Comparable Machine Learning Efficiency: Balanced Metrics for Natural Language Processing

Daniel Schönle
IDACUS Insitute
Furtwangen University
Furtwangen, Germany
daniel.schoenle@hs-furtwangen.de

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 becomes increasingly pervasive, its resource demands and financial implications escalate, necessitating energy and cost optimisations to meet stakeholder demands. Quality metrics for predictive machine learning models are abundant, but efficiency metrics remain rare. We propose a framework for efficiency metrics, that enables the comparison of distinct efficiency types. A quality-focused efficiency metric is introduced that considers resource consumption, computational effort, and runtime in addition to prediction quality. The metric has been successfully tested for usability, plausibility, and compensation for dataset size and host performance. This framework enables informed decisions to be made about the use and design of machine learning in an environmentally responsible and cost-effective manner.**

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

I. Introduction

Decades ago, the primary motivation behind the pursuit of computational efficiency was the limited computing power available at the time. Computing resources had to be used judiciously to overcome the constraints imposed by hardware limitations. The advent of powerful computing resources, particularly in the field of Machine Learning (ML), has shifted the focus to achieving superior prediction quality, relegating efficiency to a secondary concern. As ML continues to evolve, the future landscape of human-computer interaction will be profoundly influenced by the widespread adoption of Large Language Models (LLMs) [1]. This shift is also driven by the integration of ML into heterogeneous computing environments, such as edge computing on resource-limited hardware. From a green computing and sustainability perspective, the use of resource-intensive solutions such as transformer-based word embeddings or LLMs may not always be financially or environmentally viable [2]. As a result, there is a growing demand for efficiency, especially for the widespread application of ML. The objective of this publication is to contribute to the improvement of efficiency in ML by introducing robust metrics for measuring the efficiency of machine learning models.

ML research has focused 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 result comparison, best practices in ML research typically include detailed descriptions of experiments, encompassing datasets, preprocessing steps, machine learning techniques, hyper-parameters, and hardware setups [3]. The absence of a dataset-agnostic procedures and missing 'Golden Standard' datasets pose challenges in achieving true repeatability and fair comparisons in Natural Language Processing (NLP) [4]. Furthermore, evaluating the impact of novelties in ML process steps, such as improved preprocessing, on prediction quality is a complicated task, as the other ML steps may be influential.

The delicate balance between complexity and outcome is often overlooked in research efforts to utilise all available resources to reduce time-to-solution. Evaluating time and space complexity becomes 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 in an optimal way to accomplish a mission within imposed constraints.” It becomes essential to establish metrics that address the efficiency concerns alongside prediction quality in the context of ML research.

We present a proposal to fill the existing gap by introducing novel metrics for measuring the efficiency of machine learning models. By incorporating resource consumption, computational effort, and runtime considerations into our efficiency metrics, we aim to provide a holistic perspective on the true efficiency of ML models. We demonstrate the process of defining a quality focused efficiency metric (Figure 1) and present the Quality COmpact (QCO) Efficiency Metric (Equations 4 & 5). We recognise the importance of dataset-agnostic evaluation and propose solutions to address this challenge and demonstrate the advantages of our metric for evaluating hyperparameter tuning. Our goal is to empower researchers and practitioners to make informed decisions that prioritise both prediction quality and efficiency, thus advancing the field of ML towards sustainable, green, and economically feasible solutions.

The structure of the paper is as follows: Section 2 (State of the Art) covers the research on efficiency types and metrics in ML. In Section 3 the efficiency metric is presented by elaborating on its objectives, followed by the theoretical foundations of *efficiency dimensions* and concepts, and finally the

ISBN: 978-1-68558-097-1

definition of the efficiency metrics. In the subsequent Section 4, 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 5 uses two experiments to assess the performance of the metric. The results obtained are discussed in detail in Section 6, leading to the presentation of the conclusion (Section 7).

II. State of the Art

This section begins with approaches that deal with computational cost as a method. The goal of predicting computational cost incorporate with the prediction of financial cost. Then, approaches to resource effectiveness are considered. Their goal is to find algorithms that work in most cost-effective way. Finally, general approaches to efficiency metrics are examined.

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 presented here defines work and duration dimensions based on actual measurements.

Computational Cost for Deep Learning is specific to deep learning, as it relies on complex neural network architectures, which makes direct computation of complexity difficult. Several approaches attempt to predict complexity, such as the proposal 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 method presented here, but with limited validity for deep learning optimised with distributed SGD.

Resource Efficiency is important for deep learning, where hardware requirements differ from those of statistical machine learning and are constantly evolving. 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 presented here addresses this aspect to provide comprehensive statements about the entire ML task.

Efficiency Comparison plays a role in the evaluation of 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 compares 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] discusses 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].

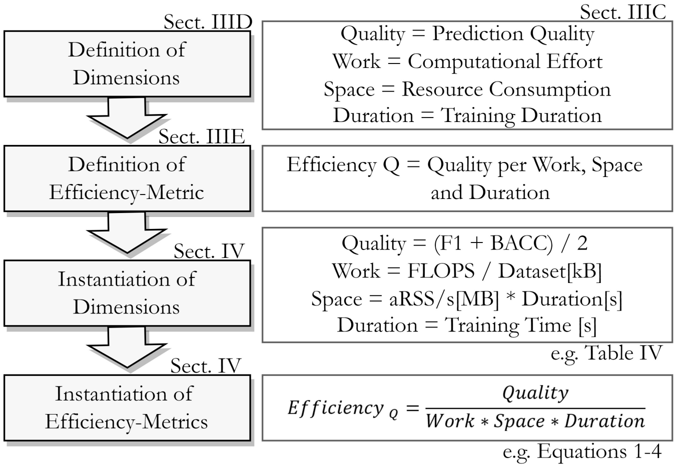
Sect. IIID
Definition of Dimensions
Sect. IIIC
Quality = Prediction Quality
Work = Computational Effort
Space = Resource Consumption
Duration = Training Duration
Sect. IIIE
Definition of Efficiency-Metric
Efficiency Q = Quality per Work, Space and Duration
Sect. IV
Instantiation of Dimensions
Quality = (F1 + BACC) / 2
Work = FLOPS / Dataset[kB]
Space = aRSS/s[MB] * Duration[s]
Duration = Training Time [s]
e.g. Table IV
Sect. IV
Instantiation of Efficiency-Metrics
$Efficiency_Q = \frac{Quality}{Work * Space * Duration}$
e.g. Equations 1-4

Fig. 1. Development of Quality Focused Efficiency Metric.

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.

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. We introduce the objectives, limitations, and use cases of efficiency in ML, and establish basic efficiency types based on trade-off relationships. Drawing inspiration from the laws of physics, we define efficiency using *efficiency dimensions* for quality, work, space, load, and duration of the ML procedure, with each dimension comprising measurements of the ML process. We outline two types of metrics, namely 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. To enhance the significance of scores, a defined procedure is employed to adjust dimensional weights and perform sophisticated measurement

TABLE I
LIMITATIONS OF *Q*UALITY *CO*MPACT (QCO) METRIC

Aspect	Valid Range or Category	Balanced
Dataset	Labelled Text Samples, Size <256 MB	Yes[1]
ML-Task	Text Classification, NLP-Tasks[4]	No[2]
Classifiers	All ML Techniques	No[3]
Host-setup	Non-HPC, Non-GPU, RAM <128GB	Yes[1]
Training Duration	48h	Yes[1]
Calculation Amount	63P FLOPS, 800M Minor Page Faults	Yes[1]

(1) Compensation, e.g., efficiency remains consistent regardless of dataset size. (2) Scores from different tasks are not comparable. (3) Provides comparable efficiency scores per ML technique. (4) untested

smoothing. As an example, we outline the metric equation for quality-focused efficiency and propose a metric definition protocol to achieve metric validity. This protocol is applied to define the quality-focused efficiency metric, including the definition of the equation for score calculation, appropriate measurement selection, smoothing of measurement values, and dimension weight development.

A. Objectives & Limitations

An objective of this approach is to enable its applicability to all ML techniques. Measurements should be available for common host setups. The efficiency should be balance effects of different host setups. Additional requirements need to be derived from use cases. Certain measurements depend on ML task characteristics, such as dataset size, or runtime conditions, such as duration (see Table I). Valid measurement ranges may be enforced by smoothing techniques.

The application of the ML process involves objectives and constraints. The following use cases have specific efficiency requirements.

1) Effects of changes in the ML process: The effects of different techniques, such as preprocessing techniques, need to be measured [19].
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 [20].
4) Parameter optimisation: Effectiveness as a cost function in the optimisation of hyperparameters or setups [21].
5) Performance comparison: Compare the performance of an ML technique on different host setups to evaluate ML efficiency [22].
6) Predicting computational costs: Predicting the cost by predicting computational effectiveness of an ML technique in a production setup [23].

B. Dimensions

The concepts of efficiency in the use cases are different, but are defined on the basis of similar components. All concepts consider the trade-off between the performance of the model and the resources consumed during training or inference. The key components under consideration for the efficiency metrics are:

Accuracy or Performance. The efficiency of the machine learning model is correlated with its accuracy or performance. Standard metrics such as accuracy, precision, recall, F1-Score, or Area Under the ROC curve (AUROC) can be used to measure this, depending on the specific task.

Resource Utilisation. Efficiency should take into consideration the resources consumed during the training or inference process. This includes computational resources like CPU, GPU, or memory usage. Efficient models should minimising resource utilisation.

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. Efficiency is affected by the complexity of the ML process, or the amount of computation needed to train the model or compute a result for inference. Efficiency is improved when the computational effort is minimised.

Training Duration. The definition of efficiency can include the time to train the machine learning model. Faster training times can be beneficial, especially in scenarios where models need to be trained frequently or where time constraints exist.

Inference Latency. For 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 [24]. This approach uses abstract dimensions, which provides adaptation through flexible adaption. 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. Measurement can be conducted using appropriate evaluation metrics tailored to the specific task, including accuracy, precision, recall, F1-Score, or AUROC. Preferably scores that compensate unbalanced dataset [6].

Work. (Or Computational Effort, Computational Complexity) The number of computational operations, such as matrix multiplications, gradient computations, data transformations as well as 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 real workload differs due to optimisation at the software and hardware level. [7]–[9]. The measurement shall count generated and processed compute steps,

ISBN: 978-1-68558-097-1

optionally data transfers through memory and network. Computational steps can be counted direct (floating point operations or instructions) [6] or indirect by measuring side-effects of computation, e.g., memory management activity.

Load. (Or Relative Resource Consumption) Relative host usage reflects the *degree* to which all available resources on the host are being used. This includes relative usage of compute units (CPU and GPU cores), and relative memory usage. It also includes information about 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, needed by the machine learning process. Space usage of memory is measured 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, such as training time or inference latency and include time to complete the ML procedure or time spent on processing units.

Other non-dimension specific measures include the characteristics of the dataset, such as information about the number of samples and the size of the dataset. For special purpose metrics, sample attributes such as number of sentences, number of words and linguistic text attributes can be obtained.

C. Efficiency

Efficiency (Cost-Effectiveness) refers to achieving a high level of performance or accuracy while optimising the utilisation of resources and minimising associated costs. It aims to strike a balance between the effectiveness (performance) of the model and the costs or resources required to achieve that effectiveness. This approach covers three concepts of cost-effectiveness:

Solution Efficiency. Efficiency as the balance solution achievement and cost. Solutions are focuses like quality, costs include efforts done and resources consumed. Every aspect is provided by one or multiple efficiency dimensions. Solution efficiency with quality focus describes how much computational effort was used to achieve the 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 is achieving low latency of ML inference.

Resource Efficiency. Efficiency as the degree to which resources are used. Resource efficiency is the capability of the ML procedure to use all available resources. It increases by adapting to host setup by using more existing resources. Important for designing hardware for specific ML Techniques and adapting ML algorithms to specific hardware [25]

TABLE II
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

TABLE III
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

Synthetic Efficiency. Efficiency as a tool for measuring special aspects of performance to analyse specific attributes, such as text quality indicators [26] or performance comparisons [27].

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, 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.

Beside the efficiency objectives, two diametral requirements on handling of the ML efficiency results are encountered: Interpretability and Usability. The more 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

Efficiency in the field of ML shows variability depending on the specific application. Metrics are proposed for specific purposes and categorised according to their level of complexity. The group of compact metrics uses a subset of dimensions that contribute to the calculation of an efficiency score, which is determined with respect to the dominant dimension. The compact efficiency (CO) metric is defined in 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 (1); based on efficiency dimensions (D) quality q, work w, space s, load l and duration d with specific dimension weights β defined by (2).

$$[F]CO(M) = (F \times \alpha) \times U \tag{1}$$

$$\begin{aligned}[F]CO(M) = {} & (q_M \times \beta_q) \times (w_M \times \beta_w) \times (s_M \times \beta_s) \\ & \times (l_M \times \beta_l) \times (d_M \times \beta_d) \times \psi \end{aligned} \tag{2}$$

$$\begin{aligned} \text{where } & D = \{r \in \mathbb{R} \,|\, r > 1\} \; and \; \{q, w, s, l, d \in D\} \\ & F \subseteq D \quad \text{and} \quad U = D \backslash F \\ & \psi = \text{Score-Compensation} \end{aligned}$$

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 (1) & (2) for the Quality-Focus, as stated by (3). The quality dimension is represent 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 measures the amount of 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 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). The dimensions are intended to increase in importance with a growth proportional to their current size, so the weights β of the dimensions and the focus weight α are treated as exponents with the respective dimension as the base.

TABLE IV
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 [28], DistilBERT [29], torch [30], pandas [31].

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

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 of 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.

IV. COMPACT METRIC INSTANTIATION

In order to use the proposed efficiency metric, the abstract definitions need to be instantiated into explicit definitions by empirical method (Table II). This requires conducting an ML experiment that maps a specific use case and involves the collection of measurements. The instantiation of the metrics thus depends on the parameters of the experiment. The validity of the metric instantiation is positively correlated with the size of the use case, such as the number of datasets.

The instantiation stages for a CO-Metric are as follows: The experiment is designed, deployed and measurements captured.

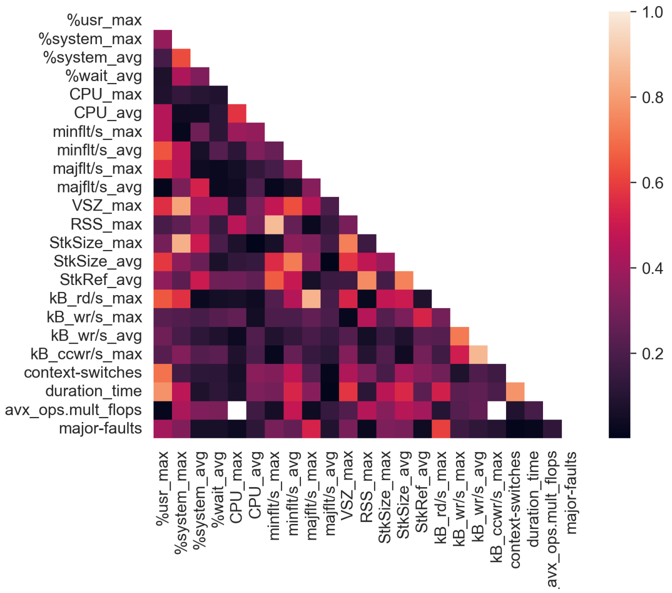

Fig. 2. Pearson Correlation Coefficients of empirical Measurement Values.

TABLE V
MEASUREMENTS

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

The formulas for the dimensions are defined and measured values are assigned (Table VI). Validity ranges are specified and the measured values are smoothed accordingly (Table VI). The determination of the dimensional weights and the score compensation factor ψ completes the Metric Equation (4).

For reasons of compactness, the instantiation is restricted to QCO metric.

A. The QCO-Metric Instance

The dimensions and the QCO score are instantiated according to the protocol given in Table II.

Use Case 1 requires efficiency as quality per work, space, and time. The QCO-Score is applicable. Experiment 1 has been set up based on Use Case 1 to instantiate a Quality Focused Metric. The required validity for different datasets and ML procedures results in the empirical variance requirements presented in Table III. To gain comparison validity among host-setups, four different computing environments were set up (No. 1-4 as shown in Table IV).

TABLE VI
QCO 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.

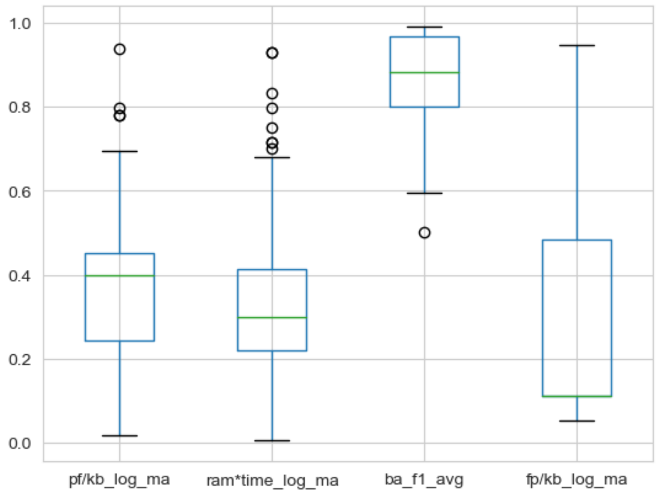

Fig. 3. Spread and Skewness per Dimension after logarithmic smoothing.

Two datasets were selected, a spam classification [32] and am movie review classification [33]. Vectorization was done using the classical TF-IDF algorithm as wells as by word embedding based on BERT. The classifiers chosen were Support Vector Machines, Naïve Bayes (NB), Gradient Descent (GD) and Random Forest (RF). In addition, a transformer-based ML procedure (DistilBERT-Setup) was performed. In the DistilBERT-Setup, the model was fine-tuned on the datasets and used for vectorization and classification.

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).

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 IV) shows Pearson Correlation Coefficients for selected measurements. `perf` was not supported on all host setups due to missing performance counters and conflicts with power saving methods. Two sets of measurements has to be set up which results in two QCO flavors: Floating Point Operation (QCO_F) based and Minor Page Fault (QCO_F) based. The selected measurements are listed in Table V.

Range definition is necessary for normalisation. Valid ranges for this QCO-Instance are listed in Table I. Normalisation is necessary as different units and types of data are used in calculation. Performing monotonic data transformation on dimensions values lead to a range between 0 to 2. The transformation range is based on the maximum values per

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

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

$$\begin{aligned}
\text{where } F1 &= \text{F1-Score, } BACC = \text{Balanced Accuracy Score,}\\
FLOPS &= \text{Floating Point Ops., } MPF = \text{Minor Page Faults,}\\
DS &= \text{Dataset-Size, } RSS = \text{Resident Set Size,}\\
D &= \text{Duration, } TOC = \text{Time on CPU}
\end{aligned}$$

dimension. The valid ranges are thus not related to measurement ranges. The definition of valid measurement ranges (Table I) enables data transformations on measurement values. After transformation values are in an closed scale with minor decreased distribution (Figure 3).

The dimension-equations are defined by interpreting the dependencies of the measurements (Table V). 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 for is defined for each measure set, which results in 4 and 5

B. 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 I.
2) Perform training on a dataset subset while capturing measurements according Table VI.
3) Calculate efficiency by equations 4 5.

C. 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).

V. Evaluation

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

A. Experiments

In Experiment 2, binary classification tasks 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)[32] and Movie Survey Classification (7.805 samples) [33]. The experiments were run on host 1 (Table IV) in two virtual hosts with different virtualisation technologies. The results of experiment 2 are shown in Table VII. To compare the QCOF and QCOP metrics in Experiment 2, two set of QCO had to be created as some FLOPS measurement 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 [29]), 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 [32], which was run on Host 5 (IV).

B. Usability

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

TABLE VII
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

Cloumns: Dataset, Vectorizer, Classifier, Duration, Quality, Time, Space, WO_F= Work (FLOPS), WO_P = Work (Minor Page Faults), QCO Metrics, Rankings by Domain EXPerts, or QCO, SMS = SMS Spam Dataset[32], IDB = IMDB Dataset[33], 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

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 VIII). 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 VII 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 relevance, which is particularly evident in the DistilBERT setups.

D. Balance & Compensation

QCO achieved a balance of aspects through compensation (Table I). 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.

TABLE VIII
Effiency 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)

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 metric framework have been defined to address different real-world scenarios and use cases. The proposed efficiency metrics provides information for identifying the optimal ML technique and hyperparameters, selecting ML techniques for limited resources or private data, comparing classification performance across different host setups, and estimating computational costs. The metric framework introduces several dimensions that collectively capture the efficiency of ML techniques to achieve these goals. These dimensions include quality, work, load, space and duration, each of which contributes to the overall efficiency score. The dimensions are designed to measure different aspects of ML performance and resource utilisation, allowing for a comprehensive evaluation. One of the key advantages 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 requires 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.

Overall, the proposed efficiency metrics framework offers a promising approach for quantifying and comparing the cost-effectiveness of machine learning methods. By providing a comprehensive view of efficiency across multiple dimensions, it enables researchers and practitioners to make informed decisions regarding ML techniques, resource allocation, and model performance optimisation. The adaptability and applicability of the metrics in different contexts make them a valuable tool for advancing the field of ML and facilitating the development of efficient and effective ML models.

VII. Conclusion and Future Work

The successful calculation and evaluation of efficiency scores will pave the way for further achievements in the efficiency of machine learning research. By introducing complex dimensions that take into account measurement correlations, such as FLOPS to data volume or memory usage to duration time, we were able to potentially balance the metric and achieve a compensation of dataset size and host-setup.

When evaluating the QCO dimensions, we encountered a limitation due to insufficient samples. However, the FLOPS-based instance showed consistency and our attempt to use a small page fault measure to support the work dimension showed partial success. To gain further insight into efficiency correlations, future work could focus on dimensions that incorporate ML-specific attributes, such as model size.

It should be noted that the validation methods of the proposed metric currently rely on peer opinion. While this provides valuable insights, we recognise the importance of statistical validation to increase the credibility and robustness of the metric.

The efficiency of machine learning methods is undoubtedly influenced by expert opinion and the relevance of quality to the specific application. However, we need to be aware of the potential exponential increase in complexity if quality is used as the only guiding principle for development. Striking a balance between different efficiency dimensions is crucial to ensure a practical and rational approach to optimising machine learning processes.

Further research and statistical validation will contribute to the refinement and wider adoption of these efficiency metrics, ultimately advancing the field of ML and facilitating the development of efficient and effective machine learning models.

Acknowledgments

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

References

[1] 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.

[2] K. Raza, V. Patle, and S. Arya, “A review on green computing for eco-friendly and sustainable it,” *Journal of Computational Intelligence and Electronic Systems*, vol. 1, no. 1, pp. 3–16, 2012.

[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.

[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.

[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.

[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.

[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.

[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*, IEEE, 2017, pp. 1916–1920.

[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*, Springer, 2009, pp. 399–407.

[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*, Springer, 2022, pp. 39–54.

[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*, PMLR, 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.

[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.

[19] 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.

[20] 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)*, IEEE, 2020, pp. 460–465.

[21] M. Feurer and F. Hutter, “Hyperparameter optimization,” *Automated machine learning: Methods, systems, challenges*, pp. 3–33, 2019.

[22] S. González-Carvajal and E. C. Garrido-Merchán, “Comparing bert against traditional machine learning text classification,” *arXiv preprint arXiv:2005.13012*, 2020.

[23] C. Zhang, M. Yu, W. Wang, and F. Yan, “{Mark}: Exploiting cloud services for {cost-effective},{slo-aware} machine learning inference serving,” in *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, 2019, pp. 1049–1062.

[24] 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.

[25] 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.

[26] C. Kiefer, *Quality indicators for text data*, BTW 2019 – Workshopband, 2019. DOI: 10.18420/btw2019-ws-15.

[27] 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.

[28] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[29] 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.

[30] R. Collobert, K. Kavukcuoglu, and C. Farabet, “Torch7: A matlab-like environment for machine learning,” in *BigLearn, NIPS Workshop*, 2011.

[31] W. McKinney, “Data Structures for Statistical Computing in Python,” in *Proceedings of the 9th Python in Science Conference*, S. van der Walt and J. Millman, Eds., 2010, pp. 56–61. DOI: 10.25080/Majora-92bf1922-00a.

[32] 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.

[33] 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.