Contention-aware Adaptive Model Selection for Machine Vision in Embedded Systems

Basar Kutukcu*, Sabur Baidya*, Anand Raghunathan†, Sujit Dey*
*Department of Electrical and Computer Engineering, University of California, San Diego
†Electrical and Computer Engineering, Purdue University
* Email: {bktkc, sbaidya, dey}@ucsd.edu; † Email: raghunathan@purdue.edu

***Abstract*—Real-time machine vision applications running on resource-constrained embedded systems face challenges for maintaining performance. An especially challenging scenario arises when multiple applications execute at the same time, creating contention for the computational resources of the system. This contention results in increase in inference delay of the machine vision applications which can be unacceptable for time-critical tasks. To address this challenge, we propose an adaptive model selection framework to mitigate the impact of system contention and prevent unexpected increase in inference delay by trading off the application accuracy minimally. The framework uses a set of hierarchical deep learning models for image classification. It predicts the inference delays of each model and selects the optimal model for each frame considering the system contention. Compared to a fixed individual model with similar accuracy, our framework improves the performance by significantly reducing the inference delay violations against a practical threshold. We implement our framework on Nvidia Jetson TX2 and show that our approach achieves a gain over the individual model by 27.6% reductions in delay violations.**

I. Introduction

Modern machine vision systems involve complex algorithms based on artificial intelligence that need significant computing resources to satisfy the performance demands. Especially, the real-time applications of machine vision systems impose stringent constraints on the end-to-end latency. However, this is very challenging for applications, e.g. autonomous systems where the computing is performed in a resource-constrained system. Moreover, the computing system running the machine vision application can share resources with other computing loads, e.g., the connected and autonomous vehicles can process camera data together with intermittently activated radar data on the same computing unit for better fused perception in crowded areas or in the presence of obstacles. In such a scenario, the machine vision on camera data can contend with the radar data processing for the computing resources, increasing the application latency. While increasing the priority of certain tasks can avoid the system level contention, it can cause a starvation for the other tasks running on the same system. Instead, an alternative approach is to handle contention within time limits of the tasks. In this work, we examine the effects of contention on an image classification system, and propose a framework which minimally compromises the accuracy of the image classification system to satisfy the latency requirements.

Since embedded systems have limited resources, running deep learning algorithms on them has been addressed by several previous efforts. Some works develop ground up efficient deep neural networks [1], [2]. Some other works make existing deep neural networks more efficient by using quantization [3], [4]. However, even with lowering the computational complexity of the deep learning algorithms, the system needs to adapt to cope with the contending processes and satisfy the performance of the image classification application.

As many modern systems may not allow accessing low level system information in real-time due to security concerns or complexity of the platform, herein, we propose an application-level data-driven predictive framework for contention-aware adaptive model selection. We implement the framework on Nvidia Jetson TX2 and show the advantage of predictive adaptation mechanism in dynamic contention scenarios.

II. System Overview

A. Machine Vision System

In this paper, we consider image classification application of the machine vision systems. There are two main metrics that define the performance of an image classification applications - inference delay and accuracy. An ideal real-time machine vision system aims to minimize the inference delay and maximize the accuracy of the image classification task.

Now, in a computing system, assigning more tasks than its capability creates contention on the specific system resources resulting in increase in delay for the completion of the tasks. The contention can be seen in different computing resources depending on the workload. For example, it can be on CPU which has a limit of issuing instructions in a certain time window, can be on memory which has a limited capacity, or can be on bus which has a limited bandwidth. These contentions and their impact are seen more frequently on the resource constrained systems such as embedded systems than servers or desktop computers.

B. Delay Accuracy Trade-off

There is a trade-off between inference delay and accuracy of an image classification system as more complex image classification algorithms result in higher accuracy but also require more time to compute those results for a given available system resources. Since contention creates dynamic variations in the available resources, a machine vision system needs to optimize the performance in terms of delay and accuracy. Typical autonomous systems need to satisfy a delay constraint while maximizing its image classification accuracy. In order to achieve that, machine vision systems can use a set of N hierarchical image classification models $\{M_i\}, i = 1(1)N$

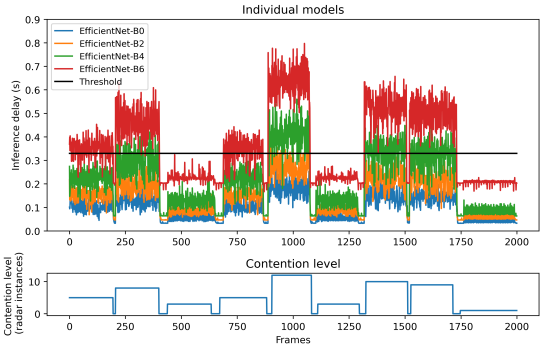

Figure 1. Inference delays of different models under changing contention

with increasing complexities. Then depending on the available resources in presence of contention, it can chose the optimal model. For example, inference delays of four different EfficientNet [5] models are shown in Figure 1. Each model's inference delay increases proportionally under increasing contention. Figure 1 shows that if we have an inference delay constraint, we can satisfy it by choosing appropriate model for appropriate contention level.

C. Model Selection Problem

However, for perfect selection of an appropriate model, one needs to have a priori knowledge about the accurate contention level for the next time slot and select the best model that fits in the available resources, which is nontrivial. However, one can estimate the forthcoming impact of contention on different image classification models ahead of time and select the best model for the next image frame.

Real-time model selection is previously investigated in literature for different scenarios. In [6], the authors measure the input image's complexity before classification and select the ideal model for the specific input image content. In [7], two models are employed, one big and one little. Each image is classified by the little model first. Then, if the classification is found unsuccessful, big model classifies the same image again. However, the unsuccessful attempts in this method increases the latency, and hence, not suitable for real-time machine vision system. Moreover, the aforementioned methods do not consider the contention into account for model selection.

In [8], the input and the contention are considered to select an approximate branch of a predefined model. The contention is determined by matching previous inference delays of approximate branches and a look-up table that consists benchmarks of each approximate branch. Our work is different in two aspects. First, instead of approximate branches, we use multiple models that are readily available in the RAM. Hence, changing models does not have any overhead. Second, our contention measurement is embedded in regression models instead of look-up tables. Also, our delay normalization mechanism allows us to use different models' inference delay to measure contention. This can be useful when contention and therefore model selection change rapidly. This work [8] also shares some concerns for using multiple models in limited memory. We address this by showing that loading multiple models does not incur the memory problem, and present more insights on this in the Performance Evaluation section.

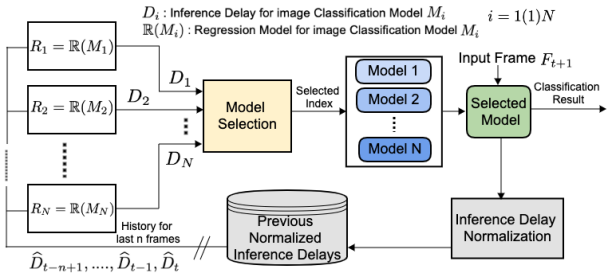

Figure 2. Overview of the proposed framework

Herein, we predict the future inference delay of the model set $\{M_i\}$ in presence of contention. Then, we find a subset of models $\{M_j | D_j < T\}$, $j = 1(1)L$, $L \leq N$, where D_j is the predicted inference delay of model M_j and T is a latency threshold of the system. After that we choose the appropriate model M_k such that the accuracy $A_k = \max\{A_j\}$.

III. Solution Framework

A. Prediction Framework

The overview of the proposed framework is shown in Figure 2. The framework employs a set of image classification models whose accuracy and inference delay increase incrementally. The framework chooses the optimal model for the next frame's classification while considering the current contention on the system. The optimal model is determined by using historical information and a set of linear regression models. The historical information comes from the previous frames' normalized inference delays. There is one regression model for each image classification model used in the framework. The regression models are trained before runtime using their corresponding image classification models on randomly changing contention level. All regression models take the same input, the previous normalized inference delays, and output the predicted inference delay for their corresponding image classification model. Then, the framework chooses the most appropriate model based on the delay threshold constraint and maximum accuracy as mentioned earlier.

Delay Normalization

Figure 3a shows the 2000 consecutive frames' inference delays for EfficientNet-B0, B2, B4 and B6 [5] under increasing contention. It shows that the inference delay values depend on two things - the system level contention, and the image classification model type. Since our framework uses historical inference delay values to represent the impact of contention, we remove the model type dependency by normalization. We use min-max normalization as following:

$$x_{normalized} = \frac{x - min(x)}{max(x) - min(x)} \quad (1)$$

The result of normalization is shown in Figure 3b. The minimum and maximum values for each model is saved before runtime and used to normalize the inference delays of the models during runtime.

Prediction and Selection

The training data is created by running each model under randomly changing contention levels. As input, the normalized data is split into chunks of *n* consecutive normalized inference delays. All of the regression models take the same input as

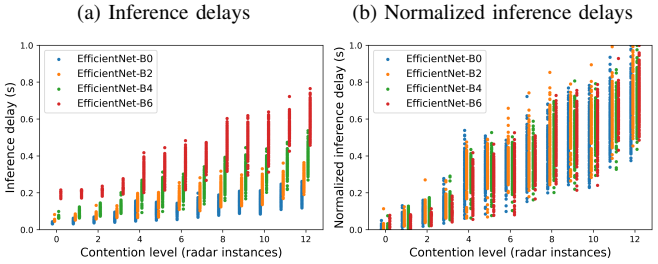

Figure 3. Inference delays under increasing contention and normalization

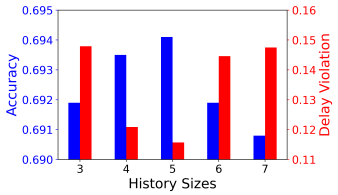

Figure 4. Effect of history sizes on accuracy and delay violations

they will be predicting in parallel using the same input. As prediction output, non-normalized delays are used. Each regression model has different output corresponding to its image classification model. Hence, each regression model takes same input, n previous normalized inference delays, and predicts its corresponding image classification model's inference time for the next frame. After this step, the framework has a predicted inference delay for each image classification model. The image classification models are already ranked in terms of accuracy on static datasets before runtime. Therefore, the framework chooses the model which has the highest rank and also a predicted inference delay under the predefined threshold.

B. Experimental setup

The experiments are performed on the Nvidia Jetson TX2 platform with the Tensorflow framework. We used EfficientNet [5] for image classification, as EfficientNet proposes multiple models with similar architectures but with scaled features. Therefore, accuracy and delay of the proposed models are incrementally ranked. For demonstration, we use 4 EfficientNet models - EfficientNet-B0, B2, B4, B6. Among these, B0 has the lowest accuracy and the smallest inference delay and B6 has the highest accuracy and the largest inference delay. We used pre-trained weights from the Tensorflow library and the ImageNet-v2 [9] dataset for testing. All of the reported accuracy values are from top-1 prediction labels.

We consider a scenario of modern vehicles, where radar processing can intermittently contend with the image classifications in different road situations. So, we created system contention from radar detection algorithms. Two different Poisson distributions are used to generate contention. One distribution $P1(\lambda_1)$ is used for the duration of the contention, other $P2(\lambda_2)$ is used to determine the contention level. Each contention period is followed by a random non-contention idle period. and we continue the cycle over 2000 frames.

We compare our predictive model selection with two reactive algorithms. The first one is called 1-step reactive which checks the last frame's inference and then, selects 1-step stronger model if the last frame's inference is below threshold. Otherwise, it selects the next (1-step) weaker model. The second reactive algorithm is called N-step reactive. This

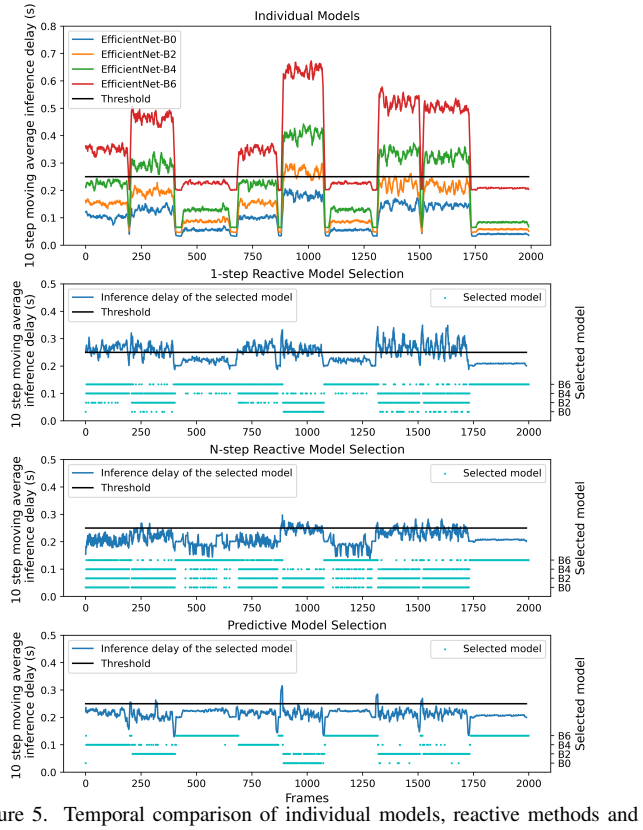

Figure 5. Temporal comparison of individual models, reactive methods and the predictive method

algorithm similarly checks the last frame's inference delay and selects 1-step stronger model if the last frame's inference is below the threshold. However, if the last frame's inference is above the threshold, it conservatively selects the weakest model for the next frame to satisfy the delay threshold.

C. Performance Evaluation

The regression models use previous n normalized inference delays, we experiment with different n values as shown in Figure 4. The optimal n value is found to be 5 since it results in a prediction framework with the highest accuracy and the smallest delay violation for image classification. A smaller n value results in a regression model which is too sensitive to noise. Similarly, a higher n value results in a regression model which is slow to react to contention changes.

The temporal plots for 2000 frames under a specific contention regime with λ_1=200 and λ_2=6 are shown in Figure 5. The inference delay and the selected model's index are given for three different model selection methods. The individual models' inference delays are also plotted for comparison purposes. The inference delays are averaged for 10 frames to smooth the plots. The delay plots for individual models show that using a single model under varying contention is not optimal. The individual plots also suggest the best model under a specific contention regime, e.g., around the frames $100, 250, 500, 1000$, the ideal models are B4, B2, B6, B0 respectively. It can be seen that the predictive method can successfully select the optimal model most of the time. It can also choose multiple models under the same contention region. For example, B0 is the ideal model around frame 1000.

Model	Accuracy (%)	Delay Violations (%)
EfficientNet-B0	59.90	0.40
EfficientNet-B2	63.80	11.25
EfficientNet-B4	69.55	39.55
EfficientNet-B6	72.15	58.45
Average-(B0-B2-B4-B6)	66.35	27.41
1-step Reactive Model Selection	69.50	30.20
N-step Reactive Model Selection	67.60	21.50
Predictive Model Selection	69.20	11.95

Table I
COMPARISON OF METHODS IN A SPECIFIC CONTENTION REGIME.

However, B2 is not completely above the limits. Therefore, B2 can also be chosen from time to time in this region since it will give a better accuracy.

Table I shows the summary of data for Figure 5 in terms of average performance of different schemes. If a frame classification takes more time than predefined threshold, we consider it as delay violation. It shows that all of the model selection methods have an accuracy close to B4, which is the second best individual model in terms of accuracy. However, the reactive methods have large delay violations as well. On the other hand, the predictive method has only 11.95% delay violation. We also experimented with the model selection methods in different contention regimes by varying λ_1 and λ_2. For our experiments, 5 different contention duration distributions λ_1=$\{40, 80, 120, 160, 200\}$ and 3 different contention level distributions with λ_2=$\{4, 6, 8\}$ are used. The unit of contention duration is frames and the unit of contention level is radar application instances. The mean accuracy and the mean delay violation results are reported in Figure 6. It shows that in every contention regime predictive method has much lower delay violations while maintaining similar or better accuracy.

An ideal model selection method would yield scatter values that forms a line with zero slope in Figure 6. Because that means the ideal method can foresee the future and select exactly the correct model for every contention level, and therefore it would not violate inference delay threshold by compromising accuracy. The Figure 6 shows that the predictive method has the smallest slope and the lowest delay violation values in each contention level compared to reactive methods.

We further tuned the predictive model selection by adjustment of variance of delays. The variance of the inference delays increase after contention level 4 as shown in Figure 3. This noise causes occasional errors in model selection. To solve this, we compute the variance of inference delays of each model in Figure 3a. Then, at runtime, if the mean normalized inference delays is the high-variance region, we add the pre-computed variances to the predictions of the regression models. The result of the variance corrected (VC) predictive model is also shown in Figure 6. Compared to the vanilla predictive model selection, the mean accuracy only reduces by 0.68% (from 0.691 to 0.686), while it improves the performance by reducing the mean delay violation by 11.85% (from 0.135 to 0.119).

The average time cost for one frame of our predictive framework is $0.29\,ms$ which is approximately 850 times smaller than average inference delay for one frame. Therefore,

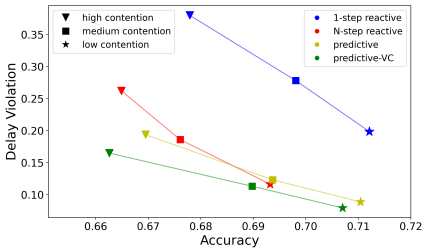

Figure 6. Delay violations vs avg. accuracy for different model selection methods under different contention levels

we can say that the time cost of the framework is insignificant. The memory cost of our framework is $3\,$GB when 4 models are used. However, $2.6\,$GB of this cost comes from Tensorflow and it is a one time cost as long as all of the models are loaded in the same process. For example, we loaded and used 25 EfficientNet-B0 instances with the cost of $4.1\,$GB which leaves a very large space for other processes or even more deep learning models in Jetson TX2 with $8\,$GB memory. Thus, RAM availability is not the bottleneck for increasing the number of models in this problem as indicated in [8]. In practice, the bottleneck is finding large number of clearly separated steps (with models or some other methods) in terms of both distinct accuracy and inference delay. Even after this bottleneck is solved, it can be rare to observe this large number of contention levels in practical applications on an embedded system to make using this large number of steps reasonable.

IV. CONCLUSION

In this paper, we proposed a contention-aware adaptive image classification model selection framework. The experimental results show that our predictive model selection outperforms the average of individual models in both accuracy and inference delay violation. Predictive model selection also outperforms the reactive model selection methods.

ACKNOWLEDGMENT

This work is partially supported by DARPA under grant number 304259-00001.

REFERENCES

[1] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size," *arXiv:1602.07360*, 2016.

[2] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in *Proceedings of the IEEE CVPR*, 2018.

[3] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv:1602.02830*, 2016.

[4] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained ternary quantization," in *ICLR*, 2017.

[5] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *ICML*, 2019.

[6] B. Taylor, V. S. Marco, W. Wolff, Y. Elkhatib, and Z. Wang, "Adaptive deep learning model selection on embedded systems," *ACM SIGPLAN Notices*, 2018.

[7] E. Park, D. Kim, S. Kim, Y. D. Kim, G. Kim, S. Yoon, and S. Yoo, "Big/little deep neural network for ultra low power inference," in *International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS*, 2015.

[8] R. Xu, R. Kumar, P. Wang, P. Bai, G. Meghanath, S. Chaterji, S. Mitra, and S. Bagchi, "ApproxNet: Content and Contention-Aware Video Analytics System for Embedded Clients," *arXiv:1909.02068*, 2020.

[9] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do imagenet classifiers generalize to imagenet?" in *ICML*, 2019, pp. 5389–5400.