Y-MAP-Net: Learning from Foundation Models for Real-Time, Multi-Task Scene Perception

Ammar Qammaz
Institute of Computer Science, FORTH and
Computer Science Department,
University of Crete
ammarkov@ics.forth.gr

Nikolaos Vasilikopoulos
Institute of Computer Science, FORTH and
Computer Science Department,
University of Crete
nvasilik@ics.forth.gr

Iason Oikonomidis
Institute of Computer Science, FORTH
oikonom@ics.forth.gr

Antonis A. Argyros
Computer Science Department, University of
Crete and Institute of Computer Science, FORTH
argyros@ics.forth.gr

***Abstract*— We present Y-MAP-Net, a Y-shaped neural network architecture designed for real-time multi-task learning on RGB images. Y-MAP-Net simultaneously predicts depth, surface normals, human pose, semantic segmentation, and generates multi-label captions in a single forward pass. To achieve this, we adopt a multi-teacher, single-student training paradigm, where task-specific foundation models supervise the learning of the network, allowing it to distill their capabilities into a unified real-time inference architecture. Y-MAP-Net exhibits strong generalization, architectural simplicity, and computational efficiency, making it well-suited for resource-constrained robotic platforms. By providing rich 3D, semantic, and contextual scene understanding from low-cost RGB cameras, Y-MAP-Net supports key robotic capabilities such as object manipulation and human–robot interaction. To encourage future research and reproducibility, we make our code publicly available [1].**

I. Introduction

Decades of research in computer vision have yielded powerful methods that solve long-standing perception challenges. Recent *foundation models* stand out for their generalization ability, achieved through immense scale and complexity. Trained on vast datasets with billions of parameters, they demonstrate zero-shot capabilities across diverse tasks and robust performance on in-the-wild data. However, their size comes at a cost: training requires PetaFLOP-scale resources, and their inference speed and memory footprint make them impractical for real-time deployment—particularly in robotics, where computational resources are often limited.

In contrast, compact convolutional architectures such as the You Only Look Once (YOLO) [2] family demonstrate how efficient models can achieve wide adoption in real-world robotic and embedded applications. This motivates the development of methods that combine the broad perceptual skills of foundation models with the efficiency of lightweight architectures, enabling real-time, multifaceted scene understanding for robotics. Such building blocks can provide robots with unified RGB-based perception, supporting tasks such as safe human–robot collaboration, dexterous manipulation, and navigation in dynamic environments.

Given this motivation, we propose Y-MAP-Net, a **Y**-shaped **M**ulti **A**ttribute **P**rediction neural **Net**work (NN) model that can be used as a closed-loop processor for RGB streams supplying human pose, depth, surface normals, image segmentation and a multiclass caption for each frame (Fig. 6). To achieve this, Y-MAP-Net adopts a multi-teacher, single-student training paradigm [3], [4], learning from task-specific foundation models supervising its predictions. This approach distills large models into a compact architecture, enabling learning without sacrificing robust generalization.

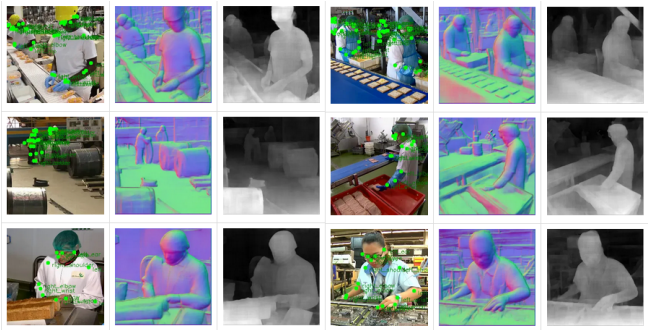

Fig. 1. Y-MAP-Net estimates human pose, depth, surface normals, segmentation and captioning from RGB in real-time. The image depicts sample pose keypoints, depth and normal estimations on publicly available youtube videos from factory floors in human-robot collaboration scenarios.

Y-MAP-Net accepts RGB input and produces 44 heatmap/image outputs and 8 caption tokens in real time. In summary, our contributions are:

- To the best of our knowledge, Y-MAP-Net is the first method to achieve end-to-end simultaneous depth, normal, and human pose estimation while also providing segmentation and captioning from monocular RGB in a single convolutional network operating in real time.
- We introduce a novel Y-shaped topology that offers a simple and pragmatic framework for multitask learning.
- Due to its efficiency on commodity hardware, Y-MAP-Net enables practical real-time deployment on resource-constrained robotic platforms in real-world scenarios.

II. Related Work

Y-MAP-Net addresses multiple computer vision subfields.

Image Classification and Captioning: AlexNET [5] was the first method to successfully deploy a large scale convolutional neural network, performing 1000-way softmax classification. DenseCap [6] moved away from labels to dense text captioning while in the Natural Language Processing (NLP) field Word2Vec [7], Fasttext [8] and GloVe [9] embeddings encoded text in a multi-dimensional vector representation. Transformers [10], a technique built from the ground up to capture textual conceptual relations, replaced other attempts based on LSTM [11] or GRUs [12]. ConvCAP [13] was the last major work on dense, purely convolutional captioning. Vision transformers (ViTs) [14] inspired the development of techniques such as CLIP [15] and DINO [16] associating image and text embeddings, paving the way for VLM foundation models like GPT4, [17] LLAMA [18] and DeepSeek VL2 [19] that successfully tackle the problem at its core.

Scene Segmentation: An alternative approach to scene captioning is regional scene annotation. YOLO [2] served as a seminal work for image region classification. Mask R-CNN [20] advanced dense segmentation and per-pixel label classification, inspiring subsequent works like Mask YOLO [21]—fusing earlier approaches [2], [20]—as well as Detectron 2 [22], DPText [23], and Segment Anything [24].

Pose Estimation: AlexNET [5] directly inspired Human Pose Estimation (HPE) methods [25] that adapted it for joint heatmap regression. OpenPose [26] was a landmark work that introduced part affinity fields (PAFs) [26]. During the same year, DensePose [27] explored an alternative formulation of the problem by regressing UV-mapped coordinates to a 3D model, thus providing a much denser result than 3D joint positions. HRNET [28] modified U-NETs [29] by featuring a pyramid of different input resolutions and achieved new levels of precision. The advent of ViTs [14] shifted focus to transformer-based architectures [30], with the Sapiens [31] foundation model being the first to achieve 2D joint, depth and normal estimation, as well as part segmentation, but focusing only on humans.

Depth and Normal Estimation: MiDaS [32] was a major push toward monocular depth estimation showcasing the power of relative depth. Metric3D, Marigold and ZoeDepth [33] further refined accuracy, while Dust3R [34], UniDepth [35] and DepthAnything (DA) [36] scaled both in model size and training data, leading to a mature framework that we use as a teacher for depth distillation training.

Training Datasets: Data sources such as Instagram-3.5B, JFT-300M, Visual Genome, and LAION-5B take advantage of the effectiveness of big data in deep learning, providing enormous training corpora for image captioning. For HPE, Humans-300M [31], Panoptic [37] and AMASS [38] are the largest datasets. Common Objects in Context (COCO) [39] remains a balanced middle ground for captioning, scene segmentation, and HPE. It has sufficient size and scope to facilitate reproducible research without being too encumbering in terms of the training resource amount it consumes.

Our approach: Y-MAP-Net stands on the "shoulders of giants" of research. At its core, our NN topology has similarities to U-NETs [29], however, being pushed to new limits in terms of concurrent tasks. Although the letter Y has been used to describe architectures that feature two output branches for specialized medical applications in the past [40], Y-MAP-Net is novel as it is, to the best of our knowledge, the first Y-shaped architecture to provide 44 heatmap outputs, as well as caption output for real-time inference. In contrast to other captioning works such as ConvCAP [13], it does not iteratively query the network for each captioning token. Additionally, it simultaneously regresses 2D joint heatmaps and PAFs, inspired by OpenPose [26] and predicts depth and surface normals supervised by DepthAnything V2 [41], with conceptual parallels to DensePose [27]. In addition, supervised by Detectron 2 [22] and DP-Text [23], Y-MAP-Net is equipped with per pixel object and scene segmentation capabilities. A multimodal method similar to ours is the Sapiens [31] foundation model. However, unlike Y-MAP-Net, Sapiens focuses solely on the human body, not producing results for the entire full-scene input image. Contrasting Sapiens 500M training corpus, our approach relies on just 0.1M samples. Architecturally, Sapiens is transformer-based, whereas we adopt a convolutional approach. Sapiens' separate encoder models for each modality range from 0.3B to 2.0B parameters. Our monolithic model is $4\times$ smaller than the aggregate size of the minimum Sapiens setup to tackle all tasks. In our model, weights are fully shared between tasks, in contrast to the sequential approach adopted in Sapiens. Finally, we include captioning on the same model, in contrast to [31] which lacks this functionality. Due to these differences in scope and design, the two approaches are not directly comparable. Even restricting the evaluation to human regions would not yield a meaningful comparison.

III. Model Architecture of Y-MAP-Net

Overview: Y-MAP-Net has a single RGB input or **encoder** branch, marked with green in Fig. 2. Outputs diverge in two dedicated paths. The first (magenta) is the pictorial **decoder** for depth, normals, heatmaps, PAFs, and segmentation masks. The second (blue) is related to **caption** output. The two paths are separated at the **bridge** section (orange in Fig. 2). This structure effectively combines a U-NET-like [29] architecture for pixel-level tasks with a densely connected token encoder ensemble for caption generation.

Encoder branch: This branch distills lower dimensional feature embeddings until arriving at the bridge. This is achieved through a series of 3×3 kernel convolutions followed by spatial dropout & avg. pooling as seen in Fig. 3.

Bridge segment: Similar to U-NETs [29], our bridge consists of 3 densely connected layers reconciling multitoken GloVe [9] embeddings and multifaceted pictorial output.

Pictorial decoder branch: This part of the network produces 44 image channels that contain pixel-level output. We observed that adding a 1×1 convolution layer with a very high number of filters prior to the output layer increases the fidelity of these images. We experiment with 512-1750

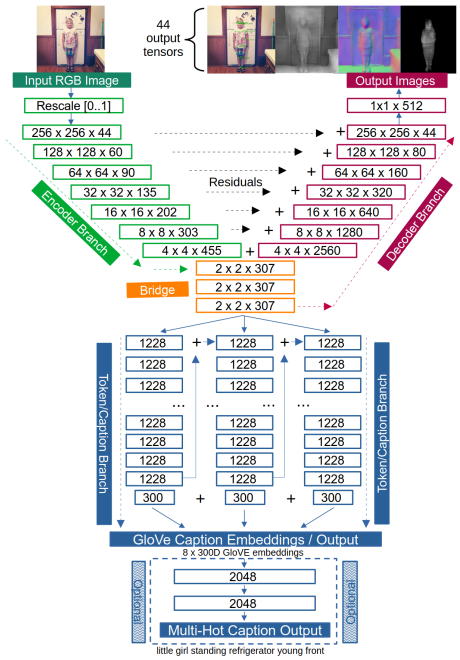

Fig. 2. Architecture of the proposed Y-MAP-Net. The flow of data from input to outputs is highlighted with arrows. Residual connections are indicated with + signs, while rectangles give a dimensionality overview for each layer. Green layers (top left) signify encoder blocks originating from 1 RGB image. The network bridge layers appear in orange color (middle). Blue layers (bottom) depict 8 captioning output token GloVe [9] vectors. These can be optionally converted to multi-hot labels by adding 2 dense layers for some applications. Magenta (top right) signifies decoder blocks that lead to 44 multi-modal pictorial outputs. The detailed encoder and decoder architecture is provided in Figure 3.

element layers (Table I), a decision affected by available H/W resources (NVIDIA RTX A6000 GPGPU with $\approx$ 49GB VRAM). We attribute the benefits of this "pixel-wise" layer to increased capacity for a richer set of activated features due to powerful gradients present directly before the final layer.

Both the pictorial output and GloVe token embeddings use hyperbolic tangent ($tanh$) output layer activations. This choice forces output in the range of [-1..1], effectively controlling gradient flow. The remaining layers utilize Leaky ReLU [42] activations that allow for a small gradient when inactive to mitigate vanishing gradients. Toward the same end, GloVe vectors from captioning output are also normalized to [-1..1], while the first layer after RGB input rescales the RGB values to [0..1]. Residual connections directly connect encoder blocks to their respective same-dimensional decoder counterparts (black horizontal arrows in Fig 2).

Token/caption output branch: The token/caption output branch departs from previous designs. In the past, captioning networks used GRUs [12] featuring recurrent connections. ConvCAP [13] maintained a purely convolutional recurrent approach expecting previously regressed tokens as input to iteratively produce the next ones until full sentence

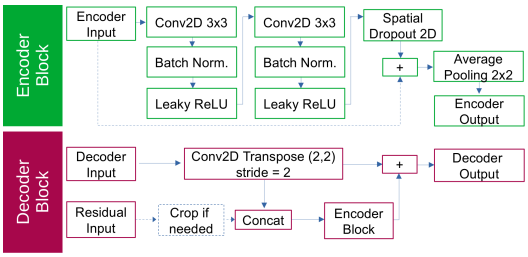

Fig. 3. Encoder (green) and decoder (magenta) blocks of Y-MAP-Net (Fig. 2) are tasked with down-scaling input images to the bridge representation and then up-scaling to pictorial outputs. They consist of layers and skip connections shown in this figure. Full Tensorflow source code available at [1].

formation. This would be a prohibitive design choice for Y-MAP-Net because it would greatly slow down the processing framerate. It should be noted that although ConvCAP experimented with an early pairwise mechanism similar to transformer attention, this proved to have a small positive impact on the core ConvCAP architecture [13]. Similarly, modern transformer architectures rely on calculating attention scores for all possible output pairs, increasing network complexity. Architectures like differential transformers [43] are now specifically built to tackle this problem by canceling out attention scores on "noisy"/unimportant token pairs, while research is being conducted on the attention heads themselves [44] and the roles they are called to accomplish. In contrast to all this complexity, we follow a more straightforward approach by eliminating the so called stop-word [45] tokens and using GloVe [9] as a continuous token vector space. This helps with ordering the various responses of the network while training it. Lastly, residual connections from each token output to the next are incorporated to form encoding chains between tokens that build on previous tokens.

The captioning branch (Fig. 2) features Layer instead of Batch normalization between layers. 20% dropout is applied after each token layer and is reduced conversely as we go through to the last token. Residual connections from one token to the next are scaled to 30% of their initial magnitude to tackle the network that produces the same token multiple times. Finally, output dimensions can vary depending on the task and computational resources. In general, Y-MAP-Net, produces 44 heatmaps and 8 tokens: Heatmaps #1 to #17 encode 2D joints, #18-#29 PAFs [26], heatmap #30 depth, #31-#33 normals and #34-#44 encode class segmentation.

2D Human Pose Estimation (HPE): HPE is formulated as a series of transformations of an input RGB image to heatmaps activated only in the vicinity of a specific human body joint. Typical skeletons like the one used in COCO include 17 joints that contain the eyes, ears, shoulders, elbows, hands, hips, knees and ankles. To limit the already very broad scope of Y-MAP-Net , we opted to not focus on more than these 17 joints. 2D keypoints are very useful for detecting the presence of humans. However, connecting the dots of the

joints to form complete skeletons is equally important. The output of a 2D HPE module should not only be 2D joints but a list of humans with each joint associated to a specific skeleton. This is achieved by PAFs [26], which are heatmaps active between neighboring joints and zero everywhere else, indicating connected joint pairs. Therefore, we include joint PAF regression using the outputs #18 to #29.

Depth and Normals Estimation: COCO [39], much like all non-synthetic, in-the-wild datasets, does not contain depth data for training. We selected Depth Anything V2 (DAv2) [41] as a teacher model to generate them. DAv2 offers state of the art accuracy, and with it we can produce a depthmap for each RGB training sample of COCO. Although DAv2 produces FP32 depth (similar to Y-MAP-Net), its implementation's .png output is encoded as 8-bit values. We therefore modify it to encode depth as 16-bit. DAv2 features 2 families of models, extracting metric and normalized range depth results. It also features multiple model sizes. We used the largest available (vitl 335M) and use normalized depth ranges since they produce rich details and are preferred in the literature for visual fidelity [32].

Given a depth image D_I for each training sample, we compute the partial gradients $\frac{\partial D_I}{\partial x}$ and $\frac{\partial D_I}{\partial y}$ applying Sobel filters along the x and y directions, respectively. After performing gradient calculations, we compute the norm of the vectors, also adding a small value ϵ to avoid division by zero. We then compute the normal images n_x, n_y and n_z. At first glance, including normal data as an output of our NN seems counterintuitive, since a network regressing depth already encodes normals in the depthmap. In practice, we observe that the normal output has better fidelity compared to the depth output. This is attributed to neighboring data that have a similar appearance in RGB maintaining similar n_x, n_y and n_z in their local neighborhoods, effectively reducing noise. We therefore leverage this behavior, using the output normals to refine the output depth as seen in Fig. 4.

Iterative depth refinement: Assuming an output depth image D_O and normal maps N_x, N_y and N_z, we compute gradients in x and y directions by locally sampling depth: $G_x = \partial D_O/\partial x, \quad G_y = \partial D_O/\partial y$. Normalizing gradients with their magnitude M we obtain the normalized maps G_x^{nor}, G_y^{nor}, G_z^{nor} and calculate the differences in the target normals and gradients of the normalized depth map to find inconsistencies: $\Delta_x = N_x - G_x^{\text{nor}}, \quad \Delta_y = N_y - G_y^{\text{nor}}, \quad \Delta_z = N_z - G_z^{\text{nor}}$. We finally update the improved output depth map D_O minimizing the normal difference: $D_O = D_O + \alpha\left(\Delta_x \cdot G_x + \Delta_y \cdot G_y + \Delta_z\right)$. This is executed for 35 iterations with $\alpha = 0.01$. We mask far areas to skip calculations and respect our real-time target. As seen in Fig. 4, this process produces noticeable improvements, sharpening resulting depthmaps and suppressing output noise.

Image Segmentation: COCO 17 contains 183 segmentation label categories that depict vehicles, animals, objects, and other elements commonly visible in digital photographs. To perform per-pixel segmentation, we need to arrange the data in a way where each pixel has an associated output label.

Fig. 4. NN output normals (left) enforced on NN depth output (mid) through our algorithm improves depth (right).

Encoding all possible discrete values into a single heatmap proved infeasible in practice for training, since the neural network predictably lacks the capacity to produce such finely detailed activations at the segmentation output layer. We thus decouple various labels as different heatmaps, similarly to the way 2D joints are treated to make the task feasible.

However, using 183 separate 256×256 heatmaps requires broadcasting $\approx$ 12 million elements per training sample, leading to prohibitively long training times. We attempted to encode all classes in one image yielding only 65K for transit to the GPGPU per training sample, and perform one hot encoding inside the loss function. Although this restores training times, with some expressivity loss, the capacity of the output layer does not allow this to work in practice. To overcome this problem, we group segmentation classes in 11 broad categories, namely Persons, Vehicles, Animals, Objects, Furniture, Appliances, Materials, Obstacles, Buildings, Nature and Text. Each label is assigned to its respective group, yielding 11 segmentation images with the union of all visible classes. This balanced divide & conquer approach helps the network tackle a large number of labels successfully while minimizing training overhead.

Multi-Label Captioning: Captioning an image can be posed as selecting words that describe it and putting them in a sentence. However, there is a severe class imbalance among different tokens/labels. Works such as [46] highlight the intricate differences between multi-label, multi-instance, and regular captioning, motivating us to follow the former. To improve data distribution and conserve NN capacity, we remove stop-words. After limiting tokens to the 2048 most popular ones, initially, we naively attempted to perform multi-hot encoding for 8 output tokens, yielding 8×2048 captioning outputs for each sample. However, this approach did not work since token order is very important and the task proved too difficult for a lightweight convolutional approach. Performing the union over these tokens and emitting a multi-hot 2048 class vector with 8 active labels per sample showed promise. This approach eliminated token ordering, significantly simplifying the task. Despite performing class weighting based on token frequencies, the model tended to only produce results for a subset of tokens and frequently made errors. This behavior is understandable: tokens like man, boy, woman, girl, person, and human have similar representations in images. However, despite their conceptual similarity, misclassifying a person as a man versus a cat incurs the same loss penalty, leading to problematic training.

To improve on this, we employed Global Vectors

(GloVe) [9] for word representation. GloVe offers different publicly available pre-trained databases. We select GloVe 6B which is uncased and trained on Wikipedia 2014 and Gigaword 5, with 6B tokens, a 400K vocabulary, and offers 50, 100, 200 or 300 dimensional embeddings for each word. Employing GloVe, we lose the ability to perform a union (or sum) of words as a combined output. However, attempting to predict 8×300 captioning outputs, this time using an MSE loss, showed signs of promise even from very early experiments. During runtime, to transform the GloVe 300D output back to words, we follow three steps. First, we threshold the 300D GloVe vector norm against zero, filtering it out when close to zero. We then select the word token with the highest cosine similarity to our output. As a final step, we can perform a second threshold operation filtering out tokens without a high cosine similarity to any word / vector.

We observed artifacts at this stage including multiple outputs that regressed to similar embedding values. To support the formation of diverse token sequences, we introduced residual connections from each token to the next. We initially observed that this further increased token repetition. However, using tanh output activations, normalizing GloVe embeddings to [-1..1], employing dropout and layer instead of batch normalization, and scaling down the magnitude of the residual connections to 30% mitigated this. As reported by methods such as ConvCAP [13], GloVe embeddings can be used as intermediate output supervision. Adding a final two-layer densely connected tail (bottom of Fig. 2) with sigmoid or softmax activations yields easy-to-use output. This eliminates the need for extra computations, since directly thresholding the returned pseudo-probabilities is sufficient.

Data Augmentation: COCO 17 contains 118K training samples. To enhance Y-MAP-Net's generalization with this relatively limited amount of data and allow for a multi-epoch training regime without over-fitting, we employ aggressive data augmentation. The average sample dimension of COCO is close to VGA resolution. However, our input and output images have a 256×256 size. Each image augmentation has an independent activation chance thus allowing multiple augmentations on an image. A sample has 45% chance for pan & zoom. It has 50% chance to have brightness/contrast alterations, out of which, for 50% samples we perform uniform brightness/contrast changes across all channels, while the rest have individual R,G,B alterations with up to a $\pm 20\%$ change. There is a 15% chance of Gaussian noise perturbation, 50% chance of adding up to 10 burned pixels (1/6553 corruption in 256×256 images). If persons are present, at least 30% of their joints remain visible in the cropped image, while if a sample does not contain any persons, there is a 1% chance of completely replacing the image with random noise. This maximizes use of our limited human pose training data while conditioning the network to filter out erratic input. Finally, pan & zoom augmentation has a strict $1.1x$ zoom limit. Since captions describe the full image, aggressive zooming may crop out smaller or peripheral objects. Zoom is limited to prevent exclusion of captioned content, which can cause concept drift (e.g. surfboards being misassociated with the sea).

Exp. No.	Layers	M.Size	In.Dim.	Out.Dim.	Pix.wise	Res.con.	Pose/Depth	Normals	PAFs	Captions	Class Grp.	HDM train	HDM val.
1	4	46M	110	48	-	-	✓	-	-	-	-	0.69	0.56
2	4	31M	220	96	-	-	✓	-	-	-	-	0.70	0.71
3	4	39M	200	100	-	-	✓	-	-	-	-	0.70	0.69
4	5	48M	256	256	-	-	✓	-	-	-	-	0.70	0.70
5	5	54M	300	288	-	-	✓	-	-	-	-	0.82	**0.84**
6	6	70M	256	256	-	-	✓	-	-	-	-	0.66	0.67
7	7	231M	128	128	-	-	✓	-	-	-	-	0.42	0.52
8	7	136M	420	384	-	-	✓	✓	-	-	-	0.75	0.76
9	7	278M	420	384	-	-	✓	✓	-	-	19	0.74	0.77
10	8	401M	420	256	-	-	✓	✓	-	-	19	0.74	0.77
11	7	315M	256	256	-	-	✓	✓	✓	-	3	0.56	0.61
12	7	316M	256	256	1.6K	-	✓	✓	✓	-	3	0.62	0.67
13	7	172M	256	256	1.6K	✓	✓	✓	✓	-	3	0.66	0.72
14	7	213M	256	256	1.7K	✓	✓	✓	✓	-	2	0.53	0.59
15	7	298M	256	256	1.5K	✓	✓	✓	✓	✓	11	0.64	0.72
16	7	347M	256	256	0.5K	✓	✓	✓	✓	✓	11	0.76	**0.83**

TABLE I. We record enc/dec branch layer depth, model size (millions of weights), input and output dimensionality (images are rectangular so 256 is equivalent to 256x256), NN features such as pixelwise layer before output, residual connections in enc/dec blocks, output modalities enabled in an experiment (PAFs, Normals, Class segmentation) and the training/validation HDM score for each configuration. **Up:** Limiting output to joint+depth modalities we find that shallow networks perform better than more complex ones that over-fit. **Down:** As more modalities are introduced the problem becomes harder and tensor sizes need to be reduced to maintain real-time performance. The final row contains the Y-MAP-Net 1-8-44 configuration. Its weights are increased compared to previous experiments due to the captioning branch. Each experiment requires $\approx$ 1 week to run.

Method	M.Size	Joints	PAFs	Depth	Normals	ClassSeg.	Text Seg.	Captions
OpenPose [26]	28M	✓	✓	-	-	-	-	-
DAv2 [41]	335M	-	-	✓	✓	-	-	-
Detectron2 [22]	63M	-	-	-	-	✓	-	-
DPText [23]	44M	-	-	-	-	-	✓	-
VisionGPT2 [47]	239M	-	-	-	-	-	-	✓
Y-MAP-Net (Ours)	347M	✓	✓	✓	✓	✓	✓	✓

TABLE II. We tackle multiple tasks with a monolithic network using 48.94% of the weights typically required. We use [22], [23], [41], [47] as teachers during training.

Model Training: Training is performed using Keras 3.9.2 / TensorFlow 2.19.0. We use a batch size of 32 on an AMD Ryzen Threadripper PRO 3955WX CPU. Backpropagation runs on an NVIDIA RTX A6000 GPU. Model weights use FP32 precision, while images are stored as INT8 to improve CPU-GPU throughput. Weights are initialized with Glorot/Xavier initialization. We use the ADAM optimizer with a learning rate starting at 2.4×10^{-4} and linearly decaying to 1.0×10^{-4} after 100 epochs. Training runs for up to 250 epochs with early stopping patience of 65 epochs.

The heatmap target outputs also have a decay schedule with joints that have a Gaussian size of 23×23 during the start of training and then decay to a target limit of 8×8.

Modality	Teacher	HDM 0.1	HDM 0.3	HDM 0.5	HDM 0.8	MSE
Joints	COCO GT	0.98	0.98	0.99	0.99	1.34
PAFs [26]	COCO GT	0.97	0.97	0.97	0.98	1.09
Depth	DAv2 [41]	0.53	0.57	0.57	0.58	58.49
Normals	DAv2 [41]	0.50	0.60	0.63	0.68	54.59
Text Segm.	DPText [23]	0.98	0.98	0.98	0.98	0.67
Class Segm.	Detectron 2 [22]	0.97	0.97	0.97	0.98	1.62

TABLE III. Quantitative results for specific NN output modalities using MSE and HDM for various $\mathcal{T}$ thresholds (0.1 - 0.8). Our network scores competitively in a wide range of tasks with depth estimation proving the hardest task.

In a similar manner, PAFs start from a line width of 6 and decay down to 2. This reduction happens linearly, with the size being reduced by one step every 20 epochs. This helps the NN to identify joints early on during training, since their magnitude is comparable to other output modalities, but also ensures better localized joint responses for fully trained models. The loss function $\mathcal{L}$ we use is the following:

$$\mathcal{L} = w \cdot \sum_i g_i \cdot \frac{1}{N} \sum_{j=1}^{N} \left(y_{\text{true},\, i,j} - y_{\text{pred},\, i,j}\right)^2 .$$

In the above equation, i runs over 7 different terms. Each MSE term represents the mean squared error between the true and predicted values for the respective channels, weighted by its corresponding gain g_i. We experimentally determined g_i values that promote harder aspects of the problem by increasing their relative contribution to the total loss. These 7 values are: $g_G = 10.0$ for GloVe tokens (tok. 1-8), $g_J = 2.4$ for joint heatmaps (chan. 1-17), $g_{\text{PAF}} = 1.0$ for PAFs (chan. 18-29), $g_D = 1.0$ (for Depth chan. 30), $g_N = 1.0$ (for Normals chan. 31-33), $g_T = 1.0$ (for Text chan. 34) and $g_S = 2.0$ (for Segmentation chan. ≥ 35). We use cosine similarity to monitor GloVe tokens. For image output we define *Heatmap Distance Metric (HDM)* which measures the fraction of all predicted pixels within relative threshold $\mathcal{T}$ from their target value. Assuming that y_{true} and y_{pred} are true and predicted heatmaps in the [0..1] range, HDM is defined as the percentage of pixels for which: $|y_{\text{true}} - y_{\text{pred}}| \leq \mathcal{T}$. In experiments, $\mathcal{T}$ = 0.1 unless otherwise stated.

IV. Experiments

Ablation Study: We conduct a series of ablation experiments with different I/O sizes, different relative depths of the network, gradually adding output modalities and broadening the size and scope of our NN until accommodating pose, depth, normal, segmentation and captioning and reaching our target. These experiments are summarized in Table I. We control network size by altering the number of filters in the enc/dec branches (Fig. 2). For example, as seen in rows #6 and #7 of Table I we can have a bigger NN that has enc / dec depth of 6 layers and a 3× larger NN that handles half of the dimensionality of input and output in each dimension. As seen in experiment #4, a 5-layer enc/dec architecture that only tackles pose and depth has excellent performance. To achieve a similar HDM metric with #4 for all involved outputs, we need a substantially larger architecture with more capacity, such as exp. #13+. Residual connections and the 1×1 pixel-wise layer, add representational capacity allowing stronger networks that tackle more modalities. However, the width of the pixel-wise layer is constrained by GPU VRAM. Introducing a captioning branch does not increase the size of the image output branch, while caption features reaching the bridge layer (Fig. 2) appear to benefit the decoder. The upper part of Table I shows that when limiting output to joint + depth, shallow networks perform better than more complex ones that suffer from over-fitting. Runtime performance results are summarized in Figure 5.

Quantitative Results: Y-MAP-Net uses DAv2 [41], Detectron 2 [22], DP-Text [23] and VisionGPT2 [47] as teacher models in the multi-teacher, single-student setting we adopt. It is trained on just 118K COCO 17 samples, orders of magnitude fewer than the millions of samples training corpora of our teacher models. Furthermore, our model is less than half the size of the combined weights of its teachers (Table II) and works in real time. Last but not least, our method uses a single set of weights to simultaneously handle all output modalities, while each teacher is specialized for only one task. We therefore expect our method to produce "distilled" output of reduced yet comparable accuracy, compared to its teachers. Since no existing method integrates the range of functionalities we target, our primary comparisons are thus made against the individual teacher models.

We evaluated our model on the COCO 17 validation set, using ground truth data generated by our teacher models. Our quantitative metrics, summarized in Table III, reveal varying performance across different tasks. Depth estimation yielded the lowest accuracy, which is consistent with its inherent complexity. In contrast to joints, PAFs and class segmentations that have coarse "on/off" values and large inactive areas, depth maps return a response for all pixels. Furthermore, even minor inaccuracies in depth predictions result in significantly larger error contributions, particularly when depth is assigned to an incorrect distance. Estimating object depth in in-the-wild images presents additional challenges due to the absence of reference scale, multiple viewpoints, or motion cues, as discussed in [48]. Our depth teacher, DAv2 [41], is trained on more than 60M images, two orders of magnitude more than our training corpus, and has a larger size, fully dedicated to its single task as seen in Table II. Normals perform better, with improved fidelity in areas with edges. We take advantage of this with our iterative algorithm (Fig. 4). Class segmentation performs better and Joint/PAF estimation achieves the highest accuracy. We partly attribute this to the quality of human-annotated training data, since unsupervised, model generated labels introduce inaccuracies. For captioning, we use the cosine similarity metric, achieving similarity scores of 0.59, 0.62, 0.64, 0.64, 0.53, 0.27, 0.08, and 0.01 for caption tokens 1 through 8, respectively. Overall, we observe good generic scene descriptions.

Qualitative Results: We tested Y-MAP-Net on a variety of input sources. We observe strong generalization on images from the COCO 17 validation set (Fig. 6). Since we aim

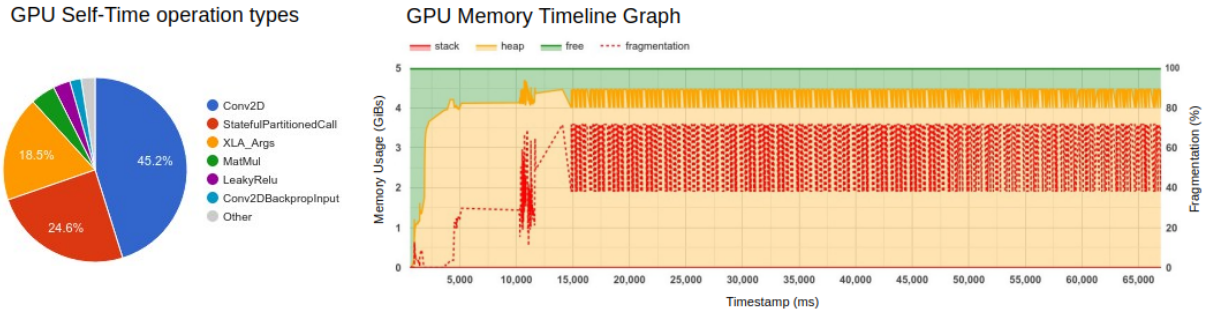

Fig. 5. We profile Y-MAP-Net using TensorBoard. **Left:** 45.2% of operations are Conv2D. On an RTX 5080, the FP32 model achieves 25.4Hz while using 21% of GPU compute capacity. **Right:** Memory profiling shows 4.71GB GPU VRAM usage, making it attractive for computer vision applications requiring multi-modal scene understanding from RGB. We only use 48.94% of our teachers' weights (Table II) to simultaneously reproduce their outputs with comparable accuracy.

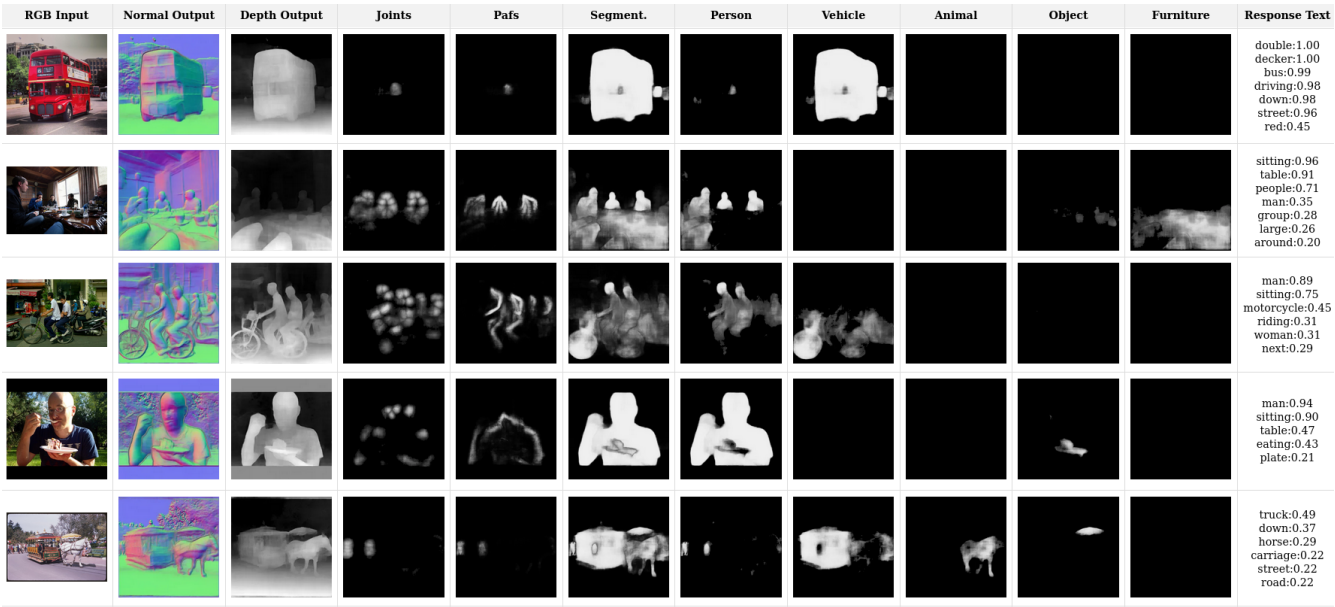

Fig. 6. Qualitative results on COCO 17 [39] validation with raw model outputs. Joints, PAFs and segmentation columns show the union of joint, PAF and class heatmaps. The "Response Text" column lists tokens with confidence $> 25\%$.

for real-time applications in production environments, we experiment with Youtube videos featuring workers in production lines (see Fig. 1). We observe normal and depth estimation consistent with input, pose estimation results that follow the bodies even when wearing protective equipment, multi-token captioning output that describes the major visible components of the RGB image and class segmentation that, when observing a few objects/subjects, is very sharp. For a more extensive collection of qualitative results, see the supplementary video material [49] accompanying this paper.

V. Conclusions

We introduced Y-MAP-Net, the first framework to jointly address depth, surface normal, human pose, semantic segmentation, and multi-label captioning in real time from monocular RGB. The proposed Y-shaped architecture is compact, versatile, and efficient, providing a unified alternative to task-specific networks and sparse architectures such as YOLO. By delivering rich multi-modal scene understanding at low computational cost, Y-MAP-Net is well suited for robotics, where onboard resources are limited but robust perception is critical for world navigation, manipulation, and human–robot interaction. Future directions include extending our multi-teacher paradigm with mixtures of experts for each modality, and scaling training to broader and robot-centric datasets captured from embodied platforms. Our code is released publicly [1] to support robotics perception research.

VI. Acknowledgments

This work was co-funded by the European Union (EU - HE Magician – Grant Agreement 101120731) and the Hellenic Foundation for Research and Innovation (HFRI) under the "1st Call for HFRI Research Projects to support Faculty members and Researchers and the procurement of high-cost research equipment", project I.C.Humans, no 91.

REFERENCES

[1] Ammar Qammaz and Antonis A Argyros, "Y-MAP-Net github repository," https://github.com/FORTH-ICS-CVRL-HCCV/Y-MAP-Net, 2026.

[2] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.

[3] Shan You, Chang Xu, Chao Xu, and Dacheng Tao, "Learning from multiple teacher networks," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 1285–1294.

[4] XB Bruce, Yan Liu, and Keith CC Chan, "Multimodal fusion via teacher-student network for indoor action recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," *NeurIPS*, 2012.

[6] Justin Johnson, Andrej Karpathy, and Li Fei-Fei, "Densecap: Fully convolutional localization networks for dense captioning," in *CVPR*, 2016.

[7] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," *NeurIPS*, 2013.

[8] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov, "Enriching word vectors with subword information," *Transactions of the association for computational linguistics*, 2017.

[9] Jeffrey Pennington, Richard Socher, and Christopher D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *NeurIPS*, 2017.

[11] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] Kyunghyun Cho, Bart Van Merriënboer, Çağlar Gulçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of EMNLP*, 2014.

[13] Jyoti Aneja, Aditya Deshpande, and Alexander G. Schwing, "Convolutional image captioning," in *CVPR*, 2018.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[15] Alec Radford et al., "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[16] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin, "Emerging properties in self-supervised vision transformers," in *ICCV*, 2021.

[17] OpenAI et al., "GPT-4 technical report," 2024.

[18] Abhimanyu Dubey et al., "The Llama 3 herd of models," 2024.

[19] Zhiyu et al. Wu, "Deepseek-VL2: Mixture-of-experts vision-language models for advanced multimodal understanding," *arXiv preprint arXiv:2412.10302*, 2024.

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask r-cnn," in *ICCV*, 2017.

[21] Jianing Sun, "Mask-YOLO," https://github.com/jianing-sun/Mask-YOLO, 2018.

[22] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[23] Maoyuan Ye, Jing Zhang, Shanshan Zhao, Juhua Liu, Bo Du, and Dacheng Tao, "DPText-DETR: Towards better scene text detection with dynamic points in transformer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, vol. 37, pp. 3241–3249.

[24] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al., "Segment anything," in *ICCV*, 2023.

[25] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," *NeurIPS*, 2014.

[26] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*, 2017.

[27] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos, "Densepose: Dense human pose estimation in the wild," in *CVPR*, 2018.

[28] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang, "Deep high-resolution representation learning for human pose estimation," in *CVPR*, 2019.

[29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.

[30] Yuanzheng Ci, Yizhou Wang, Meilin Chen, Shixiang Tang, Lei Bai, Feng Zhu, Rui Zhao, Fengwei Yu, Donglian Qi, and Wanli Ouyang, "Unihcp: A unified model for human-centric perceptions," in *CVPR*, 2023.

[31] Rawal Khirodkar, Timur Bagautdinov, Julieta Martinez, Su Zhaoen, Austin James, Peter Selednik, Stuart Anderson, and Shunsuke Saito, "Sapiens: Foundation for human vision models," in *ECCV*, 2025.

[32] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE transactions on PAMI*, vol. 44, no. 3, pp. 1623–1637, 2020.

[33] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller, "Zoedepth: Zero-shot transfer by combining relative and metric depth," *arXiv preprint arXiv:2302.12288*, 2023.

[34] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud, "Dust3r: Geometric 3d vision made easy," in *CVPR*, 2024.

[35] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu, "Unidepth: Universal monocular metric depth estimation," in *CVPR*, 2024.

[36] Lihe Yang et al., "Depth anything: Unleashing the power of large-scale unlabeled data," in *CVPR*, 2024.

[37] Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh, "Panoptic studio: A massively multiview system for social motion capture," in *ICCV*, 2015.

[38] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black, "AMASS: Archive of motion capture as surface shapes," in *ICCV*, 2019.

[39] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.

[40] Sachin Mehta, Ezgi Mercan, Jamen Bartlett, Donald Weaver, Joann G Elmore, and Linda Shapiro, "Y-net: joint segmentation and classification for diagnosis of breast biopsy images," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II 11*. Springer, 2018, pp. 893–901.

[41] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao, "Depth Anything V2," *NeurIPS*, 2024.

[42] Bing Xu et al., "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[43] Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei, "Differential transformer," *arXiv preprint arXiv:2410.05258*, 2024.

[44] Zifan Zheng, Yezhaohui Wang, Yuxin Huang, Shichao Song, Bo Tang, Feiyu Xiong, and Zhiyu Li, "Attention heads of large language models: A survey," *arXiv preprint arXiv:2409.03752*, 2024.

[45] Christopher Fox, "A stop list for general text," *SIGIR Forum*, vol. 24, no. 1–2, pp. 19–21, Sept. 1989.

[46] Wanting Ji and Ruili Wang, "A multi-instance multi-label dual learning approach for video captioning," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 17, no. 2s, June 2021.

[47] Shreyas Daniel Gaddam, "VisionGPT2," https://github.com/shreydan/VisionGPT2, 2023.

[48] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, Fisher Yu, Dacheng Tao, and Andreas Geiger, "Unifying flow, stereo and depth estimation," *IEEE Transactions on PAMI*, 2023.

[49] Ammar Qammaz and Antonis A Argyros, "Y-MAP-Net supplementary video," https://youtu.be/n6P_nXLWz1A, 2026.