning efficiency metrics.

Comparability and Generalisability: This research has advanced the field by addressing comparability through the identification of dependencies within the data. The proposed compensations have validated the conceptual framework. However, a more detailed identification of these dependencies and a refinement of the compensation procedures remain crucial. Future studies should consider additional variables, such as those listed in the 'optional' column of Table V, or the complexity of the dataset to enhance the granularity and applicability of the results.

Applicability: The measurement tool presented in this study should be developed into a comprehensive library that simplifies the process of obtaining efficiency scores. This development would make the tool more accessible to a wider user base. In addition, creating solutions tailored for cloud-based computing environments will extend the utility of the tool and facilitate its widespread adoption in different computing environments.

Validation: Expanding the scope of experiments is essential for a deeper understanding of the biases present in current efficiency metrics. Comprehensive statistical investigations should be conducted to robustly validate these metrics, reducing reliance on expert opinion and strengthening the empirical foundations of efficiency measures.

Recurrent optimisation: Continuous improvement through recurrent optimisation processes is essential. Efficiency considerations should be routinely integrated into the development cycle of machine learning algorithms. This ongoing refinement will ensure that algorithms are not only effective but also optimised for efficiency and adapted to evolving computing environments and requirements.

AutoML: Integration of efficiency analysis within the AutoML framework to enhance automatic model selection and optimisation processes by incorporating efficiency metrics.

By addressing these areas, future research can significantly contribute to the sophistication and practicality of efficiency metrics in machine learning, paving the way for green machine learning.

ACKNOWLEDGMENTS

This Research was funded by the Federal Ministry of Education and Research of Germany in the framework of FiSK (Project-Number 16DHB4005).

REFERENCES

- [1] D. Schönle, C. Reich, and D. O. Abdeslam, "Comparable Machine Learning Efficiency: Balanced Metrics for Natural Language Processing," in *GREEN 2023*, 2023.
- [2] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: State of the art, current trends and challenges," *Multimedia tools and applications*, vol. 82, no. 3, pp. 3713–3744, 2023.
- [3] D. Rousseau and A. Ustyuzhanin, "Machine Learning scientific competitions and datasets," in *Artificial Intelligence for High Energy Physics*, World Scientific, 2022, pp. 765–812. DOI: 10.1142/9789811234033_0020.
- [4] A. Ittoo and A. van den Bosch, "Text analytics in industry: Challenges, desiderata and trends," *Computers in Industry*, vol. 78, pp. 96–107, 2016. DOI: 10.1016/j.compind.2015.12.001.
- [5] R. W. Numrich, "Performance metrics based on computational action," *The International Journal of High Performance Computing Applications*, vol. 18, no. 4, pp. 449–458, 2004.
- [6] A. Vaswani *et al.*, "Tensor2Tensor for Neural Machine Translation," in *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, 2018, pp. 193–199.
- [7] D. Ghimire, D. Kil, and S.-h. Kim, "A survey on efficient convolutional neural networks and hardware acceleration," *Electronics*, vol. 11, no. 6, p. 945, 2022. DOI: 10.3390/electronics11060945.
- [8] G. Tzanos, C. Kachris, and D. Soudris, "Hardware acceleration on gaussian naive bayes machine learning algorithm," in *2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, 2019, pp. 1–5. DOI: 10.1109/mocast.2019.8741875.
- [9] W. Fu, K. Wang, C. Zhang, and J. Tan, "A hybrid approach for measuring the vibrational trend of hydroelectric unit with enhanced multi-scale chaotic series analysis and optimized least squares support vector machine," *Transactions of the Institute of Measurement and Control*, vol. 41, no. 15, pp. 4436–4449, 2019.
- [10] Z. Li, M. Paolieri, L. Golubchik, S.-H. Lin, and W. Yan, "Predicting throughput of distributed stochastic gradient descent," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 11, pp. 2900–2912, 2022. DOI: 10.1109/tpds.2022.3151739.
- [11] T.-J. Yang, Y.-H. Chen, J. Emer, and V. Sze, "A method to estimate the energy consumption of deep neural networks," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*, 2017, pp. 1916–1920. DOI: 10.1109/aacssc.2017.8335698.
- [12] J. Thomson, M. O'Boyle, G. Fursin, and B. Franke, "Reducing training time in a one-shot machine learning-based compiler," in *International workshop on languages and compilers for parallel computing*, 2009, pp. 399–407. DOI: 10.1007/978-3-642-13374-9_28.
- [13] R. Fischer, M. Jakobs, S. Mücke, and K. Morik, "A Unified Framework for Assessing Energy Efficiency of Machine Learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2022, pp. 39–54. DOI: 10.1007/978-3-031-23618-1_3.
- [14] A. Kumar, S. Goyal, and M. Varma, "Resource-efficient machine learning in 2 kb ram for the internet of things," in *International conference on machine learning*, 2017, pp. 1935–1944.
- [15] J. Huang *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7310–7311. DOI: 10.1109/cvpr.2017.351.
- [16] H. Westphal and S. Menge, "On the Supervisory Control of Distributed High-Performance Computing Systems in Engineering," *WIT Transactions on Information and Communication Technologies*, vol. 11, 1970.
- [17] D. F. Snelling, "A philosophical perspective on performance measurement," in *Computer benchmarks*, 1993, pp. 97–103.
- [18] R. W. Numrich, "Computational force, mass, and energy," *International Journal of Modern Physics C*, vol. 8, no. 03, pp. 437–457, 1997. DOI: 10.1142/s0129183197000370.
- [19] K. Deb, K. Sindhya, and J. Hakanen, "Multi-objective optimization," in *Decision sciences*, CRC Press, 2016, pp. 161–200. DOI: 10.1201/9781315183176-4.

- [20] Y. Cui, Z. Geng, Q. Zhu, and Y. Han, "Multi-objective optimization methods and application in energy saving," *Energy*, vol. 125, pp. 681–704, 2017.
- [21] P. Ngatchou, A. Zarei, and A. El-Sharkawi, "Pareto multi objective optimization," in *Proceedings of the 13th international conference on, intelligent systems application to power systems*, 2005, pp. 84–91.
- [22] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002. DOI: 10.1109/4235.996017.
- [23] M. Kim, T. Hiroyasu, M. Miki, and S. Watanabe, "SPEA2+: Improving the performance of the strength Pareto evolutionary algorithm 2," in *Parallel Problem Solving from Nature-PPSN VIII: 8th International Conference, Birmingham, UK, September 18-22, 2004. Proceedings 8*, 2004, pp. 742–751. DOI: 10.1007/978-3-540-30217-9_75.
- [24] C. A. C. Coello, *Evolutionary algorithms for solving multi-objective problems*. Springer, 2007.
- [25] H. Han, W. Lu, L. Zhang, and J. Qiao, "Adaptive gradient multiobjective particle swarm optimization," *IEEE transactions on cybernetics*, vol. 48, no. 11, pp. 3067–3079, 2017. DOI: 10.1109/tcyb.2017.2756874.
- [26] R.-J. Kuo and F. E. Zulvia, "Multi-objective cluster analysis using a gradient evolution algorithm," *Soft Computing*, vol. 24, no. 15, pp. 11 545–11 559, 2020. DOI: 10.1007/s00500-019-04620-0.
- [27] B. C. Arnold, "Pareto distribution," *Wiley StatsRef: Statistics Reference Online*, pp. 1–10, 2014. DOI: 10.1002/9781118445112.stat01100.
- [28] R. Sanders, "The Pareto principle: its use and abuse," *Journal of Services Marketing*, vol. 1, no. 2, pp. 37–40, 1987. DOI: 10.1108/eb024706.
- [29] E. Lomurno, S. Samele, M. Matteucci, and D. Ardagna, "Pareto-optimal progressive neural architecture search," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2021, pp. 1726–1734. DOI: 10.1145/3449726.3463146.
- [30] T. Elsken, J. H. Metzen, and F. Hutter, "Neural Architecture Search: A Survey," *Journal of Machine Learning Research*, 2019.
- [31] M. Davtalab-Olyaie and M. Asgharian, "On Pareto-optimality in the cross-efficiency evaluation," *European Journal of Operational Research*, vol. 288, no. 1, pp. 247–257, 2021. DOI: 10.1016/j.ejor.2020.05.040.
- [32] S. Vargas and P. Castells, "Rank and relevance in novelty and diversity metrics for recommender systems," in *Proceedings of the fifth ACM conference on Recommender systems*, 2011, pp. 109–116.
- [33] D. Schönle, C. Reich, and D. O. Abdeslam, "Linguistic driven feature selection for text classification as stop word replacement," *Journal of Advances in Information Technology*, vol. 14, no. 4, pp. 796–802, 2023. DOI: 10.12720/jait.14.4.796-802.
- [34] S. P. Bayerl *et al.*, "Offline model guard: Secure and private ML on mobile devices," in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2020, pp. 460–465. DOI: 10.23919/date48585.2020.9116560.
- [35] M. Feurer and F. Hutter, "Hyperparameter optimization," *Automated machine learning: Methods, systems, challenges*, pp. 3–33, 2019. DOI: 10.1007/978-3-030-05318-5_1.
- [36] S. González-Carvajal and E. C. Garrido-Merchán, "Comparing BERT against traditional machine learning text classification," *arXiv preprint arXiv:2005.13012*, 2020.
- [37] C. Zhang, M. Yu, W. Wang, and F. Yan, "5MArk6: Exploiting Cloud Services for 5Cost-Effective6,5SLO-Aware6 Machine Learning Inference Serving," in *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, 2019, pp. 1049–1062.
- [38] R. W. Numrich, L. Hochstein, and V. R. Basili, "A metric space for productivity measurement in software development," in *Proceedings of the second international workshop on Software engineering for high performance computing system applications*, 2005, pp. 13–16. DOI: 10.1145/1145319.1145324.
- [39] M. Capra, B. Bussolino, A. Marchisio, G. Masera, M. Martina, and M. Shafique, "Hardware and software optimizations for accelerating deep neural networks: Survey of current trends, challenges, and the road ahead," *IEEE Access*, vol. 8, pp. 225 134–225 180, 2020. DOI: 10.1109/access.2020.3039858.
- [40] C. Kiefer, "Quality Indicators for Text Data," 2019. DOI: 10.18420/btw2019-ws-15.
- [41] E. M. Dharma, F. L. Gaol, H. Warnars, and B. Soewito, "The accuracy comparison among word2vec, glove, and fasttext towards convolution neural network (cnn) text classification," *J Theor Appl Inf Technol*, vol. 100, no. 2, p. 31, 2022.
- [42] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami, "Contributions to the study of SMS spam filtering: new collection and results," in *Proceedings of the 11th ACM symposium on Document engineering*, 2011, pp. 259–262. DOI: 10.1145/2034691.2034742.
- [43] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning Word Vectors for Sentiment Analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA: Association for Computational Linguistics, 2011, pp. 142–150.
- [44] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [45] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.
- [46] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A Matlab-like Environment for Machine Learning," in *BigLearn, NIPS Workshop*, 2011.
- [47] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, Stéfan van der Walt and Jarrod Millman, Eds., 2010, pp. 56–61.