Which Design Decisions in AI-enabled Mobile Applications Contribute to Greener AI?

Roger Creus Castanyer
Universitat Politècnica de Catalunya
Barcelona, Spain
creus99@protonmail.com

Silverio Martínez-Fernández
Universitat Politècnica de Catalunya
Barcelona, Spain
silverio.martinez@upc.edu

Xavier Franch
Universitat Politècnica de Catalunya
Barcelona, Spain
xavier.franch@upc.edu

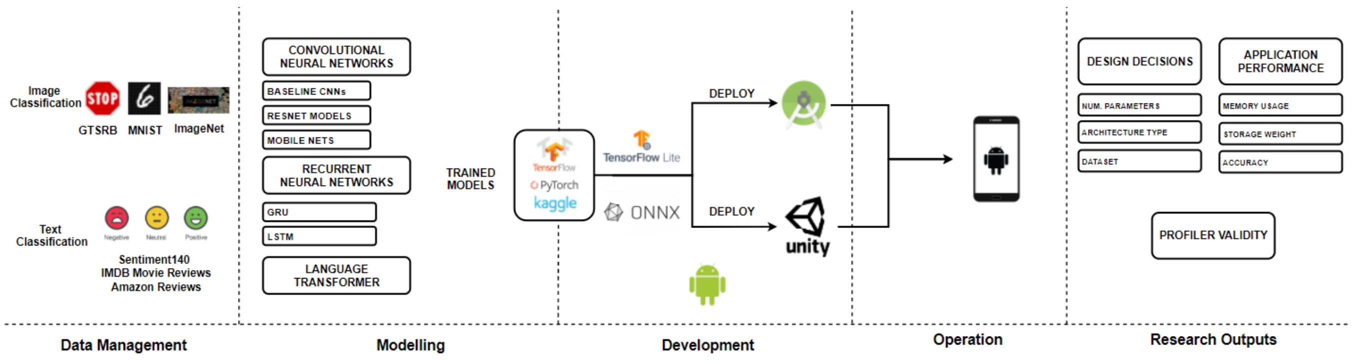

Figure 1: Schema of our empirical study in the end-to-end software engineering lifecycle for AI-enabled applications. In the *(i)* Data Management phase we obtain and preprocess the data sources; in the *(ii)* Modelling phase we train the AI models for solving the defined tasks; in the *(iii)* Development phase we convert and deploy the trained models in the applications that we build; in the *(iv)* Operation phase we make use of the applications and monitor their performance within the operation settings by means of profilers; finally, we provide answers to the RQs.

ABSTRACT

Background: The construction, evolution and usage of complex artificial intelligence (AI) models demand expensive computational resources. While currently available high-performance computing environments support well this complexity, the deployment of AI models in mobile devices, which is an increasing trend, is challenging. Mobile applications consist of environments with low computational resources and hence imply limitations in the design decisions during the AI-enabled software engineering lifecycle that balance the trade-off between the accuracy and the complexity of the mobile applications.
Objective: Our objective is to systematically assess the trade-off between accuracy and complexity when deploying complex AI models (e.g. neural networks) to mobile devices, which have an implicit resource limitation. We aim to cover *(i)* the impact of the design decisions on the achievement of high-accuracy and low resource-consumption implementations; and *(ii)* the validation of profiling tools for systematically promoting greener AI. In this way, we aim to provide a quantitative analysis of the performance of AI-enabled applications in operation with respect to their design decisions.
Method: This confirmatory registered report consists of a plan to conduct an empirical study to quantify the implications of the design decisions on AI-enabled applications performance and to report experiences of the end-to-end AI-enabled software engineering lifecycle. Concretely, we will implement both image-based and language-based neural networks in mobile applications to solve multiple image classification and text classification problems on different benchmark datasets. For that, we make use of the Android Studio and Unity3D frameworks for building mobile applications that work over Android; and TensorFlow Lite and Open Neural Network Exchange (ONNX) file formats for deploying the AI models in the applications. Overall, we plan to model the accuracy and complexity of AI-enabled applications in operation with respect to their design decisions and will provide tools for allowing practitioners to gain consciousness of the quantitative relationship between the design decisions and the green characteristics of study. Additionally, we will provide experiences in the end-to-end AI-enabled software lifecycle and discuss the challenges found, tools and practices for practitioners. Finally, we will provide an open-source data repository following the ESEM open science practices and containing all the experimentation, analysis and reports in our study.

CCS CONCEPTS

• **Software and its engineering** → *Software creation and management*; • **Computing methodologies** → *Knowledge representation and reasoning*.

KEYWORDS

AI-enabled Applications, Mobile Applications, Model Accuracy, Application Performance, Greener AI, Neural Networks

1 INTRODUCTION

Artificial Intelligence (AI) plays a key role in the world we live in. AI is about making machines mimic intelligent behaviour. Machine Learning (ML) sets the foundations of AI [34]. ML works over the most valuable and key source of knowledge that exists: data. In ML, machines take in data and learn patterns that would be difficult for humans to learn. ML is valuable in the sense that the data processing (or data classification, segmentation, representation) capabilities go far beyond than what humans can achieve. Moreover, ML finds its extension in Deep Learning (DL) which introduces complex neural network (NN) models with the purpose to increase the domains in which AI can provide great capabilities [26]. NNs are versatile models with sophisticated architectures that have been applied successfully to domains that traditional ML models cannot tackle (i.e. image-based and language-based contexts). Nowadays, NNs have achieved outstanding results and have outperformed humans in domains as diverse as machine translation, character and handwriting recognition, speech and facial recognition and videogames [30, 36, 43, 47]. Key to the success of NNs in these domains is their capability of learning complex data representations in an unsupervised manner. In this way, NNs do not demand expert feature engineering tasks, which are a bottleneck of the ML models performance. However, training NNs requires both large amounts of data and high computational resources. Anyway, the latter does not become a stopper for NNs since we are living the *Big Data, Big Compute, Big Models* revolution [4, 5, 17]. In this way, following the exponential evolution of the availability of data and computational power, the tendency of solving complex tasks with complex models is increasing [4, 15, 39, 48]. Arguably, the latter does not consider resource consumption regulations, and does not promote less data-intensive and lighter models, which sets the bases of greener AI.

For green AI we understand AI research that is more environmentally friendly and inclusive [41]. In contrast, red AI refers to AI research that seeks to improve accuracy (or related measures) through the use of massive computational power while disregarding the cost [41]. Ensuring efficient implementations is key to achieve greener AI. Some of the key practices of green AI are to measure and lower the environmental impact of AI [24], regulate the resource consumption of the AI-enabled applications [13] and balance the effectiveness and efficiency of the AI models [40, 46]. Key to these practices is the capability of measuring and monitoring the efficiency-related metrics of the AI-enabled applications in operation, which allows the developers to gain awareness of the performance of the applications with respect to their complexity.

Deploying AI models to operate in environments with low computational resources constraints the design decisions of AI-enabled solutions. One of such low computational resources ecosystem are mobile applications. With the growth of the mobile applications market, approaches on the integration of NNs in lightweight devices are becoming popular given their ability to provide great services to the user [7]. In the context of mobile applications, several works have put the focus in improving energy efficiency by understanding the best coding practices (i.e. design decisions) [1, 2, 11] and profiling (i.e. monitoring) the applications performance [9, 10, 16]. Monitoring the energy consumption, which has been studied from the software and hardware viewpoints, is not trivial [12, 16]. More recently, researchers have found that software energy consumption can be modelled (i.e. estimated) accurately [8, 16]. The CPU usage information has demonstrated to be very valuable for reaching accurate energy consumption estimations [8].

Balancing the efficiency and effectiveness of AI-enabled applications is key to pursuit greener AI. In the context of AI-enabled mobile applications, reducing the energy-consumption, memory usage (i.e. in CPU or GPU) and storage weight of the AI models while providing high-accuracy results is key for deploying a user-accommodated experience [42]. Engineering AI-enabled software that operates in edge devices is challenging [6, 7, 28, 31]. There are four main cycles in the AI-enabled software lifecycle [27]. First, the *Data Management* which consists of the collection and processing of raw data. Second, the *Modelling* of the AI components which consists of adjusting different models to obtain the best-performing possible solution. Third, the *Development* of the environment and the integration of the AI model in it. Fourth, the *System Operation* of the AI-enabled applications. This lifecycle introduces concerning challenges that are not usual in traditional software engineering [37]. Data centricity is what increases the risk in many quality attributes (i.e. adaptability, scalability, explainability, privacy) [28].

Also, there is few availability of frameworks that support the end-to-end implementation of AI-enabled applications, and they have a strong influence in the overall applications performance [19]. For this reason, there exists the need of a systematic approach for measuring the performance of an AI-enabled application that takes into account the accuracy and complexity-related metrics.

For this reason, we aim to model and investigate the contribution of the design decisions during the AI-enabled software engineering lifecycle to the overall application performance in terms of accuracy and complexity. With this, our objective is to report statistically significant relationships between the design decisions and their impact in order to build the bases for predictive tools than empower practitioners with consciousness of the performance of their AI-enabled applications.

2 EXPERIMENTAL DESIGN

2.1 Research Objectives

In the following, we define our goal and Research Questions (RQs), following the GQM guidelines [3].

Overall, our goal is to analyse *the design decisions (number of parameters, architecture type, technology, device) of AI-enabled applications* with the purpose to *assess their impact* with respect to *green characteristics (performance, memory usage, storage weight, accuracy)* from the point of view of a *developer* in the context of *applications with neural networks for image and language recognition.*

We determine the units of analysis and formulate the RQs as follows. We aim to *(i)* evaluate and quantify the implications of each of the design decisions along the AI-enabled software lifecycle with respect to the overall applications performance (i.e. accuracy and complexity); and *(ii)* leverage profiling tools for measuring the performance and support the decision-making in the design of AI-enabled applications.

We establish research objectives, variables and hypotheses from which we will perform a quantitative and empirical study. In particular, we study the systematic variation of design decisions to measure their influence on the accuracy and complexity-related metrics in an experimental set-up. For this purpose, we will implement and monitor AI-enabled applications in the vision and language domains, and we will follow the guidelines proposed by

Jedlitschka *et al.* [22]. Concretely, we investigate our implications separately among image-based and text-based problems. For that, we adopt two technology stacks to accomplish the implementation of the AI-enabled applications, and we aim to provide experience and discuss the best tools and frameworks for achieving it. Then, we profile the performance of the applications by taking into account their memory usage, storage weight and the accuracy in operation. Finally, we define linear regression equations for measuring the implications of the design decisions to the trade-off between accuracy and complexity in the applications. In this sense, our research objectives consist on finding statistically significant relations between dependent and independent variables. As no statistical method can show causation, we aim to provide a well-suited regression analysis for minimizing the error in the causal interpretation of our models [33]. For this reason, we establish the RQs as follows.

- RQ1 - What is the impact of the design decisions along the AI-enabled software lifecycle to the performance of AI-enabled applications in terms of accuracy and complexity?
  - RQ1.1 - How is the memory usage of the AI models affected by their number of parameters, architecture type and dataset used?
  - RQ1.2 - How is the storage weight of the AI models affected by their number of parameters, architecture type and dataset used?
  - RQ1.3 - How is the accuracy of the AI models affected by their number of parameters, architecture type and dataset used?
- RQ2 - Is profiling a suitable tool to systematically analyze the trade-off between accuracy and complexity of AI-enabled applications?

With *RQ1* we pretend to validate our criterion for reasoning about efficient implementations of NNs in software applications. We aim to provide a quantitative analysis of the importance and relation between design variables and performance metrics in terms of accuracy and complexity. Concretely, we aim to deploy a quantitative evaluation of the performance of the AI-enabled applications in operation. With this, our goal is to delineate the differences in performance caused by the NNs configuration (i.e. architecture type, number of parameters) and to validate them across the vision and language domains. Some examples of popular architecture types for image-based NNs are the CNN, ResNet [20] and MobileNet [21]. Then, the number of parameters represents the total sum of weights and biases that operate inside the model architecture. The latter controls the degree of complexity introduced in the models and correlates with the amount of computational power provided. Arguably, P and AT together are the core model-related design decisions of any AI model.

In comparison with *RQ1*, we establish *RQ2* as a non-causal RQ to determine whether profiling tools can become a standard approach for systematically reasoning and evaluating the performance of AI-enabled applications. For that we evaluate our subjective empirical experience when collecting and profiling data and running statistical tests on it to derive our conclusions. As we frame our study to the statistical modelling of the green characteristics from a set of design decisions, the validity of our hypotheses aim to be independent of whether we prove an existent relationship between our objects of study or we discard it. In this way, we deny any confirmation bias related to prior beliefs of ours on the possibly existing relations between the variables in the study. Moreover, we involve multiple authors in the analysis of the AI-enabled applications, so we aim to provide repeated experimentation to generalize our conclusions and remove personal interpretation biases.

2.2 Variables

In the following subsections, we respectively report the independent, dependent, and confounding variables of our experimental design.

2.2.1 *Independent Variables.*

In this study, we define multiple independent (i.e. explanatory) variables in order to evaluate the contribution of the design decisions to the overall applications performance.

Regarding the data management phase of the AI-enabled software lifecycle, we experiment with a set of different datasets for each of the vision and language domains. In this way, we define the dataset (D) categorical variable indicating which dataset is used for training the models. Implicitly, each dataset provides different quantity and quality of data and this certainly affects the performance of the models.

Regarding the modelling phase of the AI components, we experiment with *(i)* different number of parameters (P), which is a numerical variable indicating the complexity of the AI models; and *(ii)* different architecture types (AT), encoded as a categorical variable characterizing different AI-based solutions.

2.2.2 *Dependent Variables.*

We evaluate the contribution of the design decisions with respect to the overall performance of the built applications in operation. To define the applications performance we consider three dependent variables:

- Memory usage (M), which quantifies the resource consumption that the applications require to run (in ms).
- Storage weight (S), which quantifies the cost of deploying the AI models to the applications and storing them (in MB).
- Accuracy (A), which measures how capable is the application (and the AI model) of providing high-accuracy and satisfactory results in operation.

Note that we define M in ms as a variable that encodes temporal information. Conceptually, the memory usage (i.e. memory resource consumption), which is usually given in MB, is directly proportional to the time consumption in a processing unit (i.e CPU or GPU). That is, for the same operation, a processing unit both uses an amount of memory (in MB) and spends an amount of time to solve it (in ms), so both metrics are directly related. Taking the latter into account, we use the temporal information in ms to give a user-oriented interpretation of the memory usage as the AI-enabled applications real-time performance.

Regarding accuracy, we plan to measure the accuracy of the AI models when operating in the AI-enabled applications during the system operation phase of the AI-enabled software lifecycle. That is, we test the accuracy of the models with data outside the context of the training datasets to test the models' generalization capabilities. Concretely, we will craft a test dataset for each domain (image and text) containing data from the operation environment with a sufficiently large number of examples to test the applications systematically and provide statistically significant conclusions on the accuracy achieved. For the image domain we plan to report the top-3 accuracy (as we consider that the top-1 accuracy might be too demanding for large multi-class classification problems), and for the language (i.e. text) domain we will report the top-1 accuracy as text classification is generally a problem with a smaller number of classes.

2.2.3 Confounding Variables. We identify a confounding factor related to the experience and knowledge of the developer who builds the AI models and applications, since it conditions the overall performance of the AI-enabled applications in terms of their accuracy and complexity. A developer is responsible for the implementation of both multiple sophisticated AI models and mobile applications that integrate them. Moreover, it is also in charge of operating with the AI-enabled mobile applications, profiling and analyzing the applications' performance during the system operation phase. We consider that the more experience and knowledge the developer has, the less overhead in the AI-enabled applications. Ideally, as we aim to profile the performance of AI-enabled applications, we want to account only for the performance changes produced by our objects of study and not by deficient implementations. For mitigating the impact of the latter we will provide the simplest possible implementations.

2.3 Hypotheses and Analysis Operationalization

In this study there is a relation that we aim to characterize between the design decisions of AI-enabled software and its consequences. Concretely, we study the accuracy (A) and the complexity (M, S) as a model of the design decisions (P, AT, D). We do so in the vision and language domains and we model our hypotheses separately among these. The models that we consider are linear models between the design decisions (i.e. independent, explanatory variables) and the accuracy and complexity-related metrics (i.e. dependent, response variables). When fitting linear models we provide the following capabilities:

- Determining the goodness of fit by quantifying the amount of variability of the dependent variables that is captured (i.e. explained) by the set of independent variables.
- Using the fit as a predictive model for regressing accuracy and complexity responses for untested experimental conditions.
- Quantifying the strength of the relationship between the dependent and independent variables, and determining whether some independent variables may have no linear relationship with the responses at all, or identifying which subsets of independent variables may contain redundant information about the responses.
- Providing directional analysis on what independent variables contribute more to the target dependent variables.
- Determining whether the interactions between pairs of dependent variables significantly contribute to the independent variables.

With all this, we define the models as follows:

$$M = \beta_{P_1} P + \beta_{AT_1} AT + \beta_{D_1} D + \epsilon_1 \quad (1)$$

$$S = \beta_{P_2} P + \beta_{AT_2} AT + \beta_{D_2} D + \epsilon_2 \quad (2)$$

$$A = \beta_{P_3} P + \beta_{AT_3} AT + \beta_{D_3} D + \epsilon_3 \quad (3)$$

With equations 1, 2, 3 we define multiple linear regressions to explain the variability of the memory usage, storage weight and accuracy with respect to the design decisions separately (1: M; 2: S; 3: A) [25]. In this sense, with β_{P_1} we study the impact of the number of parameters only to the M. Ideally, we want all the variables ϵ_i to be of low magnitude, as it represents the unexplained variance of the dependent variables.

We fit each of the four multiple linear models with the least-squares estimation technique. In particular, we fit the coefficients β_i (for each of the design decisions) to minimize the squared error between model predictions and observed values (belonging to the data collected in our experimentation).

$$\hat{\boldsymbol{\beta}}_{\boldsymbol{i}} = \underset{\boldsymbol{\beta}_{\boldsymbol{i}}}{\arg\min} \sum_{n=1}^{N} (\boldsymbol{\beta}_{\boldsymbol{i}} * \mathbf{x}_n - \mathbf{y}_{n_i})^2$$

Where $\hat{\boldsymbol{\beta}}_{\boldsymbol{i}}$ is the vector of fitted coefficients of the *i-th* model, N is the number of experiments that we perform, $\mathbf{x}_n$ are the conditions of the *n-th* experiment and $\mathbf{y}_{n_i}$ is the observed response variable i that we profile in the *n-th* experiment (following the association of i = 1: M; 2: S; 3: A).

Then, we first test whether equations 1, 2, 3 are suitable models of the aforementioned relations. We do so with an F-test for each of the four multiple linear regressions. We define the hypotheses of the *i-th* F-test as follows, for i = 1: M; 2: S; 3: A.

$$\mathcal{H}_{0_i} : \beta_{P_i} = \beta_{AT_i} = \beta_{D_i} = 0 \qquad \text{(Null Hypothesis)}$$

$$\mathcal{H}_{1_i} : \exists \beta \in \{\beta_{P_i}, \beta_{AT_i}, \beta_{D_i}\} : \beta \neq 0 \qquad \text{(Alternative Hypothesis)}$$

In this sense, we first test whether there is a linear relation between each of the independent variables and the dependent ones. In the F-test we compare the fitted model sum of squares against the residual sum of squares, and calculate the value of the F-statistic to derive the p-value associated to a level of confidence and to reject/support the null hypothesis [38].

Furthermore, we can evaluate how valuable the fitted models are with the adjusted R^2 statistic [32]. In our multiple linear regression models the R^2 is the square of the multiple correlation coefficient, which measures how well a response variable can be predicted with a linear function of a set of independent variables.

We also formulate an hypothesis for each explanatory variable in each of the models of the independent variables ($\mathcal{H}_{P_i}, \mathcal{H}_{AT_i}, \mathcal{H}_{D_i}$) that concerns the statistical significance of each design decision in each of the specific models. Hence, we have two hypotheses for each model which sums to a total of another 9 hypotheses. With these, we aim to separately answer the research questions RQ1.1, RQ1.2 and RQ1.3. Each of these hypotheses can be tested with an adjusted t-test, which is an inferential statistic used to determine if there is a significant difference between the means of two groups [45]. Concretely, we aim to determine whether the coefficients encoding the contribution of each of the design decisions to the accuracy and complexity-related metrics are significantly different from zero. If the fitted coefficient corresponding to a design decision (i.e. of

Table 1: The variables of the study

Class	Name	Abbreviation	Description	Scale	Operationalization
Independent	Number of Parameters	P	The number of trainable parameters of the NNs	numerical	See Sections 2.2 and 2.3
	Architecture Type	AT	The architecture that defines the NNs functionality	nominal	See Sections 2.2 and 2.3
	Dataset	D	The dataset used for training the AI models	nominal	See Section 3
Dependent	Memory usage	M	The time that the AI-enabled applications need to answer a query	numerical	Profiled
	Storage Weight	S	The cost of storing the AI models in the edge devices	numerical	Retrieved from the built applications
	Accuracy	A	The precision of the AI models when answering queries in the edge devices	numerical	Measured in the system operation phase (see Section 2.2.2)
Confounding & others	Developer experience		The ability to implement high-quality software applications	-	-
	AI modelling framework		Technology used for modelling the AI components	-	Experiment with Pytorch and Tensorflow
	Export AI components		Technology used for exporting the AI components	-	Experiment with ONNX and Tensorflow Lite
	AI-enabled development framework		Technology used for developing the AI-enabled applications	-	Experiment with Unity3D and Android Studio

the number of parameters of the AI models - P) happens to be significantly different from zero and take a value y in a specific model i, we can state that if we fix all the other non-zero coefficients of model i, then for each change of 1 unit in P, the response variable of model i changes y units.

$$\mathcal{H}_{P_{0i}} : \beta_{P_i} = 0 \qquad \text{(Null Hypothesis)}$$
$$\mathcal{H}_{P_{1i}} : \beta_{P_i} \neq 0 \qquad \text{(Alternative Hypothesis)}$$
$$\mathcal{H}_{AT_{0i}} : \beta_{AT_i} = 0 \qquad \text{(Null Hypothesis)}$$
$$\mathcal{H}_{AT_{1i}} : \beta_{AT_i} \neq 0 \qquad \text{(Alternative Hypothesis)}$$
$$\mathcal{H}_{D_{0i}} : \beta_{D_i} = 0 \qquad \text{(Null Hypothesis)}$$
$$\mathcal{H}_{D_{1i}} : \beta_{D_i} \neq 0 \qquad \text{(Alternative Hypothesis)}$$

These hypotheses test whether a specific independent variable (i.e. P) is significantly adding to the variability of the *i-th* variables of study given that the *i-th* model also considers other explanatory variables (i.e. AT, D).

With all this, we pretend to *(i)* determine whether the accuracy and complexity of an AI-enabled application can be modelled by taking into account the design decisions; *(ii)* investigate the contribution of each design decision, analyse their interactions, and determine the most impactful design decisions on the overall performance of the applications.

For this reason, we explore a complete set of representative combinations of experimental settings. Concretely, we fix all the design decisions and iterate variations on a single one (and we do this for each of the design decisions) to consistently quantify the contribution of each of the design decisions.

3 DATASETS

There exists a wide variety of NN models (i.e. recurrent, convolutional, Transformer-based) for tackling different problems (i.e. image-based, language-based). To provide generalist conclusions about AI-enabled applications we aim to study a complete set of representative NN architectures in the vision and language domains. Also, we want to test the aforementioned architecture types in multiple benchmark datasets to provide a model and data-centric context. For this reason we make use of the German Traffic Sign Recognition Benchmark (GTSRB) [44], MNIST [14] and Cifar-10 [23] datasets for solving image classification, and the Sentiment140 [18], Amazon Reviews [35] and IMDB Movie Reviews [29] for solving text classification.

For image classification, the GTSRB dataset is the output of an attempt to benchmark traffic sign recognition and it contains 51,840 images representing 43 unique traffic sign classes. MNIST is a benchmark dataset of handwritten digits (i.e. contains 10 unique classes belonging to the digits) containing 60,000 training examples, and a test set of 10,000 examples. Cifar-10 is a generic dataset for image classification containing 60000 32x32 colour images in 10 classes, with 6000 images per class.

For text classification, Sentiment140 contains 1,600,000 tweets annotated with either *positive* or *negative* labels, which encode the sentiment of the tweets. Then, the Amazon Reviews for Sentiment Analysis[1] contains 4 million reviews also tagged either as positive or negative, and the IMDB Movie reviews[2] dataset contains 50,000 reviews also for binary sentiment analysis.

[1]https://www.kaggle.com/bittlingmayer/amazonreviews
[2]https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

4 EXECUTION PLAN

In the following we characterize our execution plan for answering the RQs. Figure 1 summarizes the execution plan. Overall, we pretend to develop an empirical study to provide a complete and generalist experimentation of the performance of software applications that integrate NNs.

Data Management: We make use of representative benchmark datasets for image, text and audio data. We collect, preprocess and clean the data for allowing suitable NN adjustments.

Modelling: We make use of Tensorflow and Pytorch for modelling the AI components. In particular, we train and compare convolutional, ResNet-based and MobileNets for solving image classification; and recurrent (GRUs, LSTM) and Transformer NNs for solving text classification; We believe that the aforementioned models cover a complete set of representatives of the current state-of-the-art for vision and language modelling with NNs. We will train all these models thanks to the Kaggle GPUs service. Concretely, we will train a set of hand-crafted NNs (i.e. baseline convolutional, recurrent NNs) and also leverage pre-trained and fine-tuned architectures (i.e. ResNet and MobileNets).

Development: We will experiment with *Android Studio* and *Unity3D* and leverage the capabilities of these engines to integrate AI components in the applications that we build. Also, we make use of TensorFlow Lite and ONNX for converting the trained models to serialized formats in order to deploy them in the platforms of study. Concretely, we use TensorFlow Lite and ONNX for deploying the models trained with Tensorflow and Pytorch respectively. With this, we integrate the TensorFlow Lite models to Android Studio which is the only framework supporting this file format, and the ONNX models to Unity3D for comparing other framework alternatives.

Operation: We deploy the AI-enabled applications to Android mobile devices and operate with them in the real-world context in order to evaluate their performance systematically and provide answers to our RQs. We involve multiple authors (together with multiple mobile devices) in the operation, profiling and analysis of the AI-enabled applications.

Execution of the experimental design: We systematically iterate variations in each of the design decisions separately while fixing the other ones to provide consistent conclusions.

Data collection during the execution of the empirical study: We profile the performance of the AI-enabled applications in operation. We make use of the two profilers provided by Tensorflow[3] and Unity[4] to measure the performance of the AI-enabled applications in operation. Concretely, we retrieve the time consumption in the processing units, which is directly proportional to memory consumption in MB. We also obtain the storage weight as a static property of the built AI-enabled applications, and measure their accuracy by averaging a wide range of queries in operation. With this, we investigate the implications of the design decisions to the applications performance separately among the vision and language domains. In particular, we craft an analysis dataset representing the design decisions taken in each experiment and their profiled performance in terms of accuracy and complexity.

Data analysis: We adjust the linear models described in Section 2.3 and study their validity, goodness of fit and marginal contribution of each of the variables. Then, we compare the results between vision and language domains in order to draw the conclusions.

5 THREATS TO VALIDITY

As with any empirical study, there might be limitations to our study design. Next, we report a few mitigation actions taken during the design of this protocol.

Regarding construct validity, the extent to which the theory's constructs are correctly operated in the experiment, we build convolutional (i.e. baseline, ResNet, MobileNet), recurrent (i.e. GRUs, LSTM) and Transformer-based NNs in order to mitigate mono-operation bias. Also, we measure the frameworks used for developing the AI models, exporting them and developing the AI-enabled applications to account for the performance that different technologies might offer. Furthermore, we also measure our experimentation in multiple mobile devices in order to find if their different capabilities also affects our variables of study.

As for conclusion validity, the ability to draw correct conclusion between treatment and outcome, we define and fit linear models to approximate the relation between design decisions and accuracy and complexity-related metrics. Hence, we estimate the models in order to minimize the error when fitting the collected experimental data with linear relations. In this way, the validity of our statistical tests is limited to the amount of variability that is explained with linear relations only (which we quantify and report). Also, the quality of the built AI-enabled applications depends on the experience of the developers and influences their complexity. For mitigating the latter we will provide as simple implementations as possible.

Regarding internal validity, the extent to which statements can be made about the causal effects of treatments on outcome, a wide variety of NNs are developed and deployed using different technology stacks and devices, mitigating that the results of our analysis of the overall performance of the applications are caused accidentally rather than by the variables of study themselves. However, we notice confounding factors that mostly affect the relation between the accuracy and the design decisions (i.e. quantity and quality of the data). Also, the error in the causal interpretation of our conclusions is tied to the estimation capabilities of our analysis models.

Finally, regarding external validity, our results are tied to the datasets and technology used for experimentation, and the devices used for operation, which determine the experimental settings to which the results obtained that regard the accuracy and complexity could be generalized.

ACKNOWLEDGMENTS

This work has been partially supported by the DOGO4ML Spanish research project (ref. PID2020-117191RB-I00) and by the the "Beatriz Galindo" Spanish Program BEAGAL18/00064.

[3]https://www.tensorflow.org/tensorboard/tensorboard_profiling_keras?hl=en
[4]https://docs.unity3d.com/Manual/Profiler.html

REFERENCES

[1] Abhijeet Banerjee and Abhik Roychoudhury. 2016. Automated re-factoring of android apps to enhance energy-efficiency. In *Proceedings of the International Conference on Mobile Software Engineering and Systems*. 139–150.

[2] Lingfeng Bao, David Lo, Xin Xia, Xinyu Wang, and Cong Tian. 2016. How Android App Developers Manage Power Consumption?-An Empirical Study by Mining Power Management Commits. In *2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)*. IEEE, 37–48.

[3] Victor R Basili, Gianluigi Caldiera, and Dieter H Rombach. 1994. The Goal Question Metric Approach. Encyclopedia of Software Engineering 1 (1994).

[4] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).

[5] Chansup Byun, William Arcand, David Bestor, Bill Bergeron, Matthew Hubbell, Jeremy Kepner, Andrew McCabe, Peter Michaleas, Julie Mullen, David O'Gwynn, et al. 2012. Driving big data with big compute. In *2012 IEEE Conference on High Performance Extreme Computing*. IEEE, 1–6.

[6] Roger Creus Castanyer, Silverio Martínez-Fernández, and Xavier Franch. 2021. Integration of Convolutional Neural Networks in Mobile Applications. arXiv:2103.07286 [cs.LG]

[7] Zhenpeng Chen, Yanbin Cao, Yuanqiang Liu, Haoyu Wang, Tao Xie, and Xuanzhe Liu. 2020. A comprehensive study on challenges in deploying deep learning based software. In *Proceedings of the 28th ACM ESEC/FSE*. 750–762.

[8] Shaiful Chowdhury, Stephanie Borle, Stephen Romansky, and Abram Hindle. 2019. Greenscaler: training software energy models with automatic test generation. *Empirical Software Engineering* 24, 4 (2019), 1649–1692.

[9] Shaiful Chowdhury, Silvia Di Nardo, Abram Hindle, and Zhen Ming Jack Jiang. 2018. An exploratory study on assessing the energy impact of logging on android applications. *Empirical Software Engineering* 23, 3 (2018), 1422–1456.

[10] Luis Corral, Anton B. Georgiev, Andrea Janes, and Stefan Kofler. 2015. Energy-Aware Performance Evaluation of Android Custom Kernels. In *2015 IEEE/ACM 4th International Workshop on Green and Sustainable Software*. 1–7. https://doi.org/10.1109/GREENS.2015.8

[11] Luis Cruz and Rui Abreu. 2018. Using automatic refactoring to improve energy efficiency of android apps. *arXiv preprint arXiv:1803.05889* (2018).

[12] Luis Cruz and Rui Abreu. 2019. Catalog of energy patterns for mobile applications. *Empirical Software Engineering* 24, 4 (2019), 2209–2235.

[13] Luis Cruz and Rui Abreu. 2019. EMaaS: Energy Measurements as a Service for Mobile Applications. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*. 101–104. https://doi.org/10.1109/ICSE-NIER.2019.00034

[14] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine* 29, 6 (2012), 141–142.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[16] Dario Di Nucci, Fabio Palomba, Antonio Prota, Annibale Panichella, Andy Zaidman, and Andrea De Lucia. 2017. Software-based energy profiling of Android apps: Simple, efficient and reliable?. In *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 103–114. https://doi.org/10.1109/SANER.2017.7884613

[17] Kevin Dowd and Charles Severance. 2010. High performance computing. (2010).

[18] Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford* 1, 12 (2009), 2009.

[19] Qianyu Guo, Sen Chen, Xiaofei Xie, Lei Ma, Qiang Hu, Hongtao Liu, et al. 2019. An empirical study towards characterizing deep learning development and deployment across different frameworks and platforms. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 810–822.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[21] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and H Adam Mobilenets. 2017. Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).

[22] Andreas Jedlitschka, Marcus Ciolkowski, and Dietmar Pfahl. 2008. Reporting experiments in software engineering. In *Guide to advanced empirical software engineering*. Springer, 201–228.

[23] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

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

[25] T L_etc Lai, Herbert Robbins, and C Zi Wei. 1979. Strong consistency of least squares estimates in multiple regression II. *Journal of multivariate analysis* 9, 3 (1979), 343–361.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436–444.

[27] L. E. Lwakatare, I. Crnkovic, and J. Bosch. 2020. DevOps for AI – Challenges in Development of AI-enabled Applications. In *2020 SoftCOM*. 1–6. https://doi.org/10.23919/SoftCOM50211.2020.9238323

[28] Lucy Ellen Lwakatare, Aiswarya Raj, Ivica Crnkovic, Jan Bosch, and Helena Holmström Olsson. 2020. Large-scale machine learning systems in real-world industrial settings: A review of challenges and solutions. *Information and Software Technology* 127 (2020), 106368.

[29] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, 142–150. http://www.aclweb.org/anthology/P11-1015

[30] Hui Mao, Ming Cheung, and James She. 2017. Deepart: Learning joint representations of visual arts. In *Proceedings of the 25th ACM international conference on Multimedia*. 1183–1191.

[31] Silverio Martínez-Fernández, Justus Bogner, Xavier Franch, Marc Oriol, Julien Siebert, Adam Trendowicz, Anna Maria Vollmer, and Stefan Wagner. 2021. Software Engineering for AI-Based Systems: A Survey. *arXiv preprint arXiv:2105.01984* (2021).

[32] Jeremy Miles. 2014. R squared, adjusted R squared. *Wiley StatsRef: Statistics Reference Online* (2014).

[33] Matthew B Miles and A Michael Huberman. 1994. *Qualitative data analysis: An expanded sourcebook*. sage.

[34] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. *Foundations of machine learning*. MIT press.

[35] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 188–197.

[36] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499* (2016).

[37] Lena Pons and Ipek Ozkaya. 2019. Priority Quality Attributes for Engineering AI-enabled Systems. *arXiv preprint arXiv:1911.02912* (2019).

[38] PT Pope and JT Webster. 1972. The use of an F-statistic in stepwise regression procedures. *Technometrics* 14, 2 (1972), 327–340.

[39] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3505–3506.

[40] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[41] Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2019. Green ai. *arXiv preprint arXiv:1907.10597* (2019).

[42] Julien Siebert, Lisa Joeckel, Jens Heidrich, Adam Trendowicz, Koji Nakamichi, Kyoko Ohashi, Isao Namba, Rieko Yamamoto, and Mikio Aoyama. 2021. Construction of a quality model for machine learning systems. *Software Quality Journal* (2021). https://doi.org/10.1007/s11219-021-09557-y

[43] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *nature* 550, 7676 (2017), 354–359.

[44] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. 2012. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks* 32 (2012), 323–332.

[45] Student. 1908. The probable error of a mean. *Biometrika* (1908), 1–25.

[46] Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*. PMLR, 6105–6114.

[47] Charles C. Tappert, Ching Y. Suen, and Toru Wakahara. 1990. The state of the art in online handwriting recognition. *IEEE Transactions on pattern analysis and machine intelligence* 12, 8 (1990), 787–808.

[48] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *arXiv preprint arXiv:1905.12616* (2019).