Multi-label Annotation for Visual Multi-Task Learning Models

Gaurang Sharma
Tampere University
Tampere, Finland
gaurang.sharma@tuni.fi

Alexandre Angleraud
Tampere University
Tampere, Finland
alexandre.angleraud@tuni.fi

Roel Pieters
Tampere University
Tampere, Finland
roel.pieters@tuni.fi

Abstract—Deep learning requires large amounts of data, and a well-defined pipeline for labeling and augmentation. Current solutions support numerous computer vision tasks with dedicated annotation types and formats, such as bounding boxes, polygons, and key points. These annotations can be combined into a single data format to benefit approaches such as multi-task models. However, to our knowledge, no available labeling tool supports the export functionality for a combined benchmark format, and no augmentation library supports transformations for the combination of all. In this work, these functionalities are presented, with visual data annotation and augmentation to train a multi-task model (object detection, segmentation, and key point extraction). The tools are demonstrated in two robot perception use cases. For more details, please visit https://gaurangsharma18.github.io/website/MultiLabel/index.html.

Index Terms—Multi-task models, image annotation and augmentation, object detection and segmentation, keypoint detection

I. INTRODUCTION

Advancements in deep learning have helped to address numerous problems in different domains (e.g., robotics [1], medicine [2] and agriculture [3]). Perception in particular has been of major focus, where single purpose models have been optimized and fine-tuned for one specific task at hand [4]. More recently also multi-task models have been developed [5], where a shared model concurrently learns multiple tasks. A fundamental aspect of these models is to increase data efficiency, reduce overfitting, and enable quick learning due to the use of auxiliary information. They use a unified single backbone structure having multiple descriptive heads to address technical computer vision challenges like boundary detection, semantic segmentation, object detection, etc. To effectively utilize multi-task models, multi-label annotations should be supported by the framework. This implies that images and the objects inside them can be annotated with multiple different labels (e.g., bounding boxes, polygons and key points), to be used for learning different tasks and thus generate multiple outputs. Compared to single-task models this is less computationally expensive and requires less annotated records.

Fig. 1 depicts this visually in an object detection problem. Two objects are annotated and correctly detected, however,
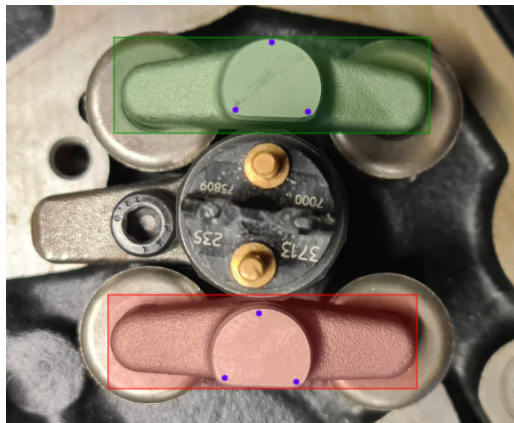
Fig. 1: Motivation for multi-task detection models. Single-task models can detect an object (bounding box), but cannot verify whether their orientation is correct. Multi-task models can provide additional information from different model outputs. In this case key points (three blue points) indicate a correct (green box) and incorrect (red box) object placement.

in this context, the lower object is incorrectly placed. From detection or segmentation alone, this cannot be determined, yet with the estimated key points as additional information (i.e., three blue points), incorrect placement can be detected. Existing labeling and annotation solutions [6] do not provide the combined annotation export functionality, which results in solutions that combine several export formats to accomplish a common task. This served as motivation for the development of a novel data pipeline to provide domain-specific annotation and augmentation strategies. Best-of-breed libraries such as Albumentation [7] and Label Studio [8] are integrated to create a generic data generation solution for training deep learning models. Our developments are evaluated by applications in object detection, object configuration estimation and depth estimation. In this work, our contributions are:

1) A novel pipeline that enables labeling of image datasets with different annotations and formats
2) The augmentation of this dataset without conversion to different formats
3) The validation of results in two industrial use cases with different computer vision tasks

II. RELATED WORK

A. Multi-task Learning

Multi-task learning [5] in the field of computer vision has shown recent success with methods such as Cross-Stitch [9] and UberNet [10]. As comparison, single-task models are more expensive to train and require a larger amount of annotated data than multi-task models. Moreover, knowing which tasks should be trained together could aid in improving prediction quality and acting as a preventative measure against the occurrence of negative transfer [11]. Performance improvement of a specific task can thus lead to performance degradation of other tasks [12]. Currently, libraries such as YOLOR [13] and YOLOv7 [14] provide configuration features to facilitate multi-task learning.

B. Image Annotation and Storing Formats

Labeling data is a critical component for deep learning as it provides the ground truth for estimating errors. Incorrectly labeled datasets lead to poor training and higher validation losses [15]. Nowadays, there are several machine learning based tools available that speed up the automatic labeling process and reduce human labor and time. MS COCO [16], PASCAL VOC [17], and YOLO are some commonly used benchmarks with different storage format styles. MS COCO is a rich (.json) format that offers a large variety of annotation storing structures. PASCAL VOC, on the other hand, supports a (.xml) structure that is useful for detection and segmentation tasks but has limitations in supporting key points. YOLO models accept (.txt) file format for detection tasks, however, YOLOv7 [14] accepts polygon segmentation and key point representations as well. While numerous labeling tools are available, each provides support for different export formats and annotation types [1]. For example, only a few tools offer direct labeling options for polygon segmentation, bounding boxes and key points simultaneously. And when the opportunity is provided, only a combined JSON export format is available, thereby not directly supporting state-of-the-art deep learning libraries. As a result, custom data loaders need to be developed to enable library-specific data formats.

C. Image Augmentation

Data augmentation is a form of data expansion that can improve model performance by assisting in its generalization, robustness and convergence. While augmentation is a well-known approach, libraries such as TensorFlow [18] and PyTorch [19] offer limited transformation strategies. Advanced libraries like Albumentations [7] provide a diverse range of transformations but currently lack in the transformation of a single benchmark format comprising polygons, keypoints, and bounding boxes simultaneously. Moreover, if custom and complex transformations are required, then libraries like OpenCV [20] are utilized for basic image manipulation algorithms.

D. Object detection and Pose estimation

Traditionally, keypoint estimation, segmentation, and object detection were considered separate problems. The solution for object detection evolved in the last 20 years from hand-crafted features to deformable transformers [21]. Research on segmentation started with thresholding and later led to methods such as semantic, instance, and panoptic segmentation. On the other hand, local features, called keypoints, became popular through corner and edge detectors [22], and today there are several ways to predict them, including Faster R-CNN [23], which is based on local keypoints. In the field of robotics and computer vision, segmentation masks are used to predict keypoints, as presented in [24]. Recently, combinations of both to predict human and object poses is addressed by [25]. In this, COCO is used as the primary format, emphasizing its relevance.

III. METHODOLOGY

In this section, we present our methodology which enables the annotation of images with multiple labels in multiple formats and the export of these in a generic format, for training a multi-task detection model.

A. Architecture

The multi-task architecture of ResNet-50 FPN Keypoint R-CNN [26] supports the combined data annotations and is utilized for training the detection model[1]. The output of the model contains all three detection formats, for object recognition, segmentation and feature extraction at the same time.

B. Data Annotation

Our pipeline starts with the image annotation of polygons, key points and bounding boxes in Label Studio. As Label Studio does not provide the functionality for simultaneously annotating features for polygons and key points, our approach updates polygon configurations by adding a key points configuration script to the user interface.Bounding boxes are extracted from the polygon annotations by selecting the outer bounds of the polygon coordinates. Following, all annotations are exported into a JSON-min format and converted into COCO format, containing all three data annotation types in a single output format.

C. Data Augmentation

Tools such as Albumentations [7] offer a wide range of image transformations. However, it does not offer augmentation support for polygon and run-length encoding COCO formats, limiting the support for an integrated augmentation strategy. To enable the augmentation of multi-label annotated data, such as bounding boxes, key points and COCO segmentation formats, the following steps are performed. First, the combined data (in COCO format) is loaded and extracted into separate arrays, such as bounding boxes, key points, area, etc. Second, polygons are converted to key points and appended to a key point array. Third, the conversion of (x,y) key points to (x,y,v) is done to maintain COCO standards, followed by

TABLE I: Frame rate and memory usage comparison between single- and multi-task models.

Model	Output	Inference (FPS)	Memory (MB)
Faster R-CNN	BB, class	11.1	~ 350
Mask R-CNN	Mask, BB, class	10.5	~ 350
Keypoint R-CNN	Kps, BB, class	9.4	~ 350
Multi-task model	Mask, kps, BB, class	8.7	~ 500
Mask + keypoint (in parallel)	Mask, kps, BB, class	4.8	~ 700

appending invisible key points to keep the number of labeled key points the same. Dataset augmentation is then performed using Albumentation's key point transformation strategy, after which the key points are converted back to polygons. As a result, this procedure enables the transformation of COCO polygon format by the Albumentation library.

IV. RESULTS AND DISCUSSION

This section presents the results of our multi-label annotation pipeline, its detection results and a discussion on its limitations.

A. Practical Use Cases and System Integration

The use cases aim to detect different objects for automation purposes. Two assembly sets are selected including a diesel engine and a planetary gearbox, and their internal parts (e.g., rocker arms, pushrods, bolts, gears, housing, etc.). The diesel engine [27] and planetary gearbox [28] datasets contain 195 and 150 RGB images, respectively, manually annotated by drawing polygons and labeling key points (see Fig. 2a and 2c). Augmentation is then performed on the annotated data by different transformation strategies, completing datasets of 280,000 and 170,000 images. Training of the multi-task models followed typical training and evaluation steps [26] with hyper-parameters such as momentum, weight decay, and regularization to achieve loss convergence in about 2000 epochs. The images were captured with an Intel Realsense D435 camera. All tools are open-source available from the dataset documentations [27], [28].

B. Object and Key point Detection

Fig. 2b and 2d present the output of the trained multi-task models on the two use cases with results for object detection and segmentation, and key point estimations. This demonstrates that a single multi-task model can utilize multiple data annotation formats as part of the dataset, train a model and achieve successful detection outputs. On average, the detection confidence is over 90 percent. Furthermore, as main beneficial outcome Table I presents that the achieved frame rate of the multi-task model is higher than running multiple models in parallel. For high-resolution camera input (1280×720), the difference is almost double, i.e., 8.7 FPS versus 4.8 FPS, respectively. Likewise, the memory requirements for a multi-task model is less than running multiple single-task models in parallel (i.e., 500 MB vs 700 MB).
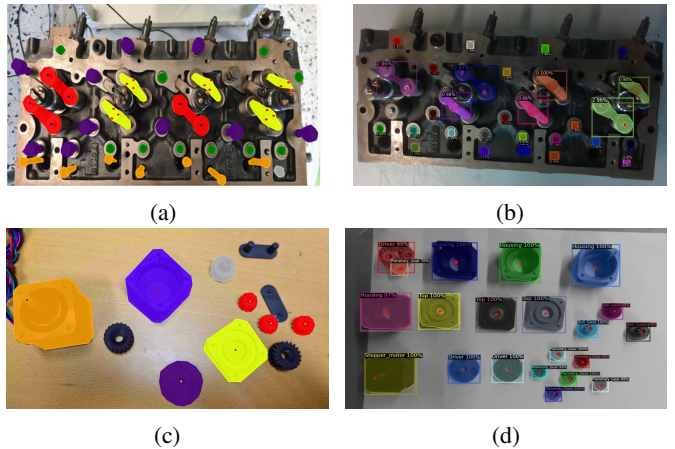

Fig. 2: Results for the Diesel engine (top row) and planetary gear (bottom row) datasets. (a) and (c) depict image annotations with polygons and key points. (b) and (d) depict the results of the trained multi-task model (bounding box and segmentation mask, and red key points).

C. Object Configuration and Depth Estimation

Two further functionalities are developed to demonstrate the use of multiple detection format outputs from the multi-task model. These are object configuration and depth estimation, based on the object segmentation masks and key points, respectively. Segmentation masks generated by the multi-task model can be used to determine rocker arm orientations with respect to the image, with the 2nd order moment of the mask. The three estimated key points on the rocker arm then verify correct configuration of the rocker arm object with respect to the engine. The process for finding the outer side for the correct configuration is to minimize the axis of the smallest 2nd order moment, then apply a cross-product between the axis coordinates and predicted key points. The one key point with an opposite sign to the others indicates the outer side. Fig. 1 illustrates the correct orientation of the rocker arm on the engine and verifies (in)correct placement.

To estimate the depth of the rocker arm target, with respect to the camera, two methods were considered: by segmentation mask or by key points (see Fig. 2b). In both cases the depth value from corresponding RGB pixel coordinates are retrieved from a depth camera (Intel Realsense D435). For a segmentation mask, the entire mask area as depth range is taken and an average depth value is returned, and for a key point, a single depth value is returned. The key point approach provides 10 times less difference in depth range (i.e., $468 \pm 5mm$) as compared to the segmentation mask, and is therefore found to be more reliable.

D. Discussion

The proposed approach enables images to be annotated by multiple different labels, including bounding boxes, polygons and key points, during a single annotation step, and store the result in a single convenient format (COCO). The annotations

are then all included in the same data augmentation process to collect a dataset utilized as input for training a multi-task model. In effect, this provides a single model with multiple detection outputs in different detection formats (bounding box, segmentation masks and key points). Compared to single-task models that only consider single annotation format input and detection output, the following benefits are identified. 1. Time-saving for image annotation and augmentation. All annotations can be done in a single image, without changing any formats for different annotation types. For example, in case of a single task-model, one image would need to be annotated three times separately for three different annotation types. 2. Memory space requirements saving in model training. As only a single multi-task model needs to be trained instead of multiple single-task models, size of the model is reduced. 3. Combining detection outputs. Detection modalities from the multi-task model can be combined to solve additional detection tasks, not directly provided by the model. For example, depth and object configuration from key points as shown by our use cases, or other post-prediction applications such as boundary or grasp pose detection, or the estimation of object properties.

A current limitation of the annotation pipeline is that it relies on existing annotation (Label Studio [8]) and augmentation (Albumentations [7]) tools. While these tools can be used under an open source license, access and support might change over time. Another limitation is that, still, annotation is a time-consuming step in the process of generating a learning-based detection model. Selection of which data to annotate and in which format (e.g., depth from pixel masks or key points) requires manual decisions and an iterative process to select a best outcome for the tasks. Automated and interactive annotation (e.g., based on machine learning [15]) and semi-automated, interactive image annotation for visual detection exist [29], but are still in early stages of research. This is a natural next-step in developing assistive tools for model training and planned as future work.

V. CONCLUSION

Learning-based multi-task models require suitable tools for the collection and annotation of data, especially when data needs multiple labels. In this work we proposed a single framework that collects and annotates all required labels (bounding boxes, polygons and key points) from a single image and stores it as a single benchmark format (COCO). Augmentation can then utilize this single format to generate a full dataset. The approach is validated with two use cases, where three different annotation types are needed per image. Results show that multi-task models provide benefits with respect to inference time and memory usage as compared to multiple single-task models. In addition, results show that multi-task models allow to combine detection outputs and solve additional detection tasks from a single model, such as the estimation of depth and object configuration.

REFERENCES

[1] A. I. Károly *et al.*, "Deep learning in robotics: Survey on model structures and training strategies," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 266–279, 2021.

[2] F. Piccialli *et al.*, "A survey on deep learning in medicine: Why, how and when?," *Information Fusion*, vol. 66, pp. 111–137, 2021.

[3] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and electronics in agriculture*, vol. 147, pp. 70–90, 2018.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[5] S. Vandenhende *et al.*, "Multi-task learning for dense prediction tasks: A survey," *IEEE T-PAMI*, vol. 44, no. 7, pp. 3614–3633, 2022.

[6] B. Pande *et al.*, "A review of image annotation tools for object detection," in *International Conference on Applied Artificial Intelligence and Computing*, pp. 976–982, 2022.

[7] Albumentations. https://albumentations.ai/, 2023.

[8] Label Studio. https://labelstud.io/, 2023.

[9] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3994–4003, 2016.

[10] I. Kokkinos, "UberNet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6129–6138, 2017.

[11] C. Fifty, E. Amid, Z. Zhao, T. Yu, R. Anil, and C. Finn, "Efficiently identifying task groupings for multi-task learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27503–27516, 2021.

[12] Z. Meng, X. Yao, and L. Sun, "Multi-task distillation: Towards mitigating the negative transfer in multi-task learning," in *IEEE International Conference on Image Processing (ICIP)*, pp. 389–393, 2021.

[13] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "You Only Learn One Representation: Unified network for multiple tasks," *arXiv preprint arXiv:2105.04206*, 2021.

[14] C.-Y. Wang *et al.*, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint arXiv:2207.02696*, 2022.

[15] P. Bhagat and P. Choudhary, "Image annotation: Then and now," *Image and Vision Computing*, vol. 80, pp. 1–23, 2018.

[16] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *European Conf. on Computer Vision (ECCV)*, pp. 740–755, 2014.

[17] M. Everingham, L. V. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, 6 2010.

[18] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[19] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[20] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[21] Z. Zou *et al.*, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, 2023.

[22] C. Harris, M. Stephens, *et al.*, "A combined corner and edge detector," in *Alvey vision conference*, vol. 15, pp. 10–5244, 1988.

[23] X. Ding *et al.*, "Local keypoint-based Faster R-CNN," *Applied Intelligence*, vol. 50, pp. 3007–3022, 10 2020.

[24] C.-C. Wong *et al.*, "Manipulation planning for object re-orientation based on semantic segmentation keypoint detection," *Sensors*, vol. 21, no. 7, p. 2280, 2021.

[25] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "kPAM: KeyPoint Affordances for Category-Level Robotic Manipulation," in *The International Symposium of Robotics Research*, pp. 132–157, 2019.

[26] Detectron2. https://github.com/facebookresearch/detectron2, 2023.

[27] G. Sharma, R. Pieters, and A. Angleraud, "Engine assembly dataset." http://dx.doi.org/10.5281/zenodo.7669593, Feb. 2023.

[28] G. Sharma and R. Pieters, "Planetary gear dataset." http://dx.doi.org/10.5281/zenodo.8223473, Aug. 2023.

[29] J. Bragantini, A. X. Falcão, and L. Najman, "Rethinking interactive image segmentation: Feature space annotation," *Pattern Recognition*, vol. 131, p. 108882, 2022.