# Cross-model temporal cooperation via saliency maps for efficient frame classification

Tomaso Trinci[1,2], Tommaso Bianconcini[1], Leonardo Sarti[1], Leonardo Taccari[1], and Francesco Sambo[1]

[1]Verizon Connect [2]University of Florence

## Abstract

*Minimizing the energy consumption of deep learning models is becoming essential due to the increasing pervasiveness of connected and mobile devices. Real-time video frame classification is a perfect example of energy-intensive task that could present battery consumption and overheating issues on embedded devices. In this paper we propose a novel architecture to tackle this problem efficiently, exploiting temporal redundancies between consecutive frames. The model consists of two convolutional neural network streams with different parameter sizes and input resolutions. Each frame is processed by only one of the streams, and the stream with the lowest input resolution and parameter size uses saliency maps generated by the other stream on a previous frame. The energy consumption can be manually controlled by choosing a proper schedule of the two streams. We show the effectiveness of our proposed architecture in a task that involves recognizing the state of the relevant traffic lights in images from on-board cameras.*

## 1. Introduction

Nowadays, the inference phase of deep networks in resource-constrained devices represents a major challenge in a lot of applications. The research has focused on different approaches to achieve a good trade-off between energy consumption and model quality [1, 4]. Real-time video processing on embedded devices is an example of application that benefits from these studies, as it involves processing a continuous stream of images, with a computational cost that grows linearly with the frame rate of the video.

In this paper, we propose a novel neural network architecture to address the problem of energy-efficient real-time video frame classification, which aims at classifying images captured by a video camera in an efficient manner. Our primary objective is to reduce the energy cost during the inference phase. To achieve this goal, we draw inspiration from the concept of *dynamic networks*, which adapt their structure or parameters based on the input during inference [6]. In particular, we focus on temporal-wise dynamic adaptation, where neighboring frames' correlation is leveraged to reduce computational load. One common solution is to share features from the past so they can be re-used with minimal cost for adaptation for augment the current representation [9]. Another solution is to squeeze the inference phase when low or no semantic variation from previous frames is detected [14, 12, 5].

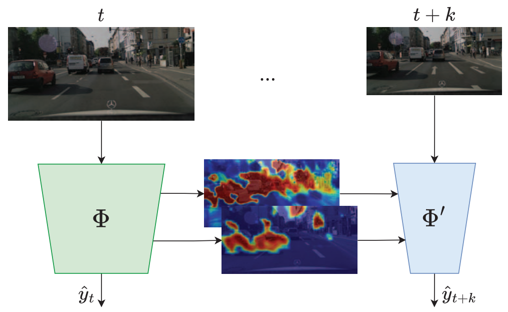

Figure 1. A wider model $\Phi$ with higher-resolution input is used at time $t$ to predict $\hat{y}_t$ and to produce saliency maps that are used to improve the quality of the prediction of the smaller $\Phi'$ at $t+k$.

We propose a model consisting of two convolutional neural network streams that differ in terms of the number of parameters and input resolution (Fig. 1). Each video frame is processed by only one of the two streams following a predefined schedule, and the smaller stream is trained to make predictions using not only the current frame, but also *saliency maps* [16] generated by the wider stream on a previous frame. This idea of utilizing saliency maps between video frames has already been explored in [13, 8] mainly with the goal of improving model quality. On the contrary, our method is primarily oriented towards reducing energy consumption. We validate the effectiveness of our proposed architecture on road scenes, by addressing the recognition of the state of relevant traffic lights in each frame. In this scenario neighboring frames are very correlated, but the state of the object of interest can change abruptly. Therefore, the model must be able to leverage information from the past, but also be prepared to adjust predictions when the state of the traffic lights change.

The main contributions of this work is a novel approach to address the problem of energy-efficient video frame classification exploiting cooperation across time between two models of different size, and provide an experimental study on a real-world problem that shows how the method is promising and worth further investigation.

## 2. Method

The proposed architecture consists of two convolutional neural network streams with identical backbone architecture and depth, but that differ in terms of input resolution and backbone width. The main concept is to use the larger stream, denoted as $\Phi$, at regular intervals every $T$ frames. This stream is responsible for generating high-quality predictions and saliency maps. These saliency maps are then used to help the smaller and more efficient stream, denoted as $\Phi'$, in processing the subsequent $T$-1 frames. The temporal redundancies are exploited by mean of the saliency maps coming from the larger stream that are used to augment the intermediate representation of the smaller stream. The choice of the scheduling policy plays a pivotal role in determining the balance between quality and efficiency. Higher values of $T$ enhance the efficiency of the overall model but might result in a decrease in prediction quality.

Before providing some additional details on the proposed architecture, let us introduce some notation. Let us define the two backbones as a sequence of convolutional blocks $\Phi = [C_1, \ldots, C_N]$, $\Phi' = [C'_1, \ldots, C'_N]$. The output of each block $C_i$ or $C'_i$ is a feature map $f_i \in \mathbb{R}^{c_i \times h_i \times w_i}$ or $f'_i \in \mathbb{R}^{c'_i \times h'_i \times w'_i}$. $\Phi$ has a larger resolution and is wider than $\Phi'$, *i.e.*, $c_i \geq c'_i$, $h_i \geq h'_i$ and $w_i \geq w'_i$ $\forall i = 1, \ldots, N$. Finally, to establish cooperation between convolutional blocks $C_i$ and $C'_i$, we introduce *saliency gates* denoted as $G_i$ and *saliency propagation modules* denoted as $G'_i$. These components enable the exchange of information and foster collaborative processing between the two streams. Further details about their architecture will be discussed in the following paragraphs.

### 2.1. $\Phi$ and saliency gates

The saliency gates, that are responsible for computing saliency maps starting from a given hidden representations in $\Phi$, were introduced for the first time in [10] and further developed in [15]. The original purpose of these gates was to encourage the network to identify salient image regions earlier along the backbone to improve accuracy on image classification or semantic segmentation. In this work we

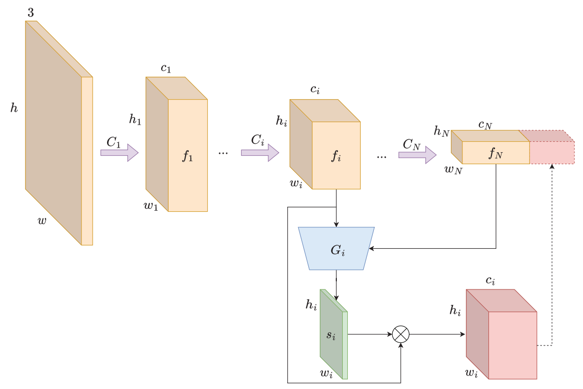

Figure 2. Architecture of $\Phi$ with a saliency gate $G_i$ after the $i$-th convolutional block.

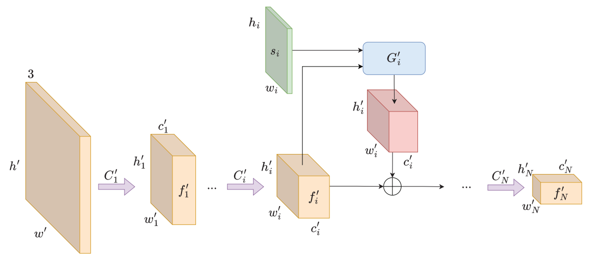

Figure 3. Architecture of $\Phi'$ with a saliency propagation module $G'_i$ in correspondence of the $i$-th convolutional block.

leverage this mechanism in a novel way, namely to share spatial priors from $\Phi$ to the smaller backbone $\Phi'$ through time.

The saliency gates can be attached along $\Phi$ after any of the convolutional blocks. Figure 2 illustrates the architecture of model $\Phi$. The saliency gate $G_i$ consist in a convolutional encoder that takes the current and the last representations, $f_i$ and $f_N$, and generates the saliency map $s_i$. To train the gate parameters along with the backbone, the saliency map is applied to the feature map $f_i$. The resulting feature map is properly resized by means of a spatial pooling, concatenated with $f_N$ and used as input to the classification head. The output of the classification head is then used in a standard cross-entropy loss function.

### 2.2. $\Phi'$ and saliency propagation module

The saliency propagation modules are responsible for propagating the prior knowledge about the scene, captured by $\Phi$ in the form of saliency maps, into $\Phi'$. Each saliency propagation module corresponds to a saliency gate, establishing a one-to-one relationship.

Figure 3 illustrates the architecture of model $\Phi'$. To take into account the temporal dynamics between frames, the saliency propagation module $G'_i$ applies an exponential decay operation to decrease the intensity of the saliency map. This decay is performed using the equation $\tilde{s}_i = s_i \cdot e^{-\frac{k}{\tau}}$, where $\tau$ represents the decay ratio, and $k$ is the temporal distance between frames. Once $\tilde{s}_i$ is computed and reshaped, it is applied element-wise to $f'_i$. The resulting tensor is then processed by three convolutional layers, each followed by batch normalization and a ReLU activation. These layers are responsible for adjusting the features, using the information encoded in $\tilde{s}_i$, and generating a tensor with the same shape as $f'_i$. The kernels of the convolutional layers ensure a sufficiently large receptive field, allowing the module to consider neighboring features to correct the spatial misalignment. The adjusted representation coming from $G'_i$ is added as a residual to the original representation $f'_i$. The resulting tensor is then used as input for the subsequent convolutional block $C'_{i+1}$, continuing the processing flow.

### 2.3. Training procedure

The whole architecture is trained in three steps. First, we train $\Phi$ to generate the saliency maps and perform the classification task. Then, we pre-train $\Phi'$ to perform the classification without saliency propagation modules, aiming to achieve better alignment between the two convolutional streams before enabling cooperation. Finally, we attach the propagation modules $G'_i$ to the backbone of $\Phi'$, which is trained to take into account the saliency maps $s_i$ generated by $\Phi$. During this phase of training, we use frames with a random temporal delay $k$ between $\Phi$ and $\Phi'$ to simulate what happens at inference time.

## 3. Experiments

The method we propose is well-suited for applications that require the sequential processing of video frames in energy-constrained environments. One such application is the relevant traffic light state recognition in driving scenes, which relies on images captured and processed by an onboard camera. The computational limitations of the camera and thermal considerations pose challenges during inference in terms of quality of the prediction and energy consumption. Furthermore, the task itself presents interesting challenges: on the one hand, the geometry of the scene, such as the existence and position of traffic lights relative to vehicles, is similar across time and is quite predictable. On the other hand, the state of the traffic light may change suddenly, making it infeasible to rely too much on occasional predictions. This task can be formulated as a ternary classification problem, in which for each frame one wants to predict if 1) there are *relevant* red lights, therefore the vehicle has to stop; 2) all relevant traffic lights are green; or 3) there are no relevant traffic lights. Note that in case of multiple traffic lights, that is very frequent at intersections, the model needs to identify which traffic signals should be considered not relevant.

In our experiments we combined two public datasets containing videos recorded from cameras or mobile phones mounted on a vehicle windshield: the DriveU Traffic Light

| Model | k = 0 | k = 1 | k = 2 | k = 3 |
|---|---|---|---|---|
| $L$ | **0.957** | 0.907 | 0.859 | 0.807 |
| Ours | 0.954 | **0.940** | **0.931** | **0.920** |

Table 1. Results on test, averaged over 3 runs, measuring performance on a frame with a temporal delay $k$.

Dataset (DTLD) [2] and the BDD100K [17] dataset. While both datasets have traffic light state annotations at the frame level, only DTLD includes information about which traffic lights are relevant to the ego-vehicle. Therefore, we manually annotated the relevance of each traffic light in the MOT subset of BDD. The final dataset includes 3,710 videos and about 50,000 frames extracted at 1 Hz from video sequences captured as vehicles approach intersections.

For all the experiments, we use images with resolutions of $1024\times512$ for $\Phi$ and $512\times256$ for $\Phi'$. The backbones are MobileNetV3-Large [7] where we set the *width multiplier* parameter at 1.0 for $\Phi$ and 0.1 for $\Phi'$ to lower the number of channels per layer. The number of parameters of $\Phi$ is over 3 millions, while $\Phi'$ has less than 50,000 parameters. For our application we found the optimal number of saliency gates is 2. We employ them after the 6-th and 10-th convolutional blocks, corresponding to approximately one third and half of the backbone depth. The exponential decay rate in the propagation modules is set to $\tau = 10$. We use frames with a random delay $k \in \{1, 2, 3\}$ during the training of the saliency propagation modules.

### 3.1. Results

We compare our method against two standard image classifiers based on MobileNetV3-Large: we denote with $L$ the model with a width multiplier $w = 1.0$ and input shape $1024 \times 512$ (analogous to $\Phi$), and with $S$ the model with $w = 0.1$ and input shape $512 \times 256$ (analogous to $\Phi'$). We run experiments for $k \in \{0, 1, 2, 3\}$, that, with the frame rate of 1 Hz in our data, means reusing information coming from up to 3 seconds in the past.

A first static comparison is reported in Table 1, where we show the performance of the large baseline $L$ and our proposed approach with different temporal delays. We use Average Precision (AP) on the *relevant red* class as our primary metric. For $L$, the results are computed predicting at time $k = 0$ and holding the predictions for $k > 0$, while for Ours we use $\Phi$ at $k = 0$ and $\Phi'$ for $k > 0$. As expected our method and the baseline have similarly good results for $k = 0$ since they use almost identical backbones. As $k$ grows, the results for $L$ naturally deteriorate. On the other hand, with our approach, leveraging the saliency maps from time $t$ allows the smaller $\Phi'$ to achieve much better performance for $k > 0$ with minimal additional compute.

Let us now focus on the dynamic setting, where we would like to assess the average prediction quality and

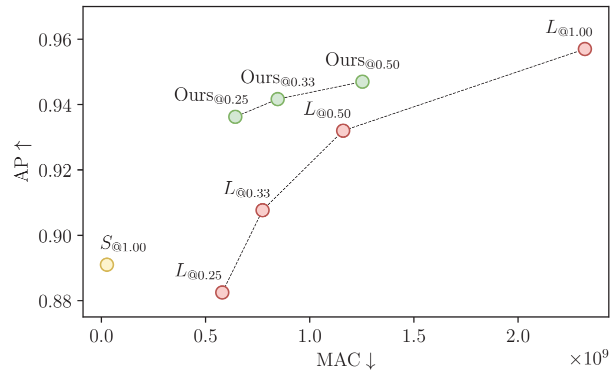

Figure 4. AP vs MACs (averaged over time).

the energy consumption of the methods when processing a stream of images. We denote with $L_{@\nu}$ and $S_{@\nu}$ the baselines obtained running $L$ and $S$ every $\frac{1}{\nu}$ frames, holding the prediction until the next inference. Similarly we denote with $\text{Ours}_{@\nu}$ our proposed approach in which $\Phi$ is run every $\frac{1}{\nu}$ frame, making the computed saliency maps available for the following frames to be used by $\Phi'$.

We evaluate both the performance and the efficiency of the models averaging over time the AP and the number of Multiply-and-accumulate (MAC) operations, respectively. Counting the MAC operations is a coarse, but common, proxy for evaluating the energy consumption in CNNs [3]. The results are shown in Figure 4. Notably, our approach leads to a significant decrease of the average MACs per frame at the expense of a small drop in terms of AP with respect to $L_{@1.0}$. On the other end of the spectrum, $S_{@1.0}$ has significantly worse quality, even if inference is performed on every frame, because of the lower input resolution and the smaller width of the network. This shows that our approach is able to effectively exploit the information from $\Phi$. It is worth noting that running $L$ at lower frequency is not enough. Indeed, analyzing the errors of $L_{@0.25}$, we verified that they are often caused by the sudden changes of traffic light states that happen in the meanwhile; whereas our proposed approach corrects the majority of these mistakes using $\Phi'$ to reverse the decision from $\Phi$, when necessary.

## 3.2. Architecture design

In this section, we provide a rationale for the architectural choices we made. The positions where we employed the gates correspond to the depths where the backbone scales down the resolution, allowing us to collect spatial priors at multiple scales. For this specific task, we experimentally notice that employing more than two gates proved to be ineffective. For example, attaching an additional gate approximately at two-thirds of the backbone depth and studying the gradients of the output with respect to the features extracted by this gate revealed that the variations in those

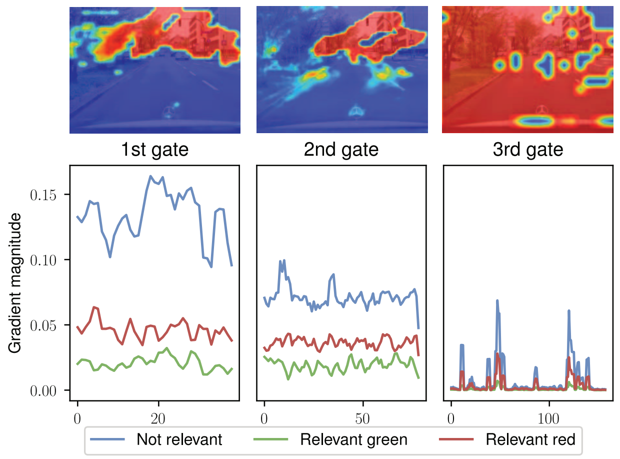

Figure 5. To assess whether additional saliency gates bring an improvement, we consider an architecture with 3 gates and compute the gradient of the model output with respect to the features generated by each gate. The output is sensitive to variations in the features generated by the 1st and 2nd gate, and their saliency maps identify good spatial priors to propagate to $\phi'$. On the contrary, the 3rd gate has a smaller impact on the output, and the corresponding saliency map fails to identify the important regions in the image.

| Saliency fusion | k = 1 | k = 2 | k = 3 |
|---|---|---|---|
| Extra channel | 0.931 | 0.925 | 0.916 |
| Element-wise | 0.940 | 0.931 | 0.920 |

Table 2. Ablation study on saliency map fusion.

features had minimal impact on the output. Consequently, the corresponding saliency maps failed to identify important regions in the images, as showed in the Figure 5.

In the proposed architecture, the saliency maps collected by $\Phi$ are applied element-wise to the features computed by $\Phi'$ before being processed by the propagation gates. We also conducted experiments by incorporating the saliency maps as additional channels to the existing features. The results in Table 2 show that element-wise fusion of saliency maps is slightly superior in accuracy, and with a slightly smaller number of MAC operations.

## 4. Conclusion

In this paper, we presented a method to achieve energy-efficient video frame classification by leveraging temporal redundancies using saliency maps. The experiments showed the effectiveness of this approach in real-world applications, such as traffic light state recognition. Our method can easily be generalized to any convolutional neural network in tasks that can benefit from temporal awareness. Future work will involve testing with various baselines, architectures, and tasks, as well as experiments considering latency in a streaming setting [11].

## References

[1] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? *Proceedings of Machine Learning and Systems*, 2:129–146, 2020. 1

[2] Andreas Fregin, Julian Müller, Ulrich Krebel, and Klaus C. J. Dietmayer. The driveu traffic light dataset: Introduction and comparison with existing datasets. *IEEE International Conference on Robotics and Automation*, pages 3376–3383, 2018. 3

[3] Eva García-Martín, Crefeda Faviola Rodrigues, Graham Riley, and Håkan Grahn. Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, 134:75–88, 2019. 4

[4] Jianping Gou, B. Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129:1789 – 1819, 2020. 1

[5] Amirhossein Habibian, Davide Abati, Taco Cohen, and Babak Ehteshami Bejnordi. Skip-convolutions for efficient video processing. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2694–2703, 2021. 1

[6] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:7436–7456, 2021. 1

[7] Andrew G. Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3. *IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019. 3

[8] Ping Hu, Fabian Caba Heilbron, Oliver Wang, Zhe L. Lin, Stan Sclaroff, and Federico Perazzi. Temporally distributed networks for fast video semantic segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8815–8824, 2020. 1

[9] Samvit Jain, Xin Wang, and Joseph E. Gonzalez. Accel: A corrective fusion network for efficient semantic segmentation on video. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8858–8867, 2019. 1

[10] S Jetley, NA Lord, Namhoon Lee, and PHS Torr. Learn to pay attention. In *6th International Conference on Learning Representations*, 2018. 2

[11] Mengtian Li, Yu-Xiong Wang, and Deva Ramanan. Towards streaming perception. In *European Conference on Computer Vision*, 2020. 4

[12] Feng Liang, Ting-Wu Chin, Yangxu Zhou, and Diana Marculescu. Ant: Adapt network across time for efficient video processing. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 2602–2607, 2022. 1

[13] Seoung Wug Oh, Joon-Young Lee, N. Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. *IEEE/CVF International Conference on Computer Vision*, pages 9225–9234, 2019. 1

[14] Amin Sabet, Jonathon Hare, Bashir M. Al-Hashimi, and Geoff V. Merrett. Temporal early exits for efficient video object detection. *ArXiv*, abs/2106.11208, 2021. 1

[15] Jo Schlemper, Ozan Oktay, Michiel Schaap, Mattias P. Heinrich, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention gated networks: Learning to leverage salient regions in medical images. *Medical Image Analysis*, 53:197 – 207, 2018. 2

[16] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013. 1

[17] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2633–2642, 2020. 3