Open access • Journal Article • DOI:10.1109/JETCAS.2018.2839347

Optimized Hierarchical Cascaded Processing — Source link

Koen Goetschalckx, Bert Moons, Steven Lauwereins, Martin Andraud ...+1 more authors

Institutions: Katholieke Universiteit Leuven

Published on: 21 May 2018 - IEEE Journal on Emerging and Selected Topics in Circuits and Systems (Springer, Cham)

Topics: Energy consumption and Deep learning

Optimized Hierarchical Cascaded Processing

Koen Goetschalckx[*], Bert Moons[*]
Steven Lauwereins, Martin Andraud, Marian Verhelst
Department of Electrical Engineering, ESAT/MICAS,
KU Leuven, Leuven, Belgium[†]

October 17, 2019

Abstract

Recently, there has been an increasing demand for advanced classification capabilities embedded on wearable battery constrained devices, such as smartphones or -watches. Achieving such functionality with a tight power and energy budget has proven a real challenge, specifically for large-scale Neural Network based applications. Previously, cascaded systems have been proposed to minimize energy consumption for such applications, either through using a single wake-up stage, or by using a linear- or tree based cascade of consecutive classifiers that allow early termination. In this work, we expand upon these concepts by generalizing cascades to hierarchical cascaded processing, where a hierarchy of increasingly complex classifiers, each designed and trained for a specific subtask is used. This hierarchical approach significantly outperforms the wake-up based approach by up to 2 orders of magnitude in energy consumption at iso-accuracy, specifically in systems with sparse input data such as speech recognition and visual object detection. This paper presents a general design framework for such systems and illustrates how to optimize them towards minimum energy consumption. The text further proposes a roofline model for cascaded systems, derives system level trade-offs and proves the approaches validity through a visual classification case-study.

1 Introduction

There is an increasing demand for advanced classification capabilities to be embedded on battery constrained devices, ranging from wearables such as smartphones and -watches to ubiquitous Internet-of-Things (IoT) sensor nodes [1] [2]. Target applications in this field vary from advanced on-device speech- and face recognition on mobile phones to advanced feature extraction on smart sensor

[*]Equally contributing authors are listed in alphabetical order.

[†]This research work was partly supported by the FWO projects nr. G0B4613, G093114, S003817N and by IWT 131361.

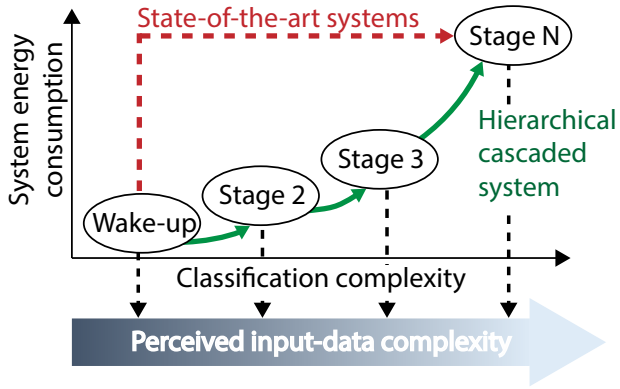


Figure 1: Illustration of a general hierarchical cascaded system

nodes, which reduce the necessary bandwidth to a central IoT node. Modern machine learning algorithms, such as Deep Neural Networks, offer the modeling capacity to perform this kind of functionality. However, they are typically too expensive in terms of energy consumption to be run on these wearable battery constrained devices.

This energy bottleneck can both be solved by developing more energy-efficient machine learning algorithms, as is for example done in the context of Deep Neural Networks [3] [4] [5] [6], and by developing more intelligent systems that can exploit application-level characteristics to increase energy efficiency. This paper focuses on the latter approach: increasing system-level energy efficiency through a hierarchical cascaded system that exploits input data statistics.

Some contemporary state-of-the-art (SotA) detection or classification systems reduce power consumption of always-on systems by adding a wake-up stage. This strategy is typically applied in video surveillance systems [7] and keyword recognition [8] or voice-activation [9, 10]. Here, a first stage performs a simple task with a low cost in stead of powering on the full more expensive functionality constantly. This reduces the average on-time of a more expensive classifier considerably and hence the global system's energy consumption. These systems typically do not expand upon this approach outside of a two-stage hierarchy, making them sub-optimal in many cases. Other alternatives try minimizing energy consumption and maximizing performance by building tree-based structures that perform multi-class recognition tasks [11] [12] [13] [14], or through building single-function cascades for binary [15] [16] [17] or multi-class problems [11]. However, none of these explicitly exploit input data characteristics. Furthermore, there exists no framework that can be used to optimize them towards minimum energy consumption at a given accuracy which is useful in the context of embedded processing on battery-constrained devices. Prior multi-class cascades can also be optimized using the presented approach and analysed using the presented roofline model in section 2.4, but are fundamentally different from the cascades discussed here. Throughout the cascade, they keep the number of

output classes constant and only refine decision boundaries in later stages, used to classify more difficult samples. In this work, decision boundaries are both added and refined throughout the different stages of the hierarchy, depending on the a-priori input probabilities of the input samples.

In this work, cascaded systems are generalized into a full-blown multi-level hierarchical cascade that outperforms the wake-up approach by up to two orders of magnitude at iso-accuracy, as it more adequately exploits data statistics. An introduction to such a system is given in Figure 1. Here, multiple hierarchical stages are cascaded and can potentially filter out data early. In this setup, both the complexity of the subtask and the cost of the subtask increase further down the hierarchical chain. Common classes such as 'silence', 'Alexa' or 'you' in the speech-recognition context are recognized early in the classification hierarchy with inexpensive classifiers, potentially preventing the final expensive stages from being powered on. Hence, even though later stages are more expensive, they are not used as much due to the discriminative functionality of the earlier stages. This proposed architecture combines the benefits of cascaded and tree-based topologies. As in the linear cascade approach, most negative samples are eliminated before the last stage(s), saving power in the overall system. As in a tree-based approach, multi-class problems are supported. Yet, early mis-detections can still be corrected in later stages. This work provides a framework used to minimizes overall energy consumption or computational cost in these hierarchical cascades, while simultaneously maximizing or maintaining system level accuracy.

In building such a framework, these contributions are key:

- We **generalize wake-up systems to hierarchical cascaded classifiers**: a sequence of increasingly complex classifiers building up to a final multi-class classification problem.
- We **propose a theoretical roofline model** to gain insight in the performance of a stage in a hierarchical system.
- We **derive general trade-offs** in a generic hierarchical model. Both the impact of input-data statistics, the number of stages used and the individual performance of stages in the hierarchy are discussed.
- We **validate** the hierarchical design approach in a visual recognition case study. This system can dynamically trade energy versus quality by changing the cascade's hyperparameters.

The full paper is organized as follows. Section II discusses the novel hierarchical approach proposed in this paper. Both the relevant terminology and the final optimization problem is discussed. Section III discusses the impact of several data- and system-level parameters on a generic hierarchal system. Section IV applies the proposed theory to a 100 face recognition application. Finally, section V concludes this work.

2 Hierarchical Cascaded Systems

This section introduces hierarchical cascaded systems, which are a multi-stage generalization of wake-up based systems.

2.1 Generalizing two-stage wake-up systems

In hierarchical cascaded systems, the overall computational cost of a classification system is minimized without sacrificing performance. This is done by building a functional hierarchy optimized for system-level energy efficiency: separate blocks with increasing functionality and cost are concatenated and jointly optimized. The hierarchy is more efficient than a single stage system, if the early stages are cheap, yet filter data adequately towards the more expensive later stages, without making too many mistakes that can not be recovered.

The overall system is divided in N stages, as depicted in Figure 2. The end task of the system is assumed to be a complex multi-class classification task. For instance, 100 faces in an image recognition application. The first task typically is a binary wake-up detector.

A typical hierarchy starts with a simple binary classification that removes the most obvious negative samples, such as background images or acoustic noise. If an input is classified as a positive sample, for example a meaningful image, the next stage is activated to make a more precise classification. Only samples that are detected as positives by the previous stage are fed to the following stage, as is the case in a classical cascade. Throughout this text, the positive class in this framework is called 'pass-on-class' (poc) and in Figure 2. However, unlike previous works, every next stage performs a more complex classification task. With this higher complexity and performance, its cost increases in a very non-linear way.

In order to build a hierarchical cascaded processing system, several alternatives at varying cost-accuracy trade-offs are built and trained for each stage of the hierarchy. Every individual stage is trained with a dataset modeling its specific subtask of the full-scale problem. From the system point of view, a stage is then abstracted by its performance (confusion matrix, ROC curve, etc.) and the whole abstracted system is then automatically optimized in order to achieve a given accuracy or recall at minimal complexity or cost. Details on this optimization problem are provided in section 2.5.

2.2 Hierarchical Cost, Precision and Recall

In order to automatically optimize hierarchical cascaded systems, descriptions of the system's total cost and recall are derived. The performance of each stage n in Figure 2 can be described using three separate entities. A stage n is defined by specifying its cost C_n and 2 per-class entities: recall $R_{n,i}$, pass-on-rate $POR_{n,i}$. Apart from these factors, the system's total cost $C_{1\rightarrow N}$ is also heavily influenced by the statistics of the input data, described by the a-priori probabilities of

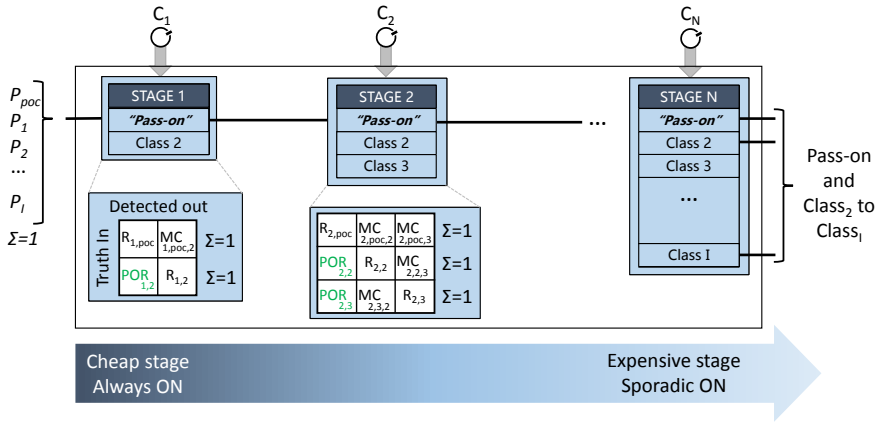


Figure 2: Basic hierarchical classifier system with N stages and I classes $\in [poc, 2, .., I]$. The a-priori chance of each class is P_i. Each stage n has an energy cost of C_i and is described through its pass-on-rate for every class POR_i, its recall R_i and its misclassification rate $MC_{i,j}$. The final per class recall is R_i as calculated in eq. 5.

occurrence P_i of each class (Figure 2). Based on all these terms, the overall system Recall and Cost can be derived.

Here, recall $R_{n,i}$ is the probability that an instance of a given class i is classified correctly in stage n,

$$R_{n,i} = P(\hat{y}_n = i | y = i) \tag{1}$$

which is according to its standard multi-class definition. The Pass-On-Rate $POR_{n,i}$ is the probability that an instance of class i gets passed along to the next classifier by stage n

$$POR_{n,i} = P(\hat{y}_n = poc | y = i) \tag{2}$$

Thus, $POR_{n,i}$ is the probability of an element of $class_i$ to be classified as the 'pass-on-class' poc in the given stage. The values of $R_{n,i}$ and $POR_{n,i}$ depend on the used classifier in the stage and from its operating point. These are both design choices and are hence part of the system optimization. If an input sample is misclassified as the pass-on-class in stage n, this can still be corrected in any of the $n+1:N$ stages. If a sample is misclassified MC as any other class in a given stage, the next stages are not powered on and the misclassification leads to a mistake that can not be corrected. The probability of misclassification of a sample of class i as class j for a given stage $MC_{n,i,j}$ is hence defined as follows:

$$MC_{n,i,j} = P(\hat{y}_n = j \notin \{poc, i\} | y = i) \tag{3}$$

This is further illustrated in Figure 2, where for the 3-class second stage the confusion matrix is given as an example. Here, rows indicate the actual class

and columns indicate the predicted class. Values classified to the first column of the confusion matrix are passed on to the next classification stage. The confusion matrix further formally illustrates how misclassified inputs $MC_{n,i,j}$, the recall $R_{n,i}$ and the pass-on-rate $POR_{n,i}$ are defined for each class in this stage. Such a confusion matrix can be generalized to any number of classes I. Note that the rows of the matrix are normalized, such that the matrix contains the classification probabilities of the given true classes instead of absolute counts, simplifying the equations.

The global cost in a hierarchy is a function of several data- and system-level characteristics. First, it is a function of the a-priori probabilities of the occurrence of each class P_i. Second, it also depends on the designed system: the cost C_n of each stage and the pass-on-rates $POR_{n,i}$ of every class in each stage. The average cost per input is hence:

$$C_{1\to N} = C_1 + \sum_{i=1}^{I}(P_i \times \sum_{n=2}^{N}(C_n \times \prod_{\eta=1}^{n-1} POR_{\eta,i})) \tag{4}$$

where $\prod_{\eta=1}^{n-1}(POR_{\eta,i})$ denotes the cumulative product of pass-on-rate for a class i up until stage $n-1$. In other words, the latter is the fraction of input samples of class i that are passed on to stage n. This factor multiplied with the probability of the class to be present as an input (P_i) and summed over all possible classes (I) is the relative number of times a stage is used per window. Multiplying this with the relative energy cost per stage (C_n) gives the average cost per sample.

The final recall of class i of the full cascade after N stages, $R_{1\to N,i}$, is independent of a-priori input statistics and can be described as:

$$R_{1\to N,i} = \sum_{n=1}^{N}\prod_{\eta=1}^{n-1}(POR_{\eta,i}) \times R_{n,i} \tag{5}$$

where $R_{n,i}$ is the recall rate for class i in stage n. If a stage n can not classify samples as class i, i is regarded as a subclass of the pass-on-class p. Thus, $R_{n,i} = 0$ and $POR_{n,i} = POR_{n,poc}$. If recall for every output is considered to be equally important, the average total recall after the first N stages is described as:

$$R_{1\to N} = \text{avg}_i(R_{1\to N,i}) \tag{6}$$

These formulas indicate high recall can either be achieved by high pass-on-rates or by high recall rates. If a sample is passed-on to the next stage, it can still be correctly classified by a later stage. However, higher pass-on-rates will also lead to a higher total cost, as this causes more activations of the later stages in the hierarchy.

When the system's average recall from equation 6 is optimized, the system's precision is also automatically maximized for every class. This is apparent from equations (7) and (8), which link the amount of true positives tp_i and misclassifications $mc_{i,j}$ to total recall $R_{1\to N}$ and precision $PR_{1\to N}$. Maximizing

total recall and total precision for any general system both require minimizing all final false positives or misclassifications.

$$R_{1\to N} = \sum_{i=1}^{I} \frac{tp_i}{tp_i + \sum_{j\neq i}^{I} mc_{i,j}} \tag{7}$$

$$PR_{1\to N} = \sum_{i=1}^{I} \frac{tp_i}{tp_i + \sum_{j\neq i}^{I} mc_{j,i}} \tag{8}$$

Here, $mc_{i,j}$ is the number of misclassifications, where a sample of class i is misclassified as class $j \neq i$. All of this information can be easily derived from the system's multi-class confusion matrix. This observation is useful, as no formula similar to equation (5) can be derived for final precision. The above shows that by optimizing for recall in any multi-class system, a high precision is automatically implied.

2.3 Processing Implications

Using a hierarchical cascade will have an impact on the *memory* requirements of the deployed system, as well as on the average processing *throughput* and the worst-case *latency* of the application. Instead of storing models only for the final functional stage in a hierarchy, one model is necessary for each stage. This increases the system's memory-requirements, but not necessarily significantly given that the total memory size is dominated by the size of the final, most complex classifier. In the example given in section 4, a 4 stage cascade achieving 85% total recall on face recognition, requires storing only 20% more weights than the single stage solution. Average throughput improves in an inversely proportional way with the system level number of operations, which is proportional to typically system costs such as energy or network complexity. Finally, the cascading approach impacts the worst-case latency, which has consequences for real-time applications. In this case, the worst-case latency is the summation of the latency of all N stages. The same 4 stage cascade example, has an increased worst-case latency of 0.2% compared to the latency of the final functional stage only, but at a 10000× lower average cost. Training times are also impacted by moving to cascaded systems, as the proposed optimization approach discussed in this paper requires training at least N classifiers. We argue the average cost benefits of a cascade, further quantified in section 4, outweighs these disadvantages.

2.4 A roofline model for Hierarchical Classifiers

In order to gain insight in the theoretical maximum performance of a classifier in a cascade, we propose a theoretical roofline model. An interesting upper-bound relationship between pass-on-rates and recall can be derived for any classifier in any stage of a hierarchical system. Based on this, we propose a roofline model for hierarchical cascades. Depending on the cost-budget of the classifier in a stage, the relationship between pass-on-rate and recall will be

closer or further from the theoretical roofline optimum. In a classifier with a pass-on-class, recall can be described for every class i and for a given stage n as $R_{n,i} = 1 - POR_{n,i} - \sum_{j \neq i}^{I} MC_{n,i,j}$. This formula can be rephrased and be used to plot a relationship between the average recall across all classes except the pass-on-class *poc*, $\text{avg}_{i \neq poc}(R_{n,i})$, against the average pass-on-rate across all classes, $\text{avg}_i(POR_{n,i})$. It is hence clear that:

$$\begin{aligned} &\text{avg}_{i \neq poc}(R_{n,i}) = \\ &1 - \frac{I}{I-1}\text{avg}_i(POR_{n,i}) + \frac{POR_{n,poc}}{I-1} - \sum_{n,i \neq p}^{I} \sum_{j=1}^{I} \frac{MC_{n,i,j}}{I-1} \end{aligned} \tag{9}$$

In the optimal case, there are no misclassifications. Samples are either passed-on or are classified correctly. In this case eq. (9) can be translated into a roofline, as given in eq. (10) and indicated in Figure 3.

$$\begin{aligned} &\text{avg}_{i \neq poc}(R_{n,i}) = 1 - \frac{I}{I-1}\text{avg}_i(POR_{n,i}) + \frac{1}{I-1} \\ &s.t.\ \text{avg}_i(POR_{n,i}) \geq 1/I \end{aligned} \tag{10}$$

Examples of ideal roofline and real curves are given in Figure 3. These real curves are taken from the face recognition hierarchy of section 4. The ideal roofline curves are according to eq. 10, while the non-ideal curves are real observed curves, that can be modeled by eq. 9. When the average POR in stage n, $\text{avg}_i(POR_{n,i})$, is below $1/I$, the behavior is non-ideal, as in this operating regime, samples of the pass-on-class are misclassified. The curve reaches a cutting point at $1/I$. Here, all samples are classified correctly (they have an average recall $R_{n,i}$ of 1) and none of them are passed-on (they have a pass-on-rate equal to 0) except for samples of the pass-on-class. At an $\text{avg}_i(POR_{n,i})$ larger than $1/I$, the optimal average recall $R_{n,i \neq poc}$ drops, in favor of more passed on samples. This does not lead to catastrophic failures, as any of the next stages can still lead to a correct classification. In the ideal roofline case, the cutting point is only a function of the number of possible classes in a given stage.

In Figure 3, the roofline curves, are compared to real performance curves taken from the face classification case study in section 4. They are for (a) a binary two-stage wake up classifier, (b) a 3-class stage and (c) a 10 class stage. Notice the cutting points at $1/I$ in each roofline. Every full line on these graphs is a classifier with a different complexity and cost. None of these real classifiers is identical to the roofline model, as real classifiers introduce misclassifications. Yet, it is clear that the roofline model offers a very good upper bound, and it is a close match for the most expensive classifier options. Naturally, the higher the complexity and modeling capacity of the classifier, the closer it comes to the optimal roofline model. The deviation of the roofline is mainly due to undesirable misclassifications, which lead to real errors that cannot be compensated for in later stages in the hierarchy.

Every classifier will hence have an associated *operating curve* in the recall vs pass-on-rate space. Its final *operating point* can be chosen to be any point on

Figure 3: Roofline model and real classifiers for the 2-, 3 and 10-class classification problems of stages WU, 2 and 3 in section 4. The respective cost of every stage in terms of necessary number of operations is given in the legend. More complex tasks require more operations to achieve a performance close to the optimal roofline.

this curve, by choosing a pass-on-threshold τ. This threshold is a discriminative threshold, similar to the one used in generating binary Receiver Operating Characteric (ROC) curves. It is the minimum confidence required for a sample to be classified as the pass-on-class *poc*. If this threshold is zero, all input samples are passed on and the $\text{avg}_i(POR_{n,i})$ will be 1. If the threshold is close to one almost no samples will be passed on in favor of more potential mis- or correct classifications. This threshold value hence determines the operating point of the classifier and can be chosen by the designer, or it can be automatically optimized. In the ideal roofline model, there are no misclassifications. Hence, the threshold directly trades recall for pass-on-rate.

Note that one bad model in the hierarchy does not necessarily destroy the system level recall, as the stage can operate in a point where it has a high pass-on rate. However, a system with high pass-on-rates does lead to a higher total cost, as more inputs are passed on to the more expensive later stages in the hierarchy. This leads to many degrees of freedom for every stage in the hierarchy, enabling to optimize cost for a given target recall, or vice versa. Optimally choosing all these characteristics per stage is a complex optimization problem that is solved throughout the rest of this paper.

2.5 Optimized Hierarchical Cascaded Sensing

Knowing that a maximal recall automatically maximizes precision and based on the previous discussion of general hierarchical sensing systems, a final two-objective optimization problem is defined as follows:

$$\begin{aligned} \min_{\boldsymbol{\tau},\boldsymbol{C}} \quad & C_{1\to N}(\boldsymbol{\tau},\boldsymbol{C},\boldsymbol{P}) \\ \max_{\boldsymbol{\tau}} \quad & R_{1\to N}(\boldsymbol{C},\boldsymbol{\tau}) \end{aligned} \tag{11}$$

Here $\boldsymbol{C}$, $\boldsymbol{P}$ and $\boldsymbol{\tau}$ are vectors representing stage level costs, a priori class probabilities and discriminating thresholds. $R_{1\to N}(\boldsymbol{C},\boldsymbol{\tau})$ and $POR_{n,i}(C_n,\tau_n)$ are all determined by the discriminative thresholds $\boldsymbol{\tau}$ and the performances of the classifiers (with cost C_n) that can also be chosen. The thresholds $\boldsymbol{\tau}$, $\boldsymbol{C}$ and the number of stages in the hierarchy N are hence the system's only optimization variables, P is known a-priori. The optimization variables, or hyperparameters, are summarized in table 1.

In section 3 this optimization problem is first solved for a general synthetic system, in search for general trends and the influence of all relevant parameters. Section 4 is a case study on hierarchical 100-class face recognition.

3 General proof of concept

In order to prove the proposed hierarchical cascaded processing concept, a generic framework is built. More specifically, for a general system, the impact of input-data statistics and system level specifications on the optimal hierarchy depth and architecture is investigated. This section hence optimizes problem (11), using estimated hierarchical POR-R curves in order to derive general trends and hierarchy design recommendations. In section 4, the theory is validated on a face recognition case study.

3.1 System description

The used generic system model mimics a cascade of hierarchical classifiers building up to a full 256-class classification system. The cascade is composed of N configurable stages. In each stage, an optimizer will choose an optimal classifier and its operating point. More specifically, in every stage, multiple classifiers are modeled with a POR-R trade-off curves such as in the roofline model. The exact operating point is actually determined by the discriminating

Table 1: Overview of optimization parameters

Parameter	Comments
C_n	Every cost is associated with a specific classifier in a stage
τ_n	Determines operating point R and POR for a given classifier
N	Number of total stages in the hierarchical cascade

threshold. The roofline itself is only a theoretical optimum and will hence have an infinite energy cost. In this example, additional non-optimal curves are added that have a lower energy cost, similar to the real curves observed in Figure 3.

The test-setup is a hierarchy that can contain a maximal of $N = 8$ stages, where every stage n classifies 2^n classes, one of which is a pass-on class. The question to be answered is how many stages the optimal hierarchy contains, and the optimal POR-R trade-off setting for each individual stage. For this setup, 256 different combination of stages are possible in the hierarchy, being all possible architectures with 1 to 8 stages. The final `END` stage is fixed as the 256-class classification stage, but the quality of the required classifier in this stage is also flexible.

In order to build a reasonable test case, several assumptions are made. First, costs increase exponentially from stage 1 to 8 according to eq (12) as the complexity of the classification tasks also increases exponentially.

$$C_n = 10^{log2(I)-1} \tag{12}$$

Second, within a single stage, a classifier has an exponentially higher cost if its performance is closer to the theoretical roofline optimum.

This R-POR-C design space is analytically modeled in this example, in order to find a suitable optimum by a steepest descent optimizer. The modeled classifier-to-classifier relative costs are illustrated for stage 1 and $N = 8$ in Figure 4 in the $\text{avg}_{i \neq poc}(R_{n,i \neq poc})/\text{avg}_i(POR_{n,i})$ space, which plot the R-POR trade-off curves for classifiers of different cost. The costs given here take the stage-to-stage cost from eq. 12 into account. Figure 4a shows the $\text{avg}_{i \neq poc}(R_{n,i \neq poc})$ vs $\text{avg}_i(POR_{n,i})$, while Figure 4b shows $POR_{poc} = R_{poc}$ vs $\text{avg}_i(POR_{n,i})$ for a binary classifier as a function of cost. The same is shown in Figure 4c and Figure 4d for the 256-class final stage. Notice none of these stages are ideal: the final average recall is below 1, even for classifiers with a very high cost.

Throughout the rest of this section, three recall-performance cases are discussed. More specifically the section looks into a high-, medium- and low-recall case, with system level recalls of 95% (high recall), 85% (medium recall) and 75% (low recall) relative to the theoretical maximum recall of the final `END` stage.

3.2 Input statistics

To estimate the impact of input data statistics, 4 different cases are looked at. A first case is where all input classes are **uniformly** distributed, meaning that they are equally likely to occur. Other cases are either with **medium**, **highly** or **extremely** skewed input statistics. The non-normalized Probability density functions (PDF) for these different cases are illustrated in Figure 5. Here, some specific classes 'C' occur much more frequently than others. An example of such a class 'C' can be 'noise', 'silence' or a common word in speech recognition, or 'background' or 'owner' in image recognition.

(a) Stage 1

(b) Stage 1

(c) Final stage

(d) Final stage

Figure 4: (a)(b) plot $\text{avg}_{i \neq poc}(R_{1,i \neq u})$ and $R_{1,u}$ versus $\text{avg}_i(POR_{1,i})$ for the first 2-class stage at different costs. (c)(d) show the same for the 256-class final stage. The legend indicates relative classifier costs.

Figure 5: Non-normalized Probability-Density-Functions (PDF) for 4 different cases. A uniform case where all classes are equally likely to appear and a medium, highly and extremely skewed distribution where some classes are more likely to appear than others.

3.3 Experiments

This subsection investigates the influence of recall targets and input statistics on the design of an optimal, minimum energy hierarchy. First, optimal cascades are designed for each of these cases. Second, the section provides insights in the design of the individual cascading blocks.

3.3.1 Optimal number of stages

Depending on target recall and input statistics, the optimal number of stages in a cascaded hierarchy will change. In order to investigate this, all possible combinations of hierarchies in the $N = 8$ setup for a 256-class classification problem are optimized for different system targets. More specifically, for every combination of total recall target, input data statistics assumption, and number of stages in the hierarchy, a complete classification cascade is optimized using the steepest descent optimizer. This optimizer selects for every stage in the hierarchy the optimal R-POR-energy setting, towards minimum system energy. The resulting energy cost of all chain optimizations are shown in Figure 6. Here, the system-level energy cost is plotted in function of the selected number of stages in the hierarchy for different target recall and data statistics assumptions. This shows that for systems with uniformly distributed input data, shallow hierarchies are optimal. If the input distribution is skewed more, cost can be reduced by going to deeper hierarchies, filtering out more samples early on. Costs can vary up to 6 orders of magnitude depending on input statistics and 2 orders of magnitude depending on the target recall. The optimal hierarchy architecture varies from 3 to 8 stages. This is further summarized in Table 2, which shows the selected stages in the optimal architecture with minimal cost. For example, in a system with uniform input distribution and a high recall target, only the last four stages are used, even though they are the most expensive. This is because because most samples can only get a correct final classification at the later stages. Thus, early stages with a limited subset of the final output classes are primarily overhead in the case of uniform inputs. However, for skewed input distributions, early stages classify most samples early at a lower costs. This gain outbalances the overhead for the few samples that need to be passed on.

Table 2 also shows the optimal cascade architecture if the hierarchy is constrained to only two stages: a first (wake-up) stage and the final stage. The same trends appear here: systems with high skew and low recall targets can use cheaper wake-up stages, with less output classes. A deep hierarchical cascade is up to 3 orders of magnitude more efficient than a two-stage (wake-up) architecture with the same system level performance, as is also illustrated in Figure 6.

3.3.2 Optimal stage metrics in a hierarchy

To gain insight in how to optimize each individual stage in the hierarchy, the settings of all optimally chosen classifier stages in a specific 6-stage medium recall - medium skew hierarchy are plotted in Figure 7. The figure shows the

(a) Uniform

(b) Medium Skew

(c) High Skew

(d) Extreme Skew

Figure 6: Optimal number of stages in the hierarchy as a function of the relative target recall. The input statistics are varied from Fig. (a) to (d) according to the distributions given in Fig. 5.

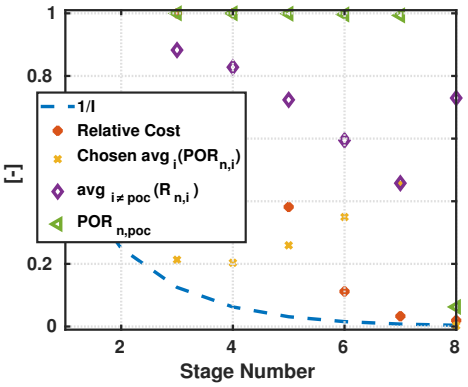


Figure 7: Relevant design metrics in an optimal 6-stage cascade with medium recall target and high skew.

Table 2: (x) Stages used in an optimal hierarchy for the 256-class problem. (o) Optimal stage choice if only a wake-up + end stage are allowed.

Rec	Distr.	S=1	2	3	4	5	6	7	END
		I=2	4	8	16	32	65	128	256
High	Uni.					x	x	x(o)	x(o)
	Med.			x	x	x	x(o)	x	x(o)
	High	x	x(o)	x	x	x	x	x	x(o)
	Ex.	x(o)	x	x	x	x			x(o)
Med	Uni.					x	x(o)	x	x(o)
	Med.			x	x	x(o)	x	x	x(o)
	High	x	x(o)	x	x	x	x	x	x(o)
	Ex.	x(o)	x	x	x	x	x	x	x(o)
Low	Uni.					x	x(o)	x	x(o)
	Med.			x	x(o)	x	x	x	x(o)
	High	x	x(o)	x	x	x	x	x	x(o)
	Ex.	x(o)	x	x	x	x	x	x	x(o)

relative cost (1 being the maximum) and $\text{avg}_i(POR_{n,i})$ of the stages, together with their $\text{avg}_{i \neq poc}(R_{n,i \neq poc})$ (1 being the theoretical optimum) and $POR_{n,poc}$ with these settings, as well as the roofline cutoff point $1/I$. Stage 1 and 2 are unused in this hierarchy as indicated in Table 2. All operating points are taken at an $\text{avg}_i(POR_{n,i})$ higher than the cut-off point $1/I$. This is required for a high pass-on-rate $POR_{n,poc}$, which is necessary to achieve a good final recall, as these pass-on-rates cumulatively multiply along the stages (equation 5). Only in the final stage 8, which doesn't have the possibility to let later stages deal with samples, the system sacrifices $POR_{n,poc}$ for a higher recall $\text{avg}_i(R_{n,i})$. Other optimal hierarchies show similar characteristics across the whole search space.

3.4 Conclusion

A general hierarchical system classifying 256 output classes was investigated using the developed roofline models and hierarchy optimization framework. This section discussed the influence of the performance goals and of skewed data statistics. With uniformly distributed input data, 4 stages are optimal in our generic model. For increasingly skewed distributions, such as speech or image data, a minimum-cost system will be deeper. This is because easily recognized classes such as 'noise' or 'background' in that context, can be dismissed early in the hierarchy at a low cost using a simple classifier. All of the tested settings using this model require a hierarchy of more than 2-stages. The more conventional wake-up system is hence never optimal.

4 Case study: hierarchical, CNN-based face recognition

4.1 A face recognition Hierarchy

To illustrate the power of the developed methodology in a real system, we apply it on an actual hierarchical face recognition system using Convolutional Neural Networks (CNN). In the most naive approach, such a system would scan small windows on different scales of a larger input image. Neural Network based large scale face recognition is very costly (1-2 mJ / 250x250 window) [18], especially in high resolution images that require a lot of windows to be processed. As input data is generally statistically skewed, it makes sense to build a processing hierarchy to exploit this and reduce the per mean sample cost of the overall system. To illustrate this, regard the pyramid-scale/sliding-window approach on 30fps Full-HD images (1920x1080 pixels) with a window size of 256, a stride of 4 and a scale factor of 2. In 30fps real-time this approach requires more than 3M window evaluations per second (100k per frame), almost all of which should be classified as backgrounds. At 1mJ/window evaluation, this consumes 3kW's of power, which is obviously infeasible on a wearable, battery constrained device. As distinguishing faces from backgrounds is a much simpler task that can be performed at ≈1uJ / subsampled 32x32 window [19], such a detector can be used as a wake-up stage for the more complex and costly face recognizer. Only if the face-detector detects a face, the more costly subsequent face recognizers are used. If not, the system goes on to the next window, reducing the average cost per window considerably depending on the input data's statistics. While it is clear that a hierarchical system can bring significant benefits for this face recognition system, it is not clear how many stages should be used in the hierarchy. If there is still statistical skewness between different face classes, e.g. the owner of a device appearing much more frequently than other faces, intermediate stages could be added to exploit this skewness in an effort to further decrease the activation rate of the costly final stage. On top of that, the R-POR-energy settings of each stage should be tuned optimally for minimal system energy consumption. To study this, Figure 8 illustrates the generalization of this wake-up approach to a hierarchical N-stage system that can ultimately distinguish between 100 faces. A window is passed on to the next stage of the hierarchy, only if it is classified as the 'pass-on-class'. In all other cases, i.e. when the image is classified as a '*background*', the '*owner*' or another specific '$face_i$' the window is considered to be classified. Again here, as the tasks in stage $n <$ `END` are cheaper than the ultimate task `END` and the distribution is skewed, cost can be reduced significantly. In this case-study, the general framework developed in section 2 is used to optimize the full hierarchy towards minimal average cost per sample, while maintaining overall face recognition accuracy. This analysis proves expanding the typical two stage wake-up system to a multi-stage hierarchical cascade reduces cost considerably. Optimal operating points for every stage are also derived.

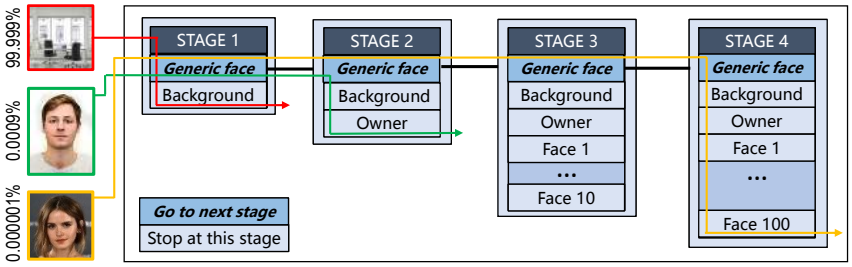


Figure 8: Illustration of the tested hierarchical face recognition example. Backgrounds and owners are much more likely to appear than specific faces, but also easier to detect with cheaper classifiers.

In order to analyze this specific case, the 100 face recognition system is split into a maximum of N=4 stages (face detection `WU` - owner detection `S2` - 10 faces `S3` - 100 faces `END`), where the last stage `END` is the full 100 face recognizer. For each of those stages, 15 CNN-based parametrized models are trained with varying complexity: the subsampled dimensions of the input image, the number of layers per network (the network depth) and the number of filters per layer (the network width). Larger networks operating on larger input images, generally achieve higher recall and precision on the input data set, albeit at a higher cost which is regarded as computing complexity in this case. The general architecture and parameters of these networks are given in Table 3, similar to the parameters used in [6]. In these networks, the input dimensions, the total number of layers (total depth) and the width of those layers are varied. The smallest network takes a subsampled 32×32 RGB input and classifies it using a 4-layer CNN with 4 filters per layer. The largest network takes a 128×128 RGB input and classifies it using a 7-layer network with up to 1024 filters per layer. All networks are trained on a 100-face subset of the VGG FACE-2 data set [20]. We use batch normalization and random data augmentation (shear, channel shifts, width shifts, height shifts, zoom and horizontal flips) to prevent overfitting.

Table 3: Parametrized network topology used for all N stages in hierarchical face recognition. All intermediate activations are LeakyRelu, the dense-layer uses a softmax activation function.

Block	(W,H) Dim.	Kernel	Stride	# Layers	Width
In. Layer	32 - 64 -128	3 - 5 - 7	1 - 2 - 4	1	4-256
Block A	32	3	1	1	4-256
MaxPool	32	2	1	1	-
Block B	16	3	1	1-2	4-512
MaxPool	16	2	1	1	-
Block C	8	3	1	1-3	4-1024
MaxPool	8	2	1	1	-
Dense	$16 \times width_C$	-	-	1	-

4.2 Hierarchical Cost, Precision and Recall

Each individual trained Convolutional Neural Network model has a specific recall and pass-on-rate for each of its output classes, together with an associated cost. All these terms are defined as in section 2.

According to the framework discussed in 2, we minimize full system cost, eq. (4), while maximizing total system recall as in eq. (6). As explained in subsection 2.2, precision will be automatically optimized as well.

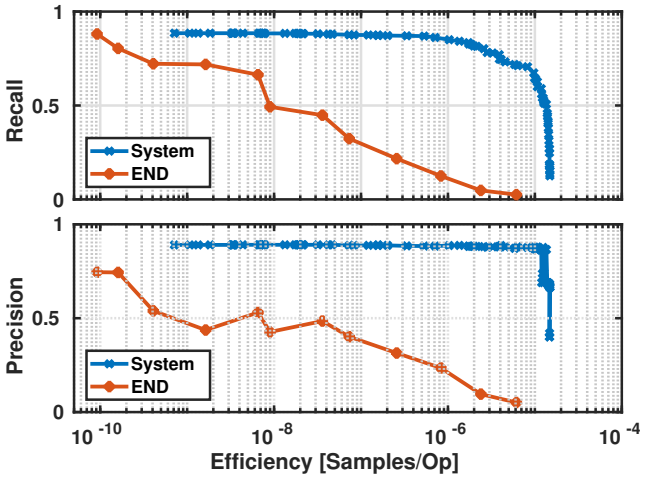
As shown by eq. (5), the full recall per class R_i depends on recall in every stage $R_{n,i}$ and the cumulative product of $POR_{n,i}$ in the previous stages. Note that theoretically the values of neither $POR_{n,i}$ and $R_{n,i}$ can be taken independently of the previous stages, as they influence the images that are offered to subsequent stages. Difficult images will be passed on more often than easy images, as they will be easily classified by one of the cheap first stages. However, in this optimization we assume the POR and R of every stage to be independent of the previous stages in order to have an analytical closed formula. We then later experimentally verify if the results under these assumption are correct, which is true in this test-case. Hence, the shown values for $POR_{n,i}$ and $R_{n,i}$ in this section are based on a representative test-set in the full hierarchy and not only based on the performance of the individual stages.

The performance of some individual classifiers in the different stages can be visualized using the proposed Roofline curves, as discussed in section 2.4 and illustrated in Figure 3. Here, the performance of the classifying stage, embedded in the hierarchy, is given in the *POR*-vs-*Recall* space. It can e.g. be seen that, for the 3rd stage, the best classifier (blue) requires $69k\times$ more operations than the cheapest depicted classifier, but at a performance that is much closer to the ideal roofline.

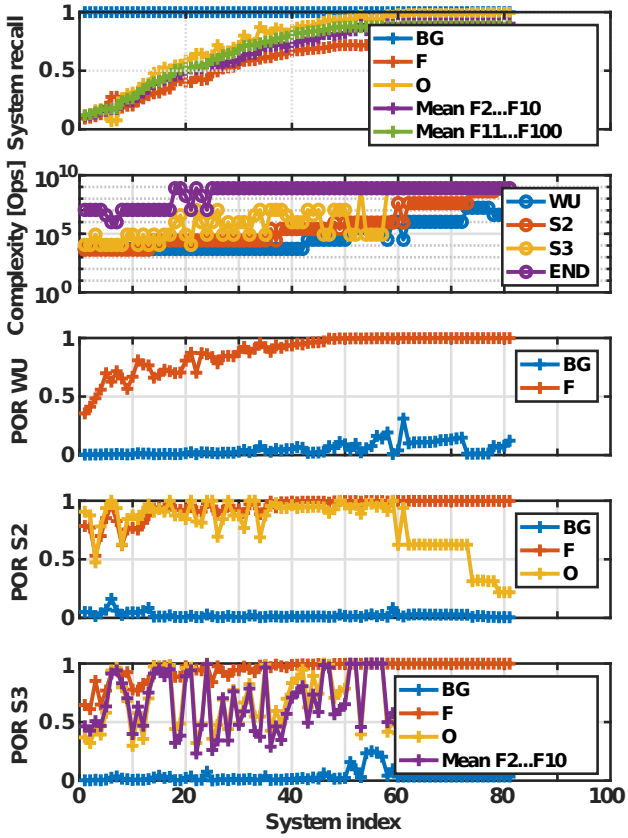
In order to estimate total cost, realistic a-priori input statistics have to be derived for all classes. Here, it is assumed that the system under investigation is used in the ESAT/MICAS - KU Leuven office in a surveillance context. In this office, on average 100 persons pass through the corridor every hour. Each of those people is in view of the system for 5 seconds. The a priori probabilities for each sub-group in this scenario are taken as [1 14.000.000 100 10 1]/(14.000.112) = ['pass-on-class' (F), background' (BG), 'owner' (O), 'face2-10', 'face11-100']. This is representative if the system investigates FHD images at 30fps with window-sizes of 256, a stride of 16 and a pyramid-scale factor of 2. When processing these windows, they are down sampled to one of the supported window sizes given in table 3: either 32×32, 64×64 or 128×128 RGB.

4.3 An optimized face recognition hierarchy

Once all relations between the overall cost $C_{1\rightarrow N}$, pass-on-rates $POR_{n,i}$ and global recall R_i are found, the hierarchy can be optimized towards minimum cost (in terms of average amount of operations per sample) and maximum overall recall. The variables in this optimization problem are the chosen classifier and the 'pass-on-class' detection thresholds at each stage, as these two factors



(a) Recall and Precision versus Efficiency.



(b) Complexity, POR and System Recall for every system in the optimization.

Figure 9: A comparison of hierarchical 100-face recognition using different stages in the Recall vs Efficiency [Frames/op] space. The upper right corner is optimal.

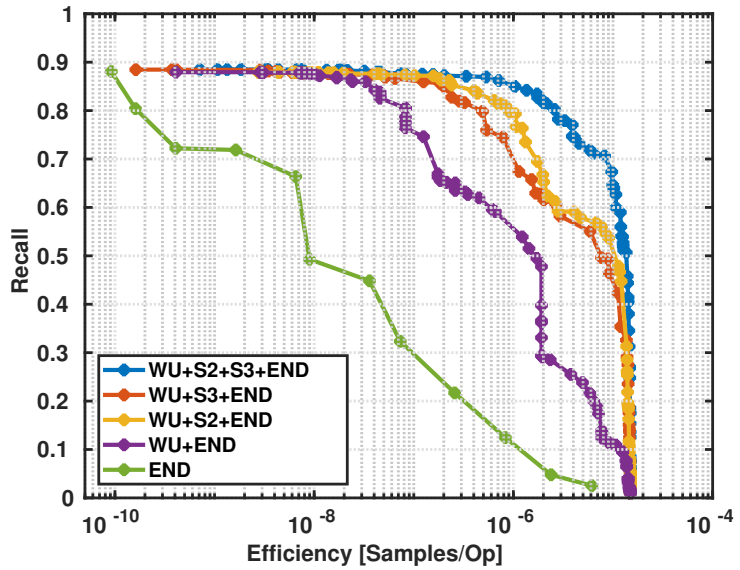
determine $R_{n,i}$ and $POR_{n,i}$. A particle swarm multi-objective optimizer is used to numerically solve this discrete optimization problem. Figure 9 shows the results of this optimization for a 4-stage face recognition pipeline.

Figure 9a shows all pareto-optimal possible systems in the efficiency-vs-recall and efficiency-vs-precision design space. Here, efficiency is described as [Sample/op] or the number of samples that can be classified per MAC-operation. For the same recall, the 4-stage system obviously requires orders of magnitude less operations than a system existing out of the `END` stage only. For example, its efficiency at an average recall of 80% is 10.000 times higher than in the single stage case. Figure 9a also experimentally verifies that optimizing for high average recall also automatically optimizes for high average precision, a claim made in section 2. If recall is high and close to its maximum, precision is also maximized. Precision and recall are high in the 4-stage system over a wide range of efficiencies, while they are only high at a very low efficiency in the 1-stage system.

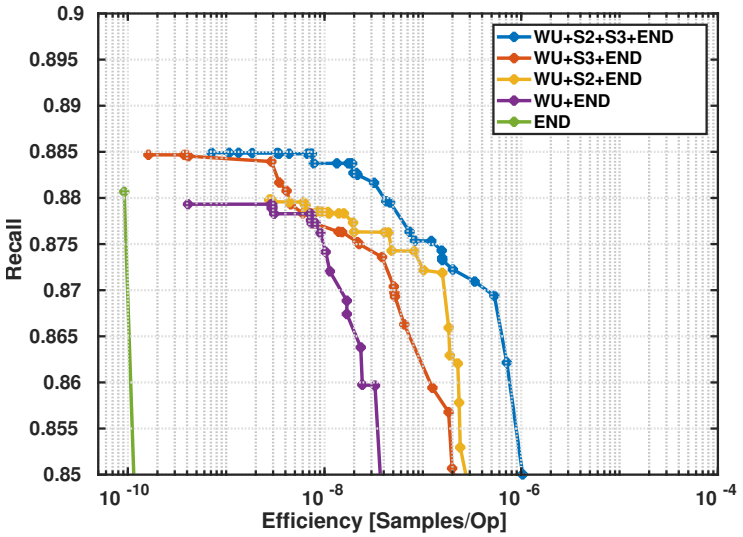
The characteristics for every pareto optimal architecture in Fiure 9a are illustrated in Figure 9b. Figure 9b shows the optimal chosen classifiers, the pass-on-rate for every class at every stage ($POR_{n,i}$) and the final recall per class, averaged per subgroup with equal a-priori probability for readability. The figure clearly indicates that systems with higher recall also require more expensive building blocks. Even at high recall, the complexity of the wake-up stage is 3 orders of magnitude lower than the complexity of the `END` stage, which explains most of the gains of the multi-stage system. All used classifiers operate near the roofline.

We expand this analysis by performing the same optimization on shallower hierarchies, from 1 stage, to the maximum of 4 stages. Figure 10 illustrates the performance of these different hierarchical architectures in the efficiency-vs-recall space for the full 100 face recognition functionality. An architecture is optimal if it achieves the highest recall at a given efficiency in terms of processed samples per operation (Samples/op). Obviously, using only the `END` stage is very inefficient. Also the more typical 2-stage hierarchy with a wake-up stag proves to be sub-optimal for this test-case, requiring 1-to-2 order of magnitude more operations at similar recall than the optimal 4-stage case. The performance gains achieved by going to deeper hierarchies with more than 4 stages are arguably minimal, especially if the maximum latency is taken into account. Figure 10b shows a zoomed-in version of the same graph, illustrating how the number of stages becomes less important close to the maximum achievable recall. This maximum recall is essentially limited by the best performance of the best possible classifier in the final stage.

Figure 9a and 10 also show the quality-vs-energy trade-off can be dynamically changed in a hierarchy, by changing the threshold that determine POR and Recall of every stage.



(a) Full Design Space



(b) Zoomed-in Design Space

Figure 10: A comparison of hierarchical 100-face recognition using different stages in the Recall vs Efficiency [Samples/op] space. The upper right corner is optimal.

5 Conclusion

This work generalizes wake-up systems to multi-stage hierarchical cascaded systems that have a lower system level cost and are better adapted to skewed data with non-uniform probability distributions. This work offers 4 key contributions. (1) We generalize wake-up systems to multi-stage hierarchical cascades. (2) We propose a design framework that can be used to simultaneously optimize performance and minimize costs in these systems and introduce a theoretical roofline model used to gain insight in the performance of the individual stages in the hierarchy. (3) We derive trends in the design of a hierarchical cascade through analyzing a general example. It is shown that, while hierarchical cascades do not bring significant benefits for uniform input data statistics, systems with skewed input data statistics, such as speech and object detection tasks, benefit from deeper cascades. If an intermediate stage is used, its optimal operating point is close to the theoretical roofline. Stages with bad recall and pass-on-rate performance are never beneficial. (4) We prove the approach works by designing a 4-stage 100 face-recognition application. An optimal operating point exists, where 4 orders of magnitude in cost-efficiency can be gained compared to the single-stage classifier and 2 orders of magnitude compared to the traditional 2-stage wake-up based system. The proposed framework and roofline model are generally applicable on many sensory data applications. We hope this framework can contribute in moving this field from ad-hoc designs, towards automated system-optimizations.

6 Acknowledgment

This work has been supported by the FWO SBO project OmniDrone under agreement S003817N, and the EU ERC project Re-SENSE under agreement ERC-2016-STG-715037.

References

[1] A. van Dam, "Beyond wimp," *IEEE Computer Graphics and Applications*, vol. 20, no. 1, pp. 50–51, Jan 2000.

[2] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *Comm. Mag.*, vol. 48, no. 9, pp. 140–150, Sep. 2010.

[3] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[4] L. Liu and J. Deng, "Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution," *arXiv preprint arXiv:1701.00299*, 2017.

[5] G. Huang, D. Che, T. Li, F. Wu, L. van der Maaten, and K. Weinberger, "Multi-scale dense networks for resource efficient image classification," *International Conference on Learning Representations (ICLR)*, 2018.

[6] B. Moons, K. Goetschalckx, N. Van Berckelaer, and M. Verhelst, "Minimum energy quantized neural networks," *Asilomar Conference on Signals, Systems and Computers*, 2017.

[7] J. Yuan, H. Y. Chan, S. W. Fung, and B. Liu, "An activity-triggered 95.3 db dr −75.6 db thd cmos imaging sensor with digital calibration," *IEEE journal of solid-state circuits*, vol. 44, no. 10, pp. 2834–2843, 2009.

[8] M. Sun, D. Snyder, Y. Gao, V. Nagaraja, M. Rodehorst, N. S. Panchapagesan, S. Matsoukas, and S. Vitaladevuni, "Compressed time delay neural network for small-footprint keyword spotting," *Proc. Interspeech 2017*, pp. 3607–3611, 2017.

[9] K. Badami, S. Lauwereins, W. Meert, and M. Verhelst, "Context-aware hierarchical information-sensing in a 6μw 90nm cmos voice activity detector," in *International Solid-State Circuits Conference (ISSCC)*, 2015, pp. 1–3.

[10] M. Price, J. Glass, and A. P. Chandrakasan, "A scalable speech recognizer with deep-neural-network acoustic models and voice-activated power gating," in *International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 244–245.

[11] S. Venkataramani, A. Raghunathan, J. Liu, and M. Shoaib, "Scalable-effort classifiers for energy-efficient machine learning," in *Proceedings of the 52Nd Annual Design Automation Conference*, ser. DAC '15. ACM, 2015, pp. 67:1–67:6.

[12] M. Li, W. Bijker, and A. Stein, "Use of binary partition tree and energy minimization for object-based classification of urban land cover," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 102, pp. 48–61, 2015.

[13] Z. E. Xu, M. J. Kusner, K. Q. Weinberger, M. Chen, and O. Chapelle, "Classifier cascades and trees for minimizing feature evaluation cost." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2113–2144, 2014.

[14] Z. Ghahramani, M. I. Jordan, and R. P. Adams, "Tree-structured stick breaking for hierarchical data," in *Advances in neural information processing systems*, 2010, pp. 19–27.

[15] P. Viola and M. Jones, "rapid object detection using a boosted cascade of simple features," in *Proc. of Conference on Computer Vision and Pattern Recognition*, 2001, pp. 511–518.

[16] M. Saberian and N. Vasconcelos, "Boosting algorithms for detector cascade learning," *Journal of Machine Learning Research*, vol. 15, pp. 2569–2605, 2014.

[17] P. Panda, A. Sengupta, and K. Roy, "Conditional deep learning for energy-efficient and enhanced pattern recognition," *arXiv preprint arXiv:1509.08971*, 2015.

[18] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "Envision: A 0.26-to-10tops/w subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm fdsoi," in *International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 246–247.

[19] D. Bankman, L. Yang, B. Moons, M. Verhelst, and B. Murmann, "An always-on 3.8 μj/86% cifar-10 mixed-signal binary cnn processor with all memory on chip in 28nm cmos," in *International Solid-State Circuits Conference (ISSCC)*, 2018, pp. 222–224.

[20] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," *arXiv preprint arXiv:1710.08092*, 2017.