Abstract

This paper introduces a new lightweight method for image recognition. ImageSig is based on computing signatures and does not require a convolutional structure or an attention-based encoder. It is striking to the authors that it achieves: a) an accuracy for 64 X 64 RGB images that exceeds many of the state-of-the-art methods and simultaneously b) requires orders of magnitude less FLOPS, power and memory footprint. The pretrained model can be as small as 44.2 KB in size. ImageSig shows unprecedented performance on hardware such as Raspberry Pi and Jetson-nano. ImageSig treats images as streams with multiple channels. These streams are parameterized by spatial directions. We contribute to the functionality of signature and rough path theory to stream-like data and vision tasks on static images beyond temporal streams. With very few parameters and small size models, the key advantage is that one could have many of these “detectors” assembled on the same chip; moreover, the feature acquisition can be performed once and shared between different models of different tasks - further accelerating the process. This contributes to energy efficiency and the advancements of embedded AI at the edge.

1. Introduction

Image recognition represents the backbone for all vision tasks in which its accuracy and efficiency are seminal for tasks such as classification, object detection, or semantic segmentation. Advances in image recognition have been achieved relying on very deep convolution models [1–4]. Most recently, Attention-based models such as Vision Transformers (ViT) [5] have also shown strong progress for vision tasks at scale when trained in large datasets. While both approaches (Convolution-based and attention-based models) have yielded high accuracies in most benchmark datasets (e.g. ImageNet [6]), this performance often comes at the expense of the model weights, the number of parameters and subsequently the time and resources needed for training and inference. Even with mobile-friendly methods (e.g. MobileNets [7–9]) and post-training quantization techniques [10, 11], the number of parameters still exceeds millions. This makes it a challenge for embedding AI for microcontrollers or devices with minimal computational resources when implementing an entire pipeline of multiple vision models.

In this paper, we introduce a new method, called ImageSig, that is suitable for training lightweight vision models. Based on rough path theory [12, 13], we treat a single image as a stream of paths over a virtual time, whereas pixels evolve instantaneously over its spatial attributes. The architecture of ImageSig relies on extracting a unique signature from these paths that can be trained directly by a fully connected layer for classification without the need for extracting features based on expensive operations such as 2D convolution layers or attention-based structures. We reveal that ImageSig can provide a faster and efficient way for training deep models with very few parameters and model footprint that is often less than 1 MB without quantization and drops as low as 44.2 KB after quantization.

2. Related work

There are three domains that are relevant to the stated issue. We have reviewed the relevant work, and here we highlight the key challenges where further research is needed.

Lightweight convolution models: Training fast, efficient, and lightweight convolution models is ongoing research [7–9]. There are several approaches that are of particular interest to scholars and have been intensively addressed in the literature. First, lightweight models have been achieved by training smaller networks of a larger one of the same family (e.g. ResNet18 vs ResNet 101 [1]). However, this approach often comes at the expense of accuracy when compared to the larger networks of the same family. The second approach is by training a larger network and distillates its knowledge to a smaller network with fewer parameters [14]. Transferring knowledge from teacher to student architecture was key in many applications. However, on others, it remains an expensive solution for train-
ing a given task. Nevertheless, the weights of a student network do not often show drastic changes in the overall number of parameters. Instead, it is often reported as a slightly lighter version of a given teacher network. Third, by relying on various factorization-based methods (e.g., separable convolutions [2], dimension-wise convolutions [15]), various mobile-friendly methods have been achieved for lightweight vision tasks such as MobileNets [7–9], ShuffleNetV2 [16], MNASNet [17], and ESPNetv2 [18], which have been proven to be more efficient in comparison to deep convolution models. Yet, these models have been reported with millions of parameters for training. Nevertheless, this lightweight comparing to deeper models, comes at the expense of accuracy. Last, post-training pruning [19] and quantization [10, 11, 20, 21] have shown success in improving model efficiency through reduced precision arithmetic without compromising its overall utility and accuracy.

**Transformers, patch and position embeddings:** Applying transformers [22] to visual perceptions has opened the door for understanding vision tasks beyond the need for a convolutional structure. In Vision Transformers (ViT) [5], a given Image is subdivided into small and fixed-size patches in which their positions are spatially registered by embeddings and then feedforward to a transformer encoder. Images are transformed into patch embeddings, in which a given 2D image \( X \in \mathbb{R}^{H \times W \times C} \) is reshaped and flattened to 2D patches \( X_P \in \mathbb{R}^{N \times (P^2 \times C)} \) where \((H, W)\) is the resolution of the given image, \(C\) is the number of channels, \(P, P\) is the resolution of the given patch, and \(N\) represents the number of patches, given that \(N = HW/P^2\). In addition to the patch embeddings, a learnable 2D position embedding is added to preserve the positional information of each patch without inductive bias; the spatial relations among the patches need to be learned from scratch. While this approach has achieved 88.55% in the ImageNet dataset, it requires a large network with millions of parameters and expensive computational resources (30 days on a TPUv3 with 8 cores) trained exclusively on large datasets. To allow ViT to be trained on small-sized datasets, a Compact Convolution Transformer (CCT) [23] is introduced. Unlike ViT, the CCT model introduced a convolution block, in which image patching and embeddings are implemented to its output to reduce the latent representation of the transformer where \(X_0 = \text{MaxPool}(\text{ReLU}(\text{Conv2d}(X)))\). This allows the model to learn from relatively smaller datasets based on the inherited inductive bias and spatial relationships of the pixels. This also allows changing the resolution of the given image without increasing the number of parameters of the model. However, this remains at the expense of the sequence length and, subsequently, the computational resources. On the other hand, a Perceiver [24] is introduced to leverage an asymmetric attention mechanism to iteratively map a byte array (at a pixel level) as an input to a smaller latent array. This approach allows the transformer to handle a large input size of various modalities by uncoupling the depth of the transformer and the exponential size of the latent representation of the transformer based on the input resolution. While there are also several modifications of transformer models that leverage one or more of the aforementioned concepts [25–31], the challenges remain on how to handle large size byte arrays to the latent representation of a given transformer encoder with respect to the number of the parameters and subsequently computational resources.

**Signature-based methods:** Based on rough path theory, a signature is a unique summary of a given path [12, 13, 32–34]. What makes signature a robust method for describing a path is its invariant characteristics to the reparameterisation of the given path. This provides a natural feature of linear functionals that only capture the key elements of the given path by mapping the orders of the information of the stream instead of mapping precisely the location of the path at each instance [13]. Transforming datasets into streams is a modelling issue that requires assumptions regarding to how the system is altered once new information arrives [32]. There are no signature-based methods for image recognition. However, Signature transforms with deep learning have been introduced [35], in which different scholars have utilised it for various machine learning applications. For example, a deep model for recognising online handwritten characters from temporal paths has been introduced [36]. Based on pose estimation, a signature transform has been applied to recognise human actions based on their landmarks [37]. This paper contributes to extending rough path theory and signature transform directly to vision tasks and stream-like data regardless of the presence of an explicit path.

### 3. Methodology

#### 3.1. Image signature

In this research, we contribute to how signature methods can be applied to stream-like data. Nevertheless, we assume that signature transforms can be perceived as the equivalent of spatial embeddings that can offer orders for a byte array (at a pixel level) while reducing its dimensionality. We assume that \(V\) is a \(d\)-dimensional vector (Banach) space of basis \(B = v_1, \ldots, v_d\) and defined based on tensor algebra as:

\[
T(V) : = \bigoplus_{k=0}^{\infty} V \otimes^k \tag{1}
\]

given that \(V \otimes^0 : = \mathbb{R}\). We denote the projection of \(T(V) \rightarrow V \otimes^n\) by \(\pi_n\) where \(n \in \mathbb{N}_0\) and the truncated tensor algebra is defined as:

\[
T(n)(V) : = \bigoplus_{k=0}^{n} V \otimes^k \tag{2}
\]
An image is transformed into paths by imagining a given 2D image \( X \in \mathbb{R}^{H \times W \times C} \) as patches of streams \( \Omega_i \subset V \) of discontinuous paths \( \gamma : [a, b] \rightarrow V \) where a given pixel \((i)\) evolves, in magnitude, from left to right over a virtual time to the adjacent pixel given that \(i \in W\). In this order, changes occur instantaneously given that signature is invariant to reparameterisations. The channels of a given image \( C \) represents the depth of a given path (See Fig. 1).

The truncated signature of \( \gamma \) at a given depth \( N \) is defined as:

\[
S_{a,b}(\gamma) = \bigoplus_{n=0}^{N} S_{a,b}^n(\gamma),
\]

(3)

given that: \( S_{a,b}^n(\gamma) = \frac{1}{n!} (\gamma_b - \gamma_a) \otimes^n \)

So the signature transform given that \( Sig^N = S(\mathbb{R}^d) \rightarrow \prod_{n=1}^{N} (\mathbb{R}^d) \otimes^n \) is defined as:

\[
Sig^N(X) = (f \cdot \int_{\Omega_{x,y} \in \hat{\gamma} X \in \mathbb{R}^d} \otimes (\mathbb{R}^d))_{i \in \mathbb{R}^d (24hi) \cdot (24di)}_{i \in \mathbb{R}^d (24hi) \cdot (24di)}
\]

(4)

And the log signature of \( \gamma \) is defined as:

\[
\log S_{a,b}(\gamma) = \sum_{n=0}^{N} (-1)^{n-1} \bigoplus_{n=0}^{N} \left( \hat{S}_{a,b}^n(\gamma) \right) \otimes^n
\]

(5)

where \( S_{a,b}^n(\gamma) = 1 \) and \( \hat{S}_{a,b}^n(\gamma) := \bigoplus_{n=1}^{N} S_{a,b}^n(\gamma) \).

The collection of paths \( N \subset \mathbb{V}_1[a, b] : V \), given that \( N \in H \) can be summarised as \( \delta_N = \frac{1}{N!} \sum_{\gamma \in N} \delta_\gamma \). The expected signature of the collective paths that represents an image is the map \( \mathbb{S} : \mathcal{P} \mathbb{V}_1([a, b] : V) \rightarrow T(V) \) in a probability (Borel) space and is computed as:

\[
\mathbb{S}(\mu) := \mathbb{E}_{\mu} \left[ S_{a,b}(X) \right] = \bigoplus_{n=0}^{N} \mathbb{E}_{\mu} \left[ S_{a,b}^n(X) \right]
\]

(6)

We highlight that the proposed image signature summarises and reduces the dimensionality of a given image \( X \) into \( \hat{X} \in \mathbb{R}^{H \times Q} \), where \( Q = \sum_{n=1}^{N} C^n \). It is worth mentioning that we also considered looking at a given image as two-directional streams of paths, horizontally and vertically, in which we concatenated both outcomes where \( \hat{X} \in \mathbb{R}^{2x(H \times Q)} \) and we refer to this method as two-directional streams. Second, the aspect ratio of a given image can be taken into consideration without the need for finding an equal dimension for both \( H \) and \( W \). Last, while the computed signature is variant to \( W \), the dimensions of \( \hat{X} \) is invariant to \( W \). This feature allows training a model in a dataset of images of various aspect ratios, providing that \( H \) remains constant.

3.2. Signature augmentation

While the computed signature is invariant to reparameterisation, it remains variant to change in colour, rotation, and displacement of a given image. Nevertheless, the order in which images are transformed is a modelling choice. Accordingly, similar to image augmentation techniques [38–40], we applied several data augmentation techniques such as adding Gaussian noise, brightness, horizontal flipping, rotation, and colour invert to images before computing their respective signature without altering a given class of an image.

3.3. Encoder architecture

After computing image signature, we experimented with three minimalist architectures to show the robustness of ImageSig; one architecture is based on using only a single FC layer to classify a given image, the second comprises a convolution encoder, and the third is based on a transformer encoder.

A single FC-based model: This model represents the simplest approach for training a vision model. After computing the image signature and flattening its output, the architecture of the model relies only on a single Fully connected layer (FC) of 50 neurons activated by a ReLU function, followed by an output softmax layer of neurons equal to the number of classes in a given dataset. Fig. 1 shows the overall architecture of ImageSig.

A 1D convolution-based model: As shown in Fig. 1, an encoder can be added to extract features and downsample the computed signature before the FC layer. The objective of this architecture is to evaluate the effect of adding a convolution block on the overall performance, bearing in mind the trade-off between performance and the overall weight of the model. Accordingly, we used a simple convolution block comprised of two CNN 1D layers, in which each layer is followed by a Max-pooling 1D layer. After the last pooling layer, the model is flattened and feedforward to a single FC layer similar to the aforementioned architecture.

Attention-based model: We also feedforward the computed signature (without the Flatten layer) to a transformer encoder similar to the one implemented in CCT model [23], but without any patch or position embeddings. And before the multi-head attention layers, we used CNN1D layers (similar to the aforementioned architecture) instead of the CNN2D layers.

3.4. Objective loss

We penalise the introduced method based on a cross-entropy coupled with a focal loss [41] to account for class imbalance for data sets of skew representation for each class. For binary classification, given that \( y \in \{0, 1\} \) and \( \hat{p} \in [0, 1] \), the objective loss \( L \) is defined as:

\[
L(y, \hat{p}) = -\alpha y (1 - \hat{p})^\gamma \log(\hat{p}) - (1 - y)\hat{p}^\gamma \log(1 - \hat{p})
\]

(7)

where \( \gamma \) represents the focusing parameter that postulates the confidence level for the contributions of correct predictions to the overall loss (the higher the \( \gamma \), the higher the
rate for down-weighting examples that are easy to classify) and \( \alpha \) represents a hyperparameter that specifies the trade-off between recall and precision by increasing or decreasing the weight of errors for the positive class (when \( \alpha = 1 \) is the same as no weighting).

4. Experiments

We report on over 40 ImageSig models, in addition to base models, trained on four datasets with different encoder architectures and various hyperparameters which have been deployed for different hardware (e.g. Raspberry Pi).

4.1. Datasets

We aim to apply our introduced method to applications to the real-world (e.g. anomaly detection from images). Accordingly, we have applied ImageSig to both benchmarked and custom datasets that are relatively small in size and imbalanced in classes which are common issues when it comes to practical applications (See Fig. 2). On the other hand, it is often under-reported how the introduced state-of-the-art methods perform in real-world applications beyond benchmarked datasets. Furthermore, we also aim to show how our method performs when it compares, for instance, to transformers that usually require large datasets for training.

Fire dataset: Finding a relatively large dataset that can represent anomaly detection such as fire detection from images remains a challenge. Accordingly, we have created our dataset that represents various conditions of the environment. The dataset comprises 18,912 images that are downloaded from the web, specifically Google Images.

This dataset is used for training, validating and testing our method. We manually labelled the dataset to fire and no fire.

Fog benchmark dataset [42]: We also implemented several experiments for fog detection from street-level images, which is a crucial task for road safety. We used the dataset introduced in WeatherNet ensemble model. Also, we compare our results to their FogNet model. The dataset,
for fog detection alone, comprises 4,345 RGB images distributed into 718 and 3,627 for positive and negative classes, respectively.

Concrete crack benchmark dataset [43, 44]: It comprises 40,000 RGB images distributed evenly between positive and negative cases for concrete crack detection.

CelebA benchmark dataset [45]: It comprises 202,599 RGB images of several facial attributes. We conducted several experiments for detecting some facial attributes (e.g., wearing hat, bald, blond hair).

In supplementary, we show the sample size per class and its weight in both training and testing sets for the four datasets. It is worth mentioning that the class weights represent \( \alpha \) values in the aforementioned focal loss function. During training and in absence of validation set, we have split the training set to train and validate sets in a ratio of 80-20\% to train and validate the models per each training cycle (epoch). During inference, we used the test set, which the models have never seen during training or validation, for evaluating our pretrained models, computing metrics, and comparing their results.

Fig. 3 shows a sample of images (from fire dataset) and their unique image signature (depth = 4), which shows how a given image can be translated to a stream of paths and projected to a lower dimension signature.

4.2. Evaluations

After training several models, we evaluated their pretrained weights on the test-set for a given dataset, which the models have never seen during training and validation. Several metrics are computed to evaluate the trade-off between a given model’s performance and efficiency for computations and inference. Based on the aforementioned objective loss, we computed accuracy (acc), Average Precision (AP) and F1 score, which is defined as: \( F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \). We also computed the Receiver operating characteristic (ROC) curve to evaluate the trade-off between true and false-positive rates.

To reflect on a given model’s computational efficiency, we have calculated the number of parameters (params) for each model, its disk size, and the Floating-point Operations Per Second (FLOPS) for computing the operations of a given model in a batch size (1).

We compared our results to state-of-the-art models (e.g., ResNet50, VGG16, and MobileNet) to reflect on both: performance and efficiency for training and computation resources. We also trained a base model of a single FC layer on images directly to show the effectiveness of ImageSig with the same FC layer unequivocally.

5. Results

In Table 1, we compare our methods with other state-of-the-art models when trained and tested on the fire dataset. It shows that our methods outperform other models when trained on the same image resolution (64,64). The table shows that ImageSig with a single FC layer is 3.6 times more accurate than relying only on a single FC layer without signature and achieves as high as VGG16 model, but with almost 50 times fewer parameters when trained on the same input size. While a higher accuracy can be achieved when training, for instance, ResNet50 model on a larger image resolution, the introduced method outperforms all other methods when it comes to model’s efficiency, the number of parameters, training time, and FLOPS. In fact, a fully trained model of ImageSig + CNN1D has achieved the highest performance, for image resolution (64,64), with the lowest disk size (0.6 MB), fewer parameters (37,112) and FLOP of 1.69 million, showing the robustness and efficiency of the introduced method.

Table 2 compares the results of ImageSig with a state-of-the-art model (FogNet) that relies on ResNet50. Similar to the previous results, while ResNet50 trained on a larger input can achieve higher accuracy, we show that ImageSig achieves a better performance in terms of AP and F1 scores with fewer parameters.

For anomaly detection, understanding the relationship between true and false positive rates is crucial. Fig. 4 shows the ROC curve for the introduced architectures of ImageSig, FC, CNN1D, Transformer, which shows a uniform relationship between true and false positive and high AUC values above 0.9 for fire and fog datasets.

Furthermore, we also compare our best results to other state-of-the-art models on the remaining two benchmarked datasets. Table 3 shows the results for detecting three facial attributes (wearing hat, bald and blond hair) on the test set of CelebA dataset. ImageSig shows competitive results in comparison to other deeper base models, bearing in mind the uneven trade-off between the difference in accuracy and the number of parameters for a given base model. Last, Table 4 shows the results for detecting concrete crack, in which ImageSig has achieved 98.7\% accuracy with an in-
put size (64 X 64) and at least 620 times fewer parameters when compared to the other base models.

6. Ablation studies

We trained several models with different hyperparameters to evaluate the performance of the introduced method. All models are trained on the fire dataset with only a single FC layer after flattening the computed signatures. The aims of these studies are to understand the effect of the input size, signature depth and the type of signature on both accuracy and number of parameters and subsequently the memory footprint for a given model. Fig. 5 shows the relationship between image resolution, signature depth and the overall performance of a given model.

Images as two directional streams: Table 1 also shows the effect of modelling images as streams by computing and combining signatures for both directions (horizontal and vertical paths) on the overall generalization of ImageSig. It shows an increase of 1.73% in accuracy when training ImageSig + CNN1D on fire dataset. However, this increase comes at the expense of doubling FLOPS (from 1.69 million to 3.49 million).

The effect of signature depth: The depth of the truncated signature directly affects the performance of a given model. Table 5 shows the results of 16 models of different image resolutions and signature depths. For a given image resolution, the increase in signature depth leads to a direct increase in the performance of a given model in a non-linear relationship, whereas an increase in depth after certain points could exacerbate the performance of the model. On the other hand, it is worth mentioning that this increase in depths (within the linear part of the relationship) comes at the expense of the number of parameters and consequently the training time and the model size. Generally, for this given classification task, it seems that a depth of 4 provides the top score in relation to the number of parameters and the model’s disk size.

The effect of the input size: While the effect of the input size on the performance of a given model is less significant when it is compared to the signature depth, a larger input size seems to provide a trained model with better accuracy at a shallower depth of a signature. This also benefits the trade-off between performance and the number of parameters. For example, in Table 5, a model of resolution (32 X 32) at depth 5 achieves a similar accuracy for a model of resolution (64 X 64) at depth 4, whereas the number of parameters in the latter model is 1.5 less than the aforementioned one. Accordingly, a smaller input size does not necessarily mean less number of parameters, but the trade-off between the input size and the depth of computed signature requires optimisation for a given task.

Signature vs logarithmic of signature: While the logarithmic version of signature leads to normalisation and dimensionality reduction of a given input, we found that the linear version of signature achieves better performance for a given model. Table 6 shows the results of four models at two different image resolutions and signature depth, in which we have changed only the type of signature for each image resolution. The table shows that the signature outperforms the log signature for both image resolutions at different depths. However, the log signature can provide models with fewer parameters and less model size and training time.

Post-training quantization: Table 7 shows the results of model size before and after post-training dynamic range quantization for state-of-the-art and imageSig methods. It shows unprecedentedly that a full vision model can reach a few kilobytes (44.2 KB) which is suitable for edge devices with limited memory and computational resources. It also shows that even after compressing the state-of-the-art methods, their size remains larger than ImageSig models without quantization.

Embedded AI at the edge: Table 8 shows that our methods outperform MobileNetV2 when it comes to efficiency, in terms of Frame Per Second (FPS), for embedded AI at the edge. It shows that our method achieves an unprecedented 15.4 FPS on Raspberry Pi v4 (CPU) for a vision model including all necessary data structuring and preprocessing and 142.3 FPS when considering only the performance of the model. It also shows a real-time performance that outperforms MobileNetV2 for Jetson nano with GPU acceleration.

Implementation details and further studies regarding performance, number of neurons in the FC layer and batch size are addressed in the supplementary section.

7. Remarks and future work

We introduced a new method, called ImageSig, for ultra-lightweight image recognition that has been tested on several datasets without any special fine-tuning for a given task. ImageSig relies on extracting a unique signature of a given
Table 1. Comparing the results of different methods on the test set of fire dataset. Both ImageSig models are trained from scratch with a signature depth = 4 and with a batch size (3000). *All convolutional models (only) are trained on this dataset via transfer learning with ImageNet weights. 1 Training the same FC layer architecture on images directly without signature. 2 It refers to computing signature for two directions of a given image. Models are trained on a single GPU (Titan V).

Table 2. Comparing the results of different methods on the test set of fog dataset. All ImageSig models are trained similar to the ones trained for fire detection without any additional fine-tuning. 1 Training the same FC layer architecture on images directly without signature.

Table 3. Comparing the results of different models on the test set of CelebA dataset. The results of base models are obtained from [45]. ANet is based on AlexNet, with estimated params ≈ 61 M. 1 Refers to “without pretraining”. * Models obtain facial parts based on landmark points. 2 Refers to ImageSig + CNN1D, trained similar to the models for the aforementioned datasets without any special fine-tuning.

Table 4. Comparing the results of different models on the test set of Concrete crack dataset. The results of ResNet models are obtained from [44]. 1 Refers to ImageSig + CNN1D, trained similar to the models for the aforementioned datasets.
Table 5. Comparing the effects of different image resolutions and signature depth on the performance. All models are trained with a single FC layer of 50 neurons for 200 epochs on fire dataset with a batch size (1024).

<table>
<thead>
<tr>
<th>Resolution</th>
<th>Depth (N)</th>
<th>acc (%)</th>
<th>Params</th>
<th>Model size</th>
<th>Training time</th>
<th>FLOPS (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(32,32)</td>
<td>1</td>
<td>83.51</td>
<td>4,952</td>
<td>84 KB</td>
<td>1.43 min</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>88.45</td>
<td>19,352</td>
<td>253 KB</td>
<td>1.21 min</td>
<td>0.04</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>90.10</td>
<td>62,552</td>
<td>759 KB</td>
<td>1.82 min</td>
<td>0.12</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>91.54</td>
<td>192,152</td>
<td>2,278 KB</td>
<td>4.16 min</td>
<td>0.38</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>92.31</td>
<td>580,952</td>
<td>6,834 KB</td>
<td>10.17 min</td>
<td>1.16</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>92.66</td>
<td>1,747,352</td>
<td>20,994 KB</td>
<td>16.34 min</td>
<td>3.49</td>
</tr>
<tr>
<td>(64,64)</td>
<td>1</td>
<td>83.45</td>
<td>9,752</td>
<td>140 KB</td>
<td>1.54 min</td>
<td>0.02</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>89.17</td>
<td>38,552</td>
<td>478 KB</td>
<td>1.94 min</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>90.80</td>
<td>124,952</td>
<td>1,490 KB</td>
<td>3.21 min</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>92.36</td>
<td>384,152</td>
<td>4,528 KB</td>
<td>5.22 min</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>92.18</td>
<td>1,161,752</td>
<td>13,640 KB</td>
<td>17.42 min</td>
<td>2.32</td>
</tr>
<tr>
<td>(128,128)</td>
<td>1</td>
<td>83.43</td>
<td>19,352</td>
<td>258.5 KB</td>
<td>1.52 min</td>
<td>0.04</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>89.17</td>
<td>76,952</td>
<td>949.7 KB</td>
<td>1.30 min</td>
<td>0.15</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>90.37</td>
<td>249,752</td>
<td>3,023 KB</td>
<td>2.81 min</td>
<td>0.50</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>92.42</td>
<td>768,152</td>
<td>9,244 KB</td>
<td>7.46 min</td>
<td>1.53</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>92.50</td>
<td>2,323,352</td>
<td>27,906 KB</td>
<td>22.26 min</td>
<td>4.65</td>
</tr>
</tbody>
</table>

Table 6. Comparing the effects of the type of image signature on the performance. All models are trained with a single FC layer with 50 neurons.

<table>
<thead>
<tr>
<th>Resolution</th>
<th>Sig. type</th>
<th>acc (%)</th>
<th>Params</th>
<th>Model size</th>
</tr>
</thead>
<tbody>
<tr>
<td>(32,32), N=5</td>
<td>Sig.</td>
<td>92.31</td>
<td>580,952</td>
<td>6,834 KB</td>
</tr>
<tr>
<td></td>
<td>Log Sig.</td>
<td>89.04</td>
<td>128,152</td>
<td>1,582 KB</td>
</tr>
<tr>
<td>(64,64), N=4</td>
<td>Sig.</td>
<td>92.36</td>
<td>384,152</td>
<td>4,528 KB</td>
</tr>
<tr>
<td></td>
<td>Log Sig.</td>
<td>88.91</td>
<td>102,552</td>
<td>1,228 KB</td>
</tr>
</tbody>
</table>

Table 7. Comparing the effects of the type of image signature on the performance. All models are trained with a single FC layer with 50 neurons.

<table>
<thead>
<tr>
<th>Method</th>
<th>Hardware</th>
<th>FPS</th>
<th>FPS*</th>
</tr>
</thead>
<tbody>
<tr>
<td>MobileNetV2</td>
<td>RaspberryPi4</td>
<td>13.9</td>
<td>7.4</td>
</tr>
<tr>
<td></td>
<td>Jetson-nano 3</td>
<td>46.9</td>
<td>19.8</td>
</tr>
<tr>
<td></td>
<td>RTX 2080ti</td>
<td>62.2</td>
<td>37.27</td>
</tr>
<tr>
<td>ImageSig + CNN1D</td>
<td>RaspberryPi4</td>
<td>142.3</td>
<td>15.4</td>
</tr>
<tr>
<td></td>
<td>Jetson-nano 3</td>
<td>442.8</td>
<td>24.3</td>
</tr>
<tr>
<td></td>
<td>RTX 2080ti</td>
<td>1008.4</td>
<td>90.69</td>
</tr>
</tbody>
</table>

Table 8. Performance of AI at the Edge. All tested models are trained on fire dataset and converted to tflite. 1 model with resolution (64,64) and depth = 4, trained on fire dataset. 2 Raspberry pi CPU arm71 linux with 4 GB Ram. 3 Jetson nano includes a GPU and 4 GB Ram. * This metric includes data structuring and prepossessing, in addition to the model inference.

8. Acknowledgments

This work was supported in part by the EPSRC [grant number EP/S026347/1], in part by The Alan Turing Institute under the EPSRC grant EP/N510129/1, in part by The Alan Turing Institute’s Data Centric Engineering Programme under the Lloyd’s Register Foundation grant G0095, in part by The Alan Turing Institute’s Defence and Security Programme, funded by the UK Government, in part by The Alan Turing Institute’s Office of National Statistics Programme, funded by the UK Government and in part by the Hong Kong Innovation and Technology Commission (InnoHK Project CIMDA).
References


