Temporal Motionless Analysis of Video using CNN in MPSoC

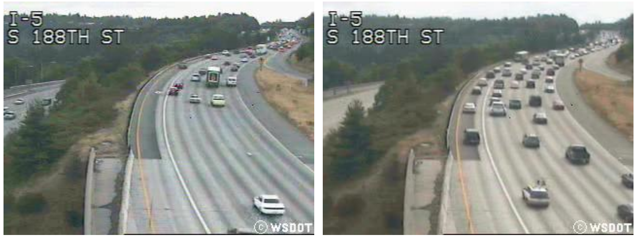
Somdip Dey*[1], Amit Kumar Singh*[2], Dilip Kumar Prasad†[3], Klaus McDonald-Maier*[4]
*School of Computer Science and Electronics Engineering, University of Essex, UK
Email: {[1]somdip.dey, [2]a.k.singh, [4]kdm }@essex.ac.uk
†Department of Computer Science, UiT The Arctic University of Norway, Tromsø, Norway
Email: [3]dilip.prasad@uit.no

***Abstract*—This paper proposes a novel human-inspired methodology called IRON-MAN (*Integrated RatiONal prediction and Motionless ANalysis* of videos) on mobile multi-processor systems-on-chips (MPSoCs). The methodology integrates analysis of the previous image frames of the video to represent the analysis of the current frame in order to perform Temporal Motionless Analysis of the Video (*TMAV*). This is the first work on *TMAV* using Convolutional Neural Network (CNN) for scene prediction in MPSoCs. Experimental results show that our methodology outperforms state-of-the-art. We also introduce a metric named, Energy Consumption per Training Image (*ECTI*) to assess the suitability of using a CNN model in mobile MPSoCs with a focus on energy consumption of the device.**

***Index Terms*—Covolutional neural network (CNN), temporal analysis, motionless analysis, video, energy efficiency, embedded device, multiprocessor systems-on-chip (MPSoCs)**

I. Introduction and Motivation

Recently there has been a huge increase in utilizing Convolutional Neural Networks (CNNs) [1], [2] to solve several real-life challenges such as traffic categorization [3], [4], weather forecasting [5], etc due to its high prediction accuracy/categorization in the aforementioned target applications. One particular case for harnessing the efficacy of visual CNN based prediction model is Intelligent Transportation Systems (ITS), which is becoming an important pillar in the modern "smart city" framework.



(a) Frame predicted as *Light* traffic category (b) Frame predicted as *Heavy* traffic category

Fig. 1. Frames (Images) from the same *Light* traffic category of UCSD dataset [6] and associated prediction by a trained CNN model [7]

Traffic load categorization is challenging given the increase in vehicles on road. Some of the popular ways of monitoring and categorizing traffic load from videos include vehicle based assessment method and holistic approach [7]. In vehicle based assessment methodologies, either vehicles are first localized on the road with a background subtraction method or the vehicles are localized with moving feature keypoints. Whereas, in holistic approach, a macroscopic analysis of traffic flow is understood through global representation of a scene, which is obtained by accounting for spatio-temporal features except tracking using background subtraction and moving feature keypoints.

In recent times, there has also been emergence of several methods capable of monitoring and analyzing traffic using motionless analysis of videos [3], [7], where videos of traffic are broken into frames instead, and the frames are analyzed for further computation or prediction. The main motivation to utilize methodologies consisting of motionless analysis of video is that it is difficult to stream high-frame rate videos gathered by a large network of interconnected cameras due to bandwidth limitation. Hence, streaming low-frame rate videos on these camera networks is very common. In many cases, it is challenging to stream more than 2 frames per second due to the limited bandwidth of the network when these cameras stream over a WIFI network [3], [7]. Moreover, to analyze video in real-time without motion features over a WIFI network is difficult due to communication bandwidth constraint and hence, it is better to analyze the image fames on the camera enabled embedded device itself [7] instead of relying on a server system over the WIFI network. Another motivation to devise such approaches in embedded devices is the affordability of such devices, e.g. in developing countries, instead of employing powerful server systems used for analysis purposes [7]. Therefore, the approach of analyzing videos without motion features on the embedded device is not just beneficial for categorizing traffic load but could be extended to several computer vision based real-world applications that require analysis of low-frame rate videos. Although motionless analysis of videos has its own benefits, it also come with limitations described with the following observation.

Although several implementations of such methods were able to achieve high prediction accuracy on known dataset [3], [7], in some test cases the prediction/analysis was not accurate at all. The reason for poor prediction/analysis is that in some cases it is difficult to predict the label of an image frame from a video if the ground truth[1] of the image is overlapping with several other categories (labels). For example, in the dataset of traffic released by UCSD [3], [6], which consists of light, heavy and traffic jam categories, two frames (images) belonging to the same category of video are predicted differently by the CNN model [7]. The reason for such behavior is that the CNN predicts the label and

This work is supported by the UK Engineering and Physical Sciences Research Council EPSRC [EP/R02572X/1 and EP/P017487/1].

[1] Ground truth of the image frame in this case is the information gained through empirical evidence as opposed to the inference made by the CNN model.

probability of it occurring on the instantaneous image frame. In Fig. 1, we notice that a trained CNN model [7] with an overall prediction accuracy of 81.25% predicted the wrong label for a frame, which falls under *Light* category, but was instead predicted as *Heavy*. However, both the frames belong to the same video under the *Light* category. If the ground truth of the two images (Fig. 1 (a) & (b)) are compared then it is justifiable that the prediction by the CNN is in fact accurate due to the fact that the traffic projected in Fig. 1.(b) is more congested than the traffic projected in 1.(a). In reality the analysis of each image frame of the video should also portray the overall analysis[2] for the video instead of the image frame itself in order to convey the temporal prediction. Since, each individual image frame of a video could lead to different analysis result (prediction/label), the temporal prediction is the prediction analysis of the video over time. This limitation is due to the fact that the trained CNN model only predicts the label or analyzes the current image frame without taking past image frames into consideration.

There have been some recent studies, which focused on future predictions of motion in ego-centric videos[3] [8], [9] or action[4] [10], [11] taken by a human being, such as predicting the future position of a person based on the current image frame. However, no study to our best knowledge has tried to predict the scene[5] [12], [13] of a video from image frames taking predictions from immediate previous frames into account to provide a more holistic analysis of the scene over a time period. Hence, we call such an analysis as ***T**emporal **M**otionless **A**nalysis of **V**ideo (TMAV)*. Several target applications of CNN such as traffic categorization require such kind of analysis in comparison with traditional ones [3], [14], [7].

In order to overcome the limitations of the existing approaches we propose **IRON-MAN**: **I**ntegrated **R**ati**ON**al prediction and **M**otionless **AN**alysis of videos using CNN, which is capable of performing *TMAV*, in Multi-Processor System-on-chips (MPSoCs). To this end, this paper makes the following contributions:

1) An energy efficient scene prediction methodology (*IRON-MAN*) based on CNN, which integrates predictions of previous image frames of a video to predict the current frame, and hence, analyze the video without using motion features.
2) A new metric named *Energy Consumption per Training Image* ($ECTI$), which will enable the choice of suitable CNN model for real-world applications on embedded devices keeping energy-efficiency in mind.
3) Validation of the proposed approach on a real hardware platform, the Odroid-XU4 [15].

II. Proposed Methodology: IRON-MAN

In our proposed approach, we utilize the concept of *Hybrid Training Method* [7], where we train our model both during offline (training period) and online (runtime/post-training period) modes. IRON-MAN (**I**ntegrated **R**ati**ON**al prediction and **M**otionless **AN**alysis of videos) has two modules in it: *Training* and *Prediction* (as shown in Fig. 2). The strength of our approach is that it provides temporal analysis of videos without motion features i.e. *TMAV*.

In the *training module*, we use *transfer learning*[6] [16], [17] by utilizing an existing pre-trained network and training the classifier with our data categories. First, we train the pre-trained CNN with our dataset, which could be either performed on the MPSoC or on a powerful computing system, which has a lot more computing resources than the MPSoC that could be leveraged to improve the training time. After the initial phase of training is complete, we evaluate the overall prediction accuracy of the trained CNN. If the overall prediction accuracy (P_i) of the CNN is equal or more than the desired quality of experience (Q) [7], then we utilize the CNN for prediction in the prediction module. However, if the desired prediction accuracy is not achieved then we *retrain* (details in Sec. II-A) the CNN with the failed predicted images. This *retrain* methodology is human-inspired as it mimics one of the key intelligence feature of a human being, which is learning from the surrounding environment to adapt. When a human being meets a new environment and is not aware of the rules and regulations associated with it, the human tries to adjust and adapt by learning the new set of rules and regulations. We have utilized the same concept in our approach as well, which is described subsequently in Sec. II-A.

After the *retraining* of the CNN, when the desired prediction accuracy is achieved we use the trained CNN in the *prediction* module. Now, instead of providing prediction result for each individual image frame, we integrate the final prediction by taking previous image frames into consideration. Our CNN model's prediction is inspired by *Bayes' theorem* and *sequential Bayesian updating* [18], where the model updates the probability of the occurring prediction label by incorporating the probability of the label occurring in the previous frames. This approach is again human-inspired as it is adopted from the ideology of humans updating their knowledge using Bayesian inference logic. Detailed inner working (*Training* and *Prediction* modules) of the IRON-MAN is provided in the following two subsections.

A. Training Module

For the proposed approach, any pre-trained CNN model such as VGG [2], ResNet [19], MobileNet [20], etc could be selected. The training module itself consist of two part training: Offline and Online mode. During the offline mode, the CNN is trained with stock images from a dataset stored on a memory. After the initial training period (offline) the CNN is then fed with live images from the camera and the prediction for each image frame is evaluated during the validation of the model. Upon failure in prediction for each image during validation testing, the image is stored in a stack implementation called "*reFeed Image Stack*". After utilizing cross-validation technique [21], where validation testing is performed on a separate dataset such as live image frames from the camera stream, the overall prediction accuracy (P_i) of the CNN model is evaluated. If the overall prediction accuracy

[2] Here, overall analysis of video means the analysis of the video as a whole as opposed to the analysis of each image frame of the video.

[3] Ego-centric video is a first-person vision technique, which acts as an artificial visual system that perceives the world around camera wearers and assist them to decide their next action.

[4] Action is based on the movement of the human being in consideration in the image frame.

[5] Scene is a place where a human being could navigate or can act within.

[6] Learning achieved by taking the convolutional base of a pre-trained network, running the new data of 4 traffic categories through it and training a new randomly initialized classifier

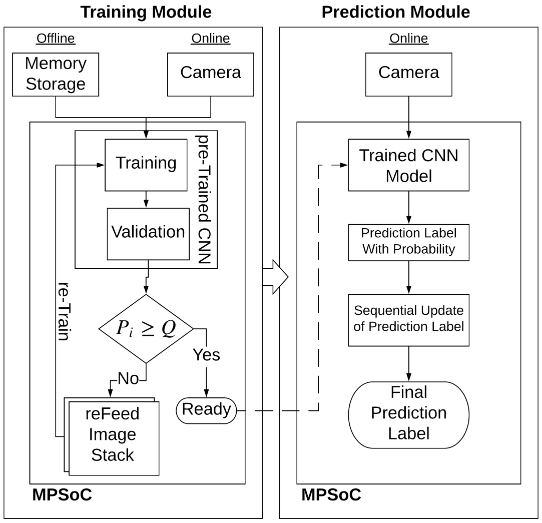


Fig. 2. IRON-MAN Model Work-flow

is equal or more than the desired quality of experience (Q) then the training process concludes and the CNN is ready to start predicting categories in the *prediction* module. The governing equation to check for the suitability of the CNN for further prediction is provided in Eq. 1[7], where I is the dataset consisting of images, i is an image in the dataset and P_i is the prediction accuracy of the CNN for i. However, if the desired quality of experience is not met in terms of prediction accuracy then the CNN is trained with the failed prediction images, which are stored in the *reFeed image stack*. We call this as the *reTrain* approach so that the CNN can achieve a higher *localized prediction accuracy*. Here, the term *localized prediction accuracy* means the prediction accuracy of the model for a set of images that is restricted to a specific task or place. In this application, it should be kept in mind that *reTrain* does not mean training the whole model from the beginning, however, it means to continue the training process with the images from the reFeed image stack in order to improve learning capability of the CNN model. The retrain mechanism is bound to improve prediction accuracy because we train the CNN model with the failed images[7] saved in the reFeed image stack and this approach mimics a human being's ability to rectify his/her mistake after making one.

$$\forall \{i \in I : i > 1\}, P_i \geq Q \tag{1}$$

B. Prediction Module

From Bayes' Theorem [18] we can represent the expression representing the *Bayesian Update Scheme* as follows:

$$posterior \propto prior \times likelihood \tag{2}$$

In Eq. 2, *posterior* is the revised probability of an event occurring after taking new information into consideration, *prior* is the probability of the event assessed before revising posterior and *likelihood* is the probability that an event which has already occurred would yield a specific outcome. Now, using sequential update scheme where we take the past into account and the modified expression for Bayesian Update Scheme is as follows:

$$new\ posterior \propto current \times new\ likelihood \tag{3}$$

In Eq. 3, *current* is the probability of some entity occurring whereas the *new likelihood* is the Bayesian Update taking posterior from the past into account. This approach sometimes is also called a Recursive Bayesian Update. For our scenario, we are trying to predict the current probability for the label (category), which becomes the *new posterior* in the equation, *current* is the probability prediction of the category of the image frame provided by the CNN and *new likelihood* is the probability of the category occurring in some previous time steps. Here, the reason to mention *some previous time steps* is because the number of previous time steps to take into consideration will be a heuristic choice of the user. In our case, we call the number of previous frames (images), which is considered to provide an integrated prediction for the chosen category, as *Frame Window*. *Frame Window* consists of N number of frames, which are taken into consideration.

If we consider that the prediction for the category in the current frame as $P^{category}_{this}$, prediction for the same category in the previous frame as $P^{category}_{this-1}$ and the total prediction accuracy of the model after the training/cross-validation of the model is complete as P_{CNN} then the updated equation for Bayes' Theorem is as follows:

$$P^{category}_{updated} \propto P^{category}_{this-1} \times P^{category}_{this} \tag{4}$$

Eq. 4 could be utilized to predict the frame using the prediction of previous frame as follows:

$$P^{category}_{updated} = \frac{P^{category}_{this-1} \times P^{category}_{this}}{(P^{category}_{this-1} \times P^{category}_{this}) + P_{CNN}} \tag{5}$$

In Eq. 5, $P^{category}_{updated}$ is the updated prediction using Bayes' Theorem for the same category by the CNN model. We should also note that both $P^{category}_{this-1}$ and $P^{category}_{this}$ are *conjugate priors* for our scenario since they belong to the same category as the posterior ($P^{category}_{updated}$) and hence, in the same probability distribution family. Now, depending on the *Frame Window*, the evaluation of $P^{category}_{updated}$ will vary, which leads us to an updated equation as follows:

$$P^{category}_{updated} = \begin{cases} \frac{P^{category}_{this-1} \times P^{category}_{this}}{(P^{category}_{this-1} \times P^{category}_{this}) + P_{CNN}}, & \text{if N = 1} \\ \prod_{1}^{N} \frac{P^{category}_{this-N} \times P^{category}_{this-(N-1)}}{(P^{category}_{this-N} \times P^{category}_{this-(N-1)}) + P_{CNN}}, & \text{if } N > 1 \end{cases} \tag{6}$$

Eq. 6 is the governing equation, which is utilized to predict the probability of the category during the *Frame Window*.

In the Prediction module, IRON-MAN has a queue implementation of the image stack (called as *Image Queue*), where the N number of frames are stored and N is defined by the user to denote the size of the *Frame Window*. When an $(N+1)^{th}$ image frame comes from the camera for prediction, the images stored at 1^{st} position of the *Image Queue* is popped out and the $(N+1)^{th}$ image frame is pushed in the

[7] Here, failed images are the image frames which were predicted/labeled incorrectly during the testing of the trained CNN model.

N^{th} position of the queue while everything getting shifted a place in the middle just like in a first-in-first-out (FIFO) queue implementation. When prediction for a particular frame is required, the prediction of the frame by the CNN model is provided as well as the prediction of the *Frame Window* is provided by using the Eq. 6. After utilizing Eq. 6 the updated prediction ($P_{updated}^{category_i}$) for a specific category i is compared with the the updated prediction of other categories and the label for the maximum value of the prediction/probability is provided as output.

C. ECTI: Energy Consumption per Training Image

A new metric, $ECTI$ (*Energy Consumption per Training Image*), is introduced to choose the suitability of a CNN model in embedded systems. If we consider ET as the total execution time period required to train the CNN with a dataset I consisting of n number of images to achieve a validation prediction accuracy of P, Q as the *quality of experience*, and the average power consumption per second during the training period as e then the equation for $ECTI$ could be defined as follows:

$$ECTI = (\frac{ET}{n} \times e) \text{ iff } P \geq Q \tag{7}$$

The unit of $ECTI$ is $kilo-watt-hour$ (kWh), where ET is represented in hours and e in *kilo-Watt* (kW). To choose the most suitable CNN for an embedded application we have to select the CNN with the least value of $ECTI$.

III. Experimental Results

Dataset used: We have performed our validation on *traffic categorization*. For our *traffic categorization* experimentation we are using the same dataset used in [3], [7].

Hardware setup: We have implemented the methodology on an Odroid-XU4 [15], which employs Exynos 5422 MPSoC [22] used in popular Samsung Note phones and phablets.

Experimental Results: To categorize traffic we chose four pre-Trained CNN models, which were trained on millions of ImageNet images, for our validation. These four CNN models are VGG16, VGG19 (a deeper network of VGG16), ResNet50 and MobileNet. In our experiments, we have chosen the $quality\ of\ experience$ (Q) to be 0.7 i.e. 70%. Through empirical evidence it was noticed that the CNN model performs best when the prediction accuracy of 70% or more is chosen in order to be utilized for traffic categorization application in the real world, hence, 0.7 was chosen to be the $quality\ of\ experience$. For VGG16, VGG19, ResNet50 and MobileNet it took us 360, 360, 330, 360 images respectively to train the pre-Trained using Transfer Learning and our proposed retrain approaches (see Sec. II-A), and gained a testing prediction accuracy of 98.93%, 96.62%, 92.79% and 85.75% respectively. The total execution time of the VGG16, VGG19, ResNet50 and MobileNet CNN model's training are 5864, 6093, 7047 and 5301 seconds respectively. The average power consumption of VGG16, VGG19, ResNet50 and MobileNet on the MPSoC during the training are 10.63, 10.67, 10.59 and 9.89 Watt respectively, and the average operating temperature on the MPSoC during training of VGG16, VGG19, ResNet50 and MobileNet are 93.6°, 94.01°, 93.6°and 93.3° centigrades respectively. Based on Eq. 7 the evaluated ECTI values for VGG16, VGG19, ResNet50 and MobileNet are 4.81×10^{-05}, 5.02×10^{-05}, 6.28×10^{-05} and 4.05×10^{-05} kWh (approx.) respectively, hence, proving MobileNet to be the most energy efficient model.

IV. Conclusion

In this paper, we propose IRON-MAN, which is capable of providing *Temporal Motionless Analysis of Videos* ($TMAV$) i.e. analyzing videos without motion features and providing a holistic temporal analysis while utilizing predictions of the past image frames into consideration. Based on the results we have shown that MobileNet is more energy efficient compared to VGG and ResNet50 models. Therefore, it is recommended to perform the training off the embedded device for improved energy efficiency or utilize MobileNet for on-device traffic categorization.

References

[1] X.-W. Chen and X. Lin, "Big data deep learning: challenges and perspectives," *IEEE access*, vol. 2, pp. 514–525, 2014.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[3] Z. Luo *et al.*, "Traffic analysis without motion features," in *2015 IEE ICIP*. IEEE, 2015, pp. 3290–3294.

[4] G. Kalliatakis *et al.*, "Detection of human rights violations in images: Can convolutional neural networks help?" *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP*, 2017.

[5] G. Zhang *et al.*, "Forecasting with artificial neural networks:: The state of the art," *International journal of forecasting*, vol. 14, no. 1, pp. 35–62, 1998.

[6] A. B. Chan and N. Vasconcelos, "Classification and retrieval of traffic video using auto-regressive stochastic processes," in *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*. IEEE, 2005, pp. 771–776.

[7] S. Dey *et al.*, "Mat-cnn-sopc: Motionless analysis of traffic using convolutional neural networks on system-on-a-programmable-chip," *NASA/ESA AHS*, 2018.

[8] T. Yagi *et al.*, "Future person localization in first-person videos," *arXiv preprint arXiv:1711.11217*, 2017.

[9] G.-Y. Lai *et al.*, "People trajectory forecasting and collision avoidance in first-person viewpoint," in *2018 IEEE ICCE-TW*. IEEE, 2018, pp. 1–2.

[10] R. De Geest, E. Gavves, A. Ghodrati, Z. Li, C. Snoek, and T. Tuytelaars, "Online action detection," in *European Conference on Computer Vision*. Springer, 2016, pp. 269–284.

[11] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

[12] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Advances in neural information processing systems*, 2014, pp. 487–495.

[13] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. IEEE, 2010, pp. 3485–3492.

[14] Z. Luo, P.-M. Jodoin, S.-Z. Su, S.-Z. Li, and H. Larochelle, "Traffic analytics with low-frame-rate videos," *IEEE TCSVT*, vol. 28, no. 4, pp. 878–891, 2018.

[15] "Odroid-xu4," https://goo.gl/KmHZRG, accessed: 2018-07-23.

[16] S. J. Pan *et al.*, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[17] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[18] A. Doucet *et al.*, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and computing*, vol. 10, no. 3, pp. 197–208, 2000.

[19] K. He *et al.*, "Deep residual learning for image recognition," in *IEEE CVPR*, 2016, pp. 770–778.

[20] A. G. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[21] S. Arlot, A. Celisse *et al.*, "A survey of cross-validation procedures for model selection," *Statistics surveys*, vol. 4, pp. 40–79, 2010.

[22] "Exynos 5 octa (5422)," https://www.samsung.com/exynos, accessed: 2018-07-23.