# YOLO Nano: a Highly Compact You Only Look Once Convolutional Neural Network for Object Detection

**Alexander Wong**[1,2], **Mahmoud Famuori**[1,2], **Mohammad Javad Shafiee**[1,2]
**Francis Li**[2], **Brendan Chwyl**[2], **and Jonathan Chung**[2]
[1]Waterloo Artificial Intelligence Institute, University of Waterloo, Waterloo, ON, Canada
[2]DarwinAI Corp., Waterloo, ON, Canada

## Abstract

Object detection remains an active area of research in the field of computer vision, and considerable advances and successes has been achieved in this area through the design of deep convolutional neural networks for tackling object detection. Despite these successes, one of the biggest challenges to widespread deployment of such object detection networks on edge and mobile scenarios is the high computational and memory requirements. As such, there has been growing research interest in the design of efficient deep neural network architectures catered for edge and mobile usage. In this study, we introduce YOLO Nano, a highly compact deep convolutional neural network for the task of object detection. A human-machine collaborative design strategy is leveraged to create YOLO Nano, where principled network design prototyping, based on design principles from the YOLO family of single-shot object detection network architectures, is coupled with machine-driven design exploration to create a compact network with highly customized module-level macroarchitecture and microarchitecture designs tailored for the task of embedded object detection. The proposed YOLO Nano possesses a model size of $\sim$4.0MB ($>$15.1$\times$ and $>$8.3$\times$ smaller than Tiny YOLOv2 and Tiny YOLOv3, respectively) and requires 4.57B operations for inference (>34% and $\sim$17% lower than Tiny YOLOv2 and Tiny YOLOv3, respectively) while still achieving an mAP of $\sim$69.1% on the VOC 2007 dataset ($\sim$12% and $\sim$10.7% higher than Tiny YOLOv2 and Tiny YOLOv3, respectively). Experiments on inference speed and power efficiency on a Jetson AGX Xavier embedded module at different power budgets further demonstrate the efficacy of YOLO Nano for embedded scenarios.

## 1 Introduction

An active area in the field of computer vision is object detection, where the goal is to not only localize objects of interest within a scene, but also assign a class label to each of these objects of interest. Considerable recent successes in the area of object detection stems from modern advances in deep learning [8, 7], particularly leveraging deep convolutional neural networks. Much of the initial focus was on improving accuracy, leading to increasingly more complex object detection networks such as SSD [11], R-CNN [2], Mask R-CNN [3], and other extended variants of these networks [6, 9, 18]. While such networks demonstrated state-of-the-art object detection performance, they were very challenging, if not impossible, to deploy on edge and mobile devices due to computational and memory constraints. In fact, even faster variants such as Faster R-CNN [15] have inference speeds at low single-digit frame rates when running on embedded processors. This greatly limits the widespread adoption of such networks for a wide range of applications such as unmanned aerial vehicles, video surveillance, autonomous driving where local embedded processing is required.

To address this challenge of achieving embedded object detection, there has been a growing interest in the exploration and design of highly efficient deep neural network architectures for object detection that are more well-suited for edge and mobile devices [12, 13, 14, 23, 4, 17]. A particularly interesting

EMC2: 5th Edition Co-located with NIPS'19.

family of object detection networks designed around efficiency is the YOLO family of neural network architectures [12, 13, 14], which leverage a number of design principles to create single-shot architectures which can achieve embedded object detection performance on high-end desktop GPUs. However, these network architectures remain too large for many edge and mobile scenarios (e.g., $\sim$240MB in the case of the YOLOv3 architecture), and their inference speeds drop considerably when running on edge and mobile processors due to computational complexity (e.g., $>$65B operations in the case of YOLOv3). To address this issue, Redmon et al. introduced the Tiny YOLO family of network architectures, which has greatly reduced model sizes at a cost of object detection performance.

In this study, we are motivated to explore a human-machine collaborative design strategy to designing highly compact deep convolutional neural networks for the task of object detection, where principled network design prototyping is coupled with machine-driven design exploration. More specifically, we leverage the design principles from the YOLO family of single-shot object detection network architectures within this human-machine collaborative design strategy to create YOLO Nano, a highly compact network with highly customized module-level macroarchitecture and microarchitecture designs tailored for the task of embedded object detection.

## 2 Methods

In this study, we introduce YOLO Nano, a highly compact deep convolutional neural network for embedded object detection designed using a human-machine collaborative design strategy [21]. The human-machine collaborative design strategy for designing YOLO Nano comprises of two main design stages: i) principled network design prototyping, and ii) machine-driven design exploration.

### 2.1 Principled network design prototyping

The first design stage in creating YOLO Nano is a principled network design prototyping stage, where we create an initial network design prototype (denoted as $\varphi$), based on human-driven design principles to guide the machine-driven design exploration stage. More specifically, we construct an initial network design prototype based on the design principles of the YOLO family of single-shot architecture [12, 13, 14]. A standout characteristic of the YOLO family of network architectures is that, unlike region proposal-based networks which rely on the construction of a regional proposal network to generate proposals for where objects lie in the scene followed by classification on the generated proposals, they instead leverage a single network architecture to process the input image and generate the output results. As such, all object detection predictions for a single image are made in a single forward pass, compared to hundreds to thousands of passes that need to be performed to get the final results for region proposal-based networks. This makes the YOLO family of network architectures significantly faster to run, and thus better suited for embedded object detection.

The initial design prototype used in this study draws inspiration from the YOLO family of network architectures and is comprised of a stack of feature representation modules, with shortcut connections between the modules as with [14]. Also, as with [14], the feature representation modules are configured in a way, similar to feature pyramid networks [10], such that it is capable of representing features at three different scales. These feature representation modules are followed by several convolutional layers, with output being a three-dimensional tensor that encodes bounding box, objectness, and class predictions for three different scales. As a result, this initial design prototype architecture design allows for efficient multi-scale object detection.

The actual macroarchitecture and microarchitecture designs of the individual modules and layers in the final YOLO Nano network architecture, as well as the number of network modules, are left for the machine-driven design exploration stage to determine automatically given data as well as human-specified design requirements and constraints designed specifically around edge and mobile scenarios with limited computational and memory capabilities.

### 2.2 Machine-driven design exploration

Using the initial network design prototype ($\varphi$), data, as well as human-specified design requirements catered to edge and mobile usage as a guide, a machine-driven design exploration stage is then leveraged to determine the module-level macroarchitecture and microarchitecture designs for the proposed YOLO Nano network architecture. More specifically, machine-driven design exploration is achieved in this study in the form of generative synthesis [22], which is capable of determining the optimal macroarchitecture and microarchitecture designs of the final network architecture within the human-specified requirements and constraints. The overall goal of generative synthesis is to learn generative machines that can generate deep neural networks that meet design requirements and constraints, and can be described as follows. This is formulated within the concept of generative

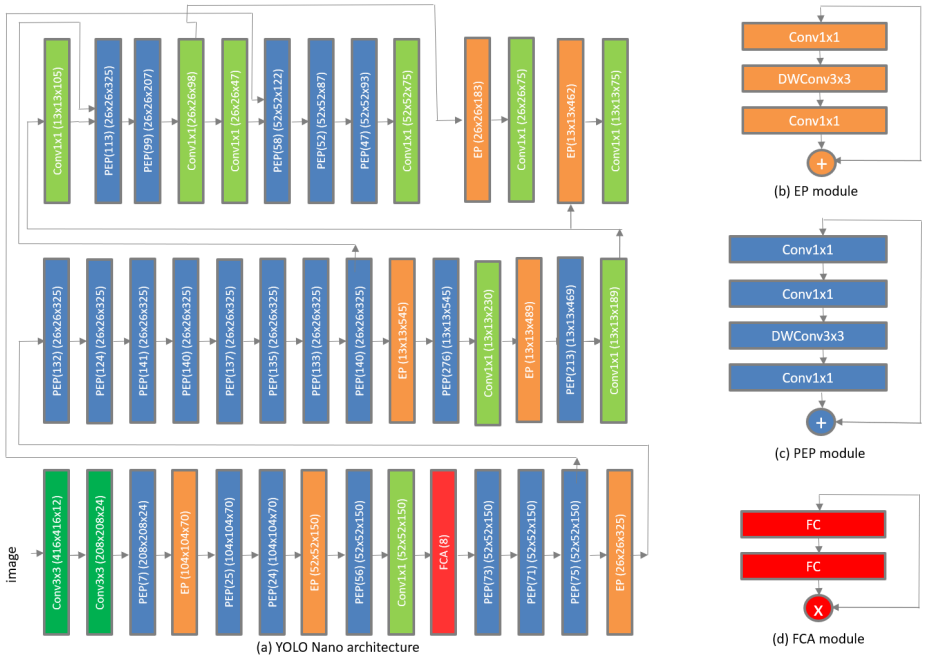


Figure 1: YOLO Nano network architecture. Note that PEP($x$) indicates $x$ channels in the first projection layer of a residual PEP module, and FCA($x$) indicates reduction ratio of $x$

synthesis as a constrained optimization problem for determining a generator $\mathcal{G}$ that, given a set of seeds $S$, can generate networks $\{N_s | s \in S\}$ maximizing a universal performance function $\mathcal{U}$ (e.g., [20]) while satisfying requirements and constraints defined via an indicator function $1_r(\cdot)$:

$$\mathcal{G} = \max_{\mathcal{G}} \mathcal{U}(\mathcal{G}(s)) \text{ subject to } 1_r(\mathcal{G}(s)) = 1, \ \forall s \in S. \tag{1}$$

Since it is computationally intractable to solve for the globally optimal solution in the constrained optimization problem posed in Eq. 1 given the enormity of the feasible region, we instead solve for an approximate solution $\hat{\mathcal{G}}$ via iterative optimization, where the initial solution $\hat{\mathcal{G}}_0$ is guided by $\varphi, \mathcal{U}$, and $1_r(\cdot)$, and progressively updated such that each successive approximate solution $\hat{\mathcal{G}}_k$ achieving a higher $\mathcal{U}$ than previous approximate solutions (i.e., $\hat{\mathcal{G}}_1, \ldots, \hat{\mathcal{G}}_{k-1}$, etc.) while still constrained by $1_r(\cdot)$. The final approximate solution $\hat{\mathcal{G}}$ is then used to create the proposed YOLO Nano network.

To guide the generative synthesis process towards learning generative machines that generate object detection networks for edge and mobile scenarios that are not only highly efficient and compact but also provide strong object detection performance, one of the key steps is to configure the indicator function $1_r(\cdot)$ to enforce the appropriate design requirements and constraints. In this study, the indicator function $1_r(\cdot)$ was set up such that: i) mean average precision (mAP) $\geq$ 65% on VOC 2007, ii) computational cost $\leq$ 5B operations, and iii) 8-bit weight precision. The computational cost constraint is set such that the computational cost of the resulting YOLO Nano network is below that of Tiny YOLOv3 [14], one of the most popular compact networks for embedded object detection.

## 3 YOLO Nano Architectural Design

The network architecture of the proposed YOLO Nano network for embedded object detection is shown in Figure 1, with several interesting observations worth discussing below.

### 3.1 Residual Projection-Expansion-Projection Macroarchitecture

The first notable observation about the YOLO Nano network architecture that differs significantly from the YOLO family of networks is that it is comprised of modules with unique residual projection-expansion-projection (PEP) macroarchitectures, in addition to expansion-projection (EP) macroarchitectures like those found in [16, 19, 1]. The residual PEP macroarchitecture consists of: i) a projection layer with $1\times1$ convolutions that projects output channels into an output tensor with lower dimensionality ii) an expansion layer with $1\times1$ convolutions, that expands the number of channels to a higher dimensionality, iii) a depth-wise convolution layer that performs spatial convolutions with a different filter on each of the the individual output channels from the expansion layer, and iv) a projection layer with $1\times1$ convolutions that projects output channels into an output tensor with lower

dimensionality. The use of residual PEP macroarchitectures enables significant reductions in the architectural and computational complexity while preserving model expressiveness.

### 3.2 Fully-connected Attention Macroarchitecture

The second notable observation about the YOLO Nano network architecture is the strategic introduction of light-weight fully-connected attention (FCA) within the network by the machine-driven design exploration process, which is in contrast to fixed module-level introduction in other design exploration methods [19]. As with [5], the FCA macroarchitecture consists of two fully-connected layers that learn the dynamic, non-linear inter-dependencies between channels and produces modulation weights for re-weight the channels via channel-wise multiplication. The use of FCA facilitates for dynamic feature recalibration based on global information to pay more attention to informative features, thus enabling better utilization of available network capacity. This in turn allows for a strong balance between reduced architectural and computational complexity and model expressiveness.

### 3.3 Macroarchitecture and Microarchitecture Heterogeneity

The third notable observation about the YOLO Nano network architecture is that there is high heterogeneity in terms of not only macroarchitectures (a diverse mix of PEP modules, EP modules, FCA, as well as individual $3\times3$ and $1\times1$ convolution layers), but also in terms of the microarchitectures of the individual feature representation modules and layers, with each module or layer in the network having unique microarchitectures. The benefit of having high microarchitecture heterogeneity in the YOLO Nano network architecture is that it enables each component of the network architecture to be uniquely tailored to achieve a very strong balance between architectural and computational complexity and model expressiveness. This architectural diversity in YOLO Nano also demonstrates the advantage of leveraging a machine-driven design exploration strategy as flexible as generative synthesis as it would be impossible for a human designer, or other design exploration methods such as [19, 1] to customize a network architecture to this level of architectural granularity.

## 4 Experimental Results and Discussion

To study the efficacy of YOLO Nano for embedded object detection, we examine its model size, object detection accuracy, and computational cost on the PASCAL VOC datasets. For comparison purposes, the Tiny YOLOv2 network [13] and the Tiny YOLOv3 network [14] were used as a baseline references given that they are amongst the most popular compact deep neural networks for embedded object detection given their small model sizes and low computational complexities. The VOC2007/2012 datasets consist of natural images that have been annotated with 20 different types of objects. The deep neural networks were trained using the VOC2007/2012 training datasets, and the mean average precision (mAP) was computed on the VOC2007 test dataset to evaluate the object detection accuracy of the deep neural networks, as is standard practice in research literature.

Table 1 shows the model sizes and the object detection accuracies of the proposed YOLO Nano network as well as Tiny YOLOv2 and Tiny YOLOv3. First, it was observed that the model size of YOLO Nano was 4.0MB, which is $>15.1\times$ and $>8.3\times$ smaller than Tiny YOLOv2 and Tiny YOLOv3, respectively, which is very important for edge and mobile scenarios given the memory constraints. Second, YOLO Nano, despite being much smaller in model size, achieved an mAP of 69.1% on the VOC 2007 test dataset, which is $\sim$12% and $\sim$10.7% higher than that of Tiny YOLOv2 and Tiny YOLOv3, respectively. Third, YOLO Nano requires just 4.57 billion operations to perform inference, which is >34% lower than Tiny YOLOv2 and $\sim$17% lower than Tiny YOLOv3.

Table 1: Object detection accuracy results of tested compact networks on VOC 2007 test set. Input size is $416\times416$ for all tested networks. Best results are highlighted in **bold**.

| Model Name | Model size | mAP (VOC 2007) | computational cost (ops) |
|---|---|---|---|
| Tiny YOLOv2 [13] | 60.5MB | 57.1% | 6.97B |
| Tiny YOLOv3 [14] | 33.4MB | 58.4% | 5.52B |
| YOLO Nano | **4.0MB** | **69.1%** | **4.57B** |

Finally, to investigate the real-world performance of YOLO Nano within an embedded scenario, we evaluated the inference speed and power efficiency of YOLO Nano running on a Jetson AGX Xavier embedded module at different power budgets. At 15W and 30W power budgets, YOLO Nano achieved inference speeds of $\sim$26.9 FPS and $\sim$48.2 FPS, respectively, resulting in power efficiencies of $\sim$1.97 images/sec/watt and $\sim$1.61 images/sec/watt, respectively. These experimental results show that the proposed YOLO Nano network, created through a human-machine collaborative design strategy, provides a strong balance between accuracy, size, and computational complexity that makes it well suited for embedded object detection for edge and mobile scenarios.

## References


[1] X. Chu, B. Zhang, R. Xu, and J. Li. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. *arXiv preprint arXiv:1907.01845*, 2019.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[3] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask r-cnn. *ICCV*, 2017.

[4] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[5] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu. Squeeze-and-excitation networks. *IEEE TPAMI*, 2019.

[6] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *IEEE CVPR*, 2017.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[8] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 2015.

[9] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[10] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[13] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *arXiv preprint*, 1612, 2016.

[14] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[15] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[16] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[17] M. J. Shafiee, B. Chywl, F. Li, and A. Wong. Fast YOLO: A fast you only look once system for real-time embedded object detection in video. *arXiv preprint arXiv:1709.05943*, 2017.

[18] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016.

[19] M. Tan, B. Chen, R. Pang, V. Vasudevan, and Q. V. Le. Mnasnet: Platform-aware neural architecture search for mobile. *arXiv preprint arXiv:1807.11626*, 2018.

[20] A. Wong. Netscore: Towards universal metrics for large-scale performance analysis of deep neural networks for practical usage. *arXiv preprint arXiv:1806.05512*, 2018.

[21] A. Wong, Z. Q. Lin, and B. Chwyl. Attonets: Compact and efficient deep neural networks for the edge via human-machine collaborative design. *arXiv preprint arXiv:1903.07209*, 2019.

[22] A. Wong, M. J. Shafiee, B. Chwyl, and F. Li. Ferminets: Learning generative machines to generate efficient neural networks via generative synthesis. *Advances in neural information processing systems Workshops*, 2018.

[23] B. Wu, F. Iandola, P. H. Jin, and K. Keutzer. Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. *arXiv preprint arXiv:1612.01051*, 2016.