

# Transfer Learning for the ICU: Comparing the Performance of YOLOv5 and Faster R-CNN X101-FPN

Ally Nakamura  
Stanford University

allynak@stanford.edu

Zane Durante\*

Stanford University  
durante@stanford.edu

Kachachan Em Chotitamavee  
Stanford University

kchotita@stanford.edu

Ehsan Adeli\*

Stanford University  
eadeli@stanford.edu

Roshan Swaroop  
Stanford University

swaroop1@stanford.edu

Fei Fei Li\*

Stanford University  
feifeili@cs.stanford.edu

## Abstract

*The recent success of transfer learning in a variety of object detection tasks have proven particularly useful in fields such as medicine where training data is limited or difficult to access. Empirical studies on transfer learning has also found that the characteristics of the pre-training data itself has a very minor impact on the performance of transfer learning models themselves. But what about the fine-tuning dataset? What characteristics of the transfer learning dataset can we leverage to maximize transfer learning performance? This study proposed two such characteristics, and empirically investigated the effects of varying the fine-tuning dataset size, as well as varying whether the fine-tuning datasets were in-distribution, out-of-distribution, or very out-of-distribution, on the performance of transfer learning object detection tasks using the state-of-the-art YOLOv5 and Detectron2 Faster R-CNN models. Our overall findings suggest that both models actually perform better on transfer learning tasks with images that are significantly out-of-distribution, and that reducing the size of the fine-tuning dataset generally did not have a significant impact on model performance. However, our results may be skewed by the homogeneous nature of the datasets we used, and further study would be required to validate our findings.*

## 1. Introduction

The Stanford Vision Lab’s Partnership in AI-Assisted Care (PAC) is an interdisciplinary collaboration between the School of Medicine and the Computer Science department focusing on cutting edge computer vision and machine learning technologies to solve some of healthcare’s most important problems. One of PAC’s primary focuses is to enable hardware-equipped hospitals to better understand pa-

tient status in clinical settings such as Intensive Care Units (ICUs), where comprehensive understanding of a patient’s real-time condition can play a critical difference in recovery. Activity detection in ICUs is currently performed manually by trained personnel, primarily nurses, who log the activities as they occur. This process is both expensive and time consuming. An ideal system would automatically give an annotated list of all activities that occurred in the ICU over the day, thus reducing the monitoring workload of trained personnel and leading to a quicker and safer recovery of the patient, while providing benefits such as activity-based costing.

Despite the clear benefits object detection models can provide in clinical settings, there are several barriers to implementation. The most prominent challenge is in collecting and annotating enough medical data within a reasonable timeframe to sufficiently train a model and reach useful performance. This is especially true in clinics that handle highly protected patient data, where most data must be annotated manually by parties that have clearance. As such, there is a significant need for pre-trained models that perform well through transfer learning on out-of-distribution clinical datasets to reduce resource costs and training time. The relevant pre-trained object detection models are described below.

### 1.1. Varying Pre-training data for Transfer Learning

Because medical data is sparse and difficult to access, it would be greatly beneficial if medical models could be effectively trained with transfer learning. This study explored the performances of a range of models that were first trained on COCO, and then fine-tuned on a range of smaller medical datasets afterwards.

It has been empirically shown that transfer learning is effective, especially in a medical setting. As Huh et. al. con-

cluded, pre-training with ImageNet is "the de-facto method to solving a wide range of computer vision problems". Morid et. al. validated this approach by performing a meta-analysis of 102 different CNN models that used transfer-learning through ImageNet for medical tasks, and found that studies generally achieve reasonable performance on their target task, and that transfer learning through ImageNet is an "effective way to approach medical tasks".

But how important are the underlying features of the pre-training dataset that is being used to perform transfer learning? Huh et. al. explored the effects of modifying ImageNet on transfer-learning performance, and found that a drastic reduction in the number of classes or the number of images per class does not significantly affect the transfer-learning performance. For instance, using half the amount of pre-training data per class reduced the transfer learning mean average Precision by only 3.7, from 58.3 to 54.6. Similarly, using only 127 instead of the original 1000 classes lead to a average decrease in mean average Precision of only 3.15, from 55.25 to 52.1. Huh et. al. concluded that, given the same budget of pre-training images, training with fewer classes but more images per class performed slightly better at transfer tasks.

## 1.2. Varying fine-tuning data and model for Transfer learning

This study extends the work of Huh et. al. and explores the effects of varying the size as well as the in or out-of-distribution nature of the transfer learning training data, while keeping the pre-training data constant. To align with the needs of the Stanford Vision Lab, the study focuses on object detection tasks, and the YOLOV5 and Detectron2 Faster R-CNN X101-FPN models were chosen for its state-of-the-art effectiveness in such tasks. Generally speaking, the models differ in that YOLOV5 is generally faster but less accurate, whereas Faster R-CNN is better for smaller datasets. The PAC is considering using one of these models for further work in object detection, and this study seeks to either validate or invalidate that choice. Both models were pre-trained on the COCO dataset, and then fine-tuning was performed on an in-distribution data set, an out-of-distribution dataset, and a very out-of-distribution dataset. The sizes of each of these datasets were also varied, and the relative mAP were measured for each of these tasks. See section 4, Methods and Experimental Setup, for more details.

## 1.3. Summary of results

The following is a summary of our key findings:

1. *How important is the in or out-of-distribution nature of the transfer learning data set?*

Both models performed better on the out-of-distribution

data sets compared to the in-distribution data sets, but this might be particular to the datasets we used for this experiment.

2. *How important is the size of the fine-tuning data set in in-distribution cases?*

The size of the fine-tuning data set did not significantly impact the performance of the Faster R-CNN X101-FPN model, while shrinking the data set led to poor performance of the YOLOv5 model.

3. *How important is the size of the fine-tuning data set in out-of-distribution cases?*

Decreasing the size of the training data did not severely impact either of the models, but YOLOv5 displayed more consistency across all training sizes.

4. *How important is the size of the fine-tuning data in very out-of-distribution cases?*

Both models performed reasonably well on the BCCD dataset, with minimal changes in performance across all variations in training data set size.

A more detailed discussion of our findings is provided under the Discussion section.

## 2. Related Work

The interplay between computer vision and healthcare settings is still only newly understood, with a foremost publication emerging in 2021, Deep learning-enabled medical computer vision by Esteva et. al. The "clinical deployment" section has greatly influenced the work of PAC by structuring this ICU task into four broad categories: assessment of data, planning for model limitations, community participation, and trust building. The first two will be the primary focus of our project as we attempt to detect medical objects in ICUs with reasonable accuracy under both data annotation constraints and out-of-distribution data samples for a large-scale object recognition dataset such as COCO. The article also discusses "ambient intelligence" applications of computer vision which describes techniques where a continuous, non-invasive awareness of activity in a physical space can "provide clinicians, nurses, and other healthcare workers with assistance such as patient monitoring, automated documentation, and monitoring for protocol compliance." A richer technical understanding of this problem space has come from a flagship paper of the PAC, Multi-Object Multi-Actor Activity Parsing by Zelun Luo et. al. as part of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021). The publication delineates a framework for extracting meaning about complete actions from complex scenes, with the goal of applying such techniques to healthcare settings. MOMA introduces activity parsing, a partonomy-based activity recognition framework that considers the hierarchy and composition of complex human activities. More specifically, the approach utilizes hyper-

graphs and hierarchical video annotations that identify the activity, sub-activity, and atomic actions that round out a complex scene. Alongside these structural approaches to semantically understanding a scene, the publication introduces a hypergraph neural network which takes in both temporal and image frame information to determine activity in a scene.

A number of papers have also deeply studied transfer learning in CNN, but the majority of them have been mostly concerned with the factors that affect pre-training efficacy. For example, Agarwal et. al. studied the question of whether overfitting is a problem in pre-training and if training should be terminated early to prevent such issues, and Yosinski et. al. explored the layers that should be trained to maximize transfer learning effectiveness. Azizpour et. al. also explored the importance of the distance between the source and target class, as well as the various factors that impact transfer learning performance, and found that transfer learning is more effective, and the various negative factors are less punitive, the closer the target task is to the source task. See section 1.1 for further discussion on some of these studies.

Our work is an extension of these studies, and focused only on the impact of varying fine-tuning dataset distribution characteristics and size in image classification tasks.

### 3. Data

For the pre-training process for the object classification task, the COCO dataset was the primary dataset of choice. Although the COCO dataset is relatively small, consisting of only 328,000 images and 80 object categories, we chose the dataset because it better aligned with the objectives of the PAC project and because the performance impacts of pre-training on a smaller dataset should be fairly minimal, as Huh et al. concluded. We also used YOLOV5 and Faster R-CNN X101 FPN models that were pre-trained on the same COCO dataset for the purpose of this experiment, which we selected because the PAC is also considering using these models in future works.

For fine-tuning, we used three different data sets, one for each of the following segment: in-distribution, out-of-distribution, and very out-of-distribution. For the in-distribution case, we used a vehicles dataset from Open Images. This dataset contains 627 images and classifies each image into one particular type of vehicle, such as car, bus, or truck. The majority of the classes provided in this dataset also coincide with classes that exists in COCO, making this dataset in-distribution for our purposes. For the out-of-distribution dataset, we used a Mask Wearing dataset collected by Cheng Hsun Teng from Eden Social Welfare Foundation, Taiwan and relabeled by Roboflow. This dataset contains 149 images and separates individuals who wear masks from individuals who do not. While COCO does

have a persons class and can accurately detect that a person exists, it does not out-of-the-box know how to distinguish whether an individual is wearing a mask or not, making this transfer task out-of-distribution. Lastly, for the very out of distribution dataset, we used the BCCD dataset, which is a dataset of blood cell photos, provided for open source usage by cosmicad and akshaylambda on GitHub. The dataset contains 364 images and classifies images into the classes of White Blood Cell, Red Blood Cell, and Platelets.

Since the primary purpose of the study was to measure the impact of fine-tuning dataset size on each of these datasets, to make the comparisons congruent we preprocessed the data by taking a subsample of images from each of these datasets in three different ways.

We first created a sample set such that the total amount of objects (defined as number of class annotations) in each of the training dataset is the same. For instance, the BCCD set has 4888 total labels, whereas the masks set only has 304 labels. As such, we took a subsample of 300 labels from each of these dataset to be used for the rest of the study. For our experiment, we will vary the amount of labels we use for fine-tuning (150, 300) and measure the impact of that on our fine-tuning results.

We also created a sample set that has the same amount of images from each of the dataset; since the mask dataset has the lowest at 105 images, we took samples of 100 images from each of the datasets. For our experiment, we will vary the amount of images we use for fine-tuning (50, 100) and measure the impact of that on our fine-tuning results.

Lastly, we also processed the entire datasets as a whole. For this portion of the experiment, we will simply take a flat percentage of the datasets we have (25%, 50%, 75%, 100%), and measured the impact of that on our fine-tuning results.

In summary, we separately controlled for number of class instances across datasets, number of raw images fed into each object detector across datasets, and then exhausted the three datasets to evaluate differences in the underlying object detector models themselves.

Since the class distribution of each of these datasets were also rather skewed (with red blood cells being roughly 80% of all the objects in the BCCD dataset, and cars being almost 50% of all the objects in the Vehicles dataset), we also took care to ensure that the class distribution of the subsamples we trained on were more uniform by selecting images to correspond to each class equally when possible. This was a reasonable preprocessing step to do because the eventual dataset of medical data that our model will be trained on will also most likely be approximately uniform as well.

For the sake of consistency, a resampling of each of these datasets, to all be 416x416 in size, was used as well. For both the masks and vehicle dataset in particular, auto-orientation was also done to ensure that landscape and por-

trait images are aligned in the same direction.

The figures below shows an example image, along with their ground truth bounding boxes, from each of the datasets we used. (Figures 1-3)

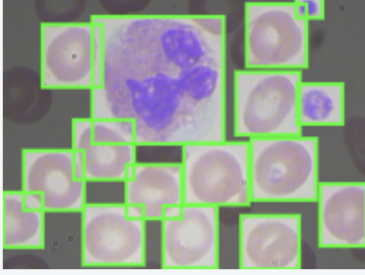


Figure 1. Example image from BCCD data set



Figure 2. Example image from mask data set

## 4. Methods and Experimental Setup

For our experiment, we were looking in particular for object detectors to use on images. In clinical settings such as ICUs where PAC eventually seeks to provide real-time information to physicians and other medical staff, tasks



Figure 3. Example image from vehicles data set

such as specific instance segmentation are not as high of a priority as simply generating bounding boxes efficiently. Thus, we used two models that appeared on the Papers With Code COCO test-dev benchmark leaderboard: YOLOv5 and Faster R-CNN X101-FPN (Detectron2).

The YOLOv5 model is a state-of-the-art CNN based image detection model that "predicts bounding boxes and class probabilities directly from full images in one evaluation". The particular model we used for this study was developed by Glenn Jocher from Ultralytics. YOLOv5 specifically is the most actively maintained port of YOLO in Python and a previous iteration, YOLOv4, performed extremely competitively on the COCO test-dev leaderboard. YOLOv5 divides images into a grid system, and each grid detects objects within itself. YOLOv5 uses a two-stage object detector, and separates the model into the input, backbone, neck, and head sections. The backbone model is a feature extractor that uses a classification model such as VGG16. The neck is a feature aggregator that collects feature maps from differing stages from the backbone model. The head is an object detector that reports whether a particular grid has an object in it but not necessarily what class of object it is. All the sections are used in conjunction according to the model shown in figure 4, and each grid predict the existence of bounding boxes and their coordinates relative to the grid coordinate, alongside the object labels and their probability of being within that grid. Then, YOLO uses non-maximal suppression to retain only the largest prediction, to avoid duplicate predictions between adjacent cells.

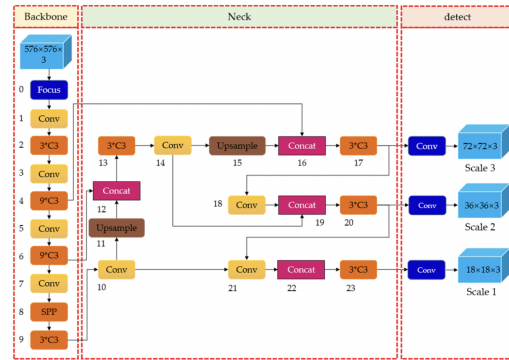


Figure 4. Model Architecture for YOLOv5

The specific Detectron2 model used in this study was the Faster R-CNN X101-FPN developed by Facebook AI which was pre-trained on the 2017 MS COCO dataset. Faster R-CNN is an improvement on Fast R-CNN, which itself is an improvement on R-CNN. R-CNN uses selective search to extract region proposals of bounding boxes, computes CNN features on each of those bounding boxes, and then classifies the regions from there. Fast R-CNN improves on R-CNN by computing the CNN features for each region

of interest in a single forward pass, achieving significant space and time complexity improvements. Faster R-CNN improves on this still by combining the region proposal and evaluation together by using a Region Proposal Network that operates and shares convolutional features with the Fast R-CNN model, yielding state-of-the-art performance in objection detection tasks.

For our experiments, the datasets were also divided into their respective training, validation, and testing splits as described below:

- For the Mask Wearing dataset, both models relied on a train-valid-test split of 105-29-15.
- For the BCCD dataset, both models relied on a train-valid-test split of 765-73-36.
- For the Vehicles-OpenImages dataset, both models relied on a train-valid-test split of 439-125-63.

Throughout the experimentation, the training data sizes were varied, while the validation and testing sets remained the same.

After obtaining the pre-trained models, we first verified the pre-trained models of YOLOv5 and Faster R-CNN X101-FPN on a small subsample of COCO to ensure that both models are congruent with one another and can overfit to the data. We achieved an accuracy rate of over 98% and judged that to be sufficient for the purpose of our experiment.

Then, for each of the test cases (namely the 25%, 50%, 75%, and 100% case for each of the three data sets; the 150 objects and 300 object cases for each of the three data sets; and the 50 and 100 images cases for each of the three data sets), we perform the following procedures: we first remove the classification layer from both models, as that is specific to the pre-training task and are not reusable. Next, we randomly initialize a new classification layer with the correct number of output units for each of the target tasks in our experiment, and append that to both models. Lastly, we run stochastic gradient descent with nesterov momentum on the binary cross entropy loss, with a learning rate of 0.001, which is 1/10th of the learning rate used for the pre-training tasks. We made this choice following Agrawal et. al., who recognized that it was beneficial to do so to "prevent clobbering the CNN's initialization to control overfitting". To be concrete, we compute the BCE loss through PyTorch's BCEwithLogitsLoss function, which is a combination of the Sigmoid and BCE loss layers that takes advantage of the log-sum-exp trick for numerical stability. The full loss function can be described as:

$$l(x, y) = L = \{l_1, \dots, l_N\}^T$$

where  $l_n = -w_n[y_n \cdot \log(\sigma(x_n)) + (1 - y_n) \cdot \log(1 - \sigma(x_n))]$

The other hyperparameters were chosen in accordance with the work of Huh et. al., who found that this particular set of parameters (i.e. SGD with nesterov momentum=0.9) yield the highest accuracy across all their experimentation.

The standardized metrics we used across our experiments to evaluate performance are consistent with offerings on MS COCO's detection evaluation page. We primarily used average precision (AP) at an intersection over union (IoU) of .50 for simplicity based on its use in PASCAL VOC, but we also took note of AP@[0.5:0.95] for our own comparisons. (This varies the IoU values from 50 to 95, at a step of 5 to create 10 precision-recall pairs which can then be averaged to produce the metric.) We also follow COCO's somewhat dubious norm of describing AP vs. mAP where "AP is averaged over all categories. Traditionally, this is called 'mean average precision' (mAP). We make no distinction between AP and mAP (and likewise AR and mAR) and assume the difference is clear from context."

## 5. Results

| Model                      | AP-mask | AP-no-mask | AP50  |
|----------------------------|---------|------------|-------|
| Faster R-CNN X101 FPN-25%  | 52.54   | 18.79      | 60.29 |
| YOLOv5-25%                 | 29.6    | 2.14       | 15.9  |
| Faster R-CNN X101 FPN-50%  | 53.88   | 24.85      | 71.47 |
| YOLOv5-50%                 | 29.8    | 12.6       | 15.1  |
| Faster R-CNN X101 FPN-75%  | 55.24   | 26.02      | 74.32 |
| YOLOv5-75%                 | 28.1    | 19.8       | 14.7  |
| Faster R-CNN X101 FPN-100% | 57.21   | 27.08      | 76.18 |
| YOLOv5-100%                | 27.5    | 23.0       | 14.9  |

Table 1. Average precision results for Faster R-CNN X101 FPN and YOLOv5 models that were trained on 25%, 50%, 75%, and 100% of the mask training data.

| Model                      | AP-Platelets | AP-RBC | AP-WBC | AP50  |
|----------------------------|--------------|--------|--------|-------|
| Faster R-CNN X101 FPN-25%  | 35.47        | 54.86  | 72.07  | 84.09 |
| YOLOv5-25%                 | 83.8         | 76.9   | 79.5   | 95.1  |
| Faster R-CNN X101 FPN-50%  | 35.35        | 58.66  | 74.62  | 84.10 |
| YOLOv5-50%                 | 84.4         | 80.8   | 81.2   | 95.7  |
| Faster R-CNN X101 FPN-75%  | 36.72        | 59.31  | 74.53  | 86.12 |
| YOLOv5-75%                 | 85.1         | 84.3   | 96.7   | 88.4  |
| Faster R-CNN X101 FPN-100% | 38.78        | 60.44  | 74.89  | 87.86 |
| YOLOv5-100%                | 87.2         | 87.0   | 98.0   | 90.7  |

Table 2. Average precision results for Faster R-CNN X101 FPN and YOLOv5 models that were trained on 25%, 50%, 75%, and 100% of the BCCD training data.

| Model                      | AP-Ambulance | AP-Bus | AP-Car | AP-Motorcycle | AP-Truck | AP50  |
|----------------------------|--------------|--------|--------|---------------|----------|-------|
| Faster R-CNN X101 FPN-25%  | 93.63        | 55.02  | 44.32  | 0.0           | 0.0      | 50.43 |
| YOLOv5-25%                 | 18.0         | 3.4    | 2.5    | 19.6          | 12.2     | 10.4  |
| Faster R-CNN X101 FPN-50%  | 93.44        | 61.05  | 45.75  | 0.0           | 0.0      | 51.68 |
| YOLOv5-50%                 | 54.7         | 15.6   | 11.5   | 22.3          | 11.5     | 23.1  |
| Faster R-CNN X101 FPN-75%  | 93.50        | 65.14  | 44.34  | 0.0           | 0.0      | 51.70 |
| YOLOv5-75%                 | 56.6         | 32.2   | 20.0   | 25.6          | 13.4     | 35.7  |
| Faster R-CNN X101 FPN-100% | 93.72        | 66.89  | 44.74  | 0.0           | 0.0      | 51.80 |
| YOLOv5-100%                | 63.6         | 50.9   | 28.6   | 39.6          | 21.8     | 40.9  |

Table 1. Average precision results for Faster R-CNN X101 FPN and YOLOv5 models that were trained on 25%, 50%, 75%, and 100% of the vehicles (in-distribution) training data.

| Model                 | BCCD (300) | Mask (300) | Vehicle (300) | BCCD (150) | Mask (150) | Vehicle (150) |
|-----------------------|------------|------------|---------------|------------|------------|---------------|
| Faster R-CNN X101 FPN | 75.25      | 76.02      | 50.68         | 73.12      | 72.38      | 48.92         |
| YOLOv5                | 8.99       | 3.8        | 5.48          | 6.77       | 14.9       | 8.6           |

Table 2. Average precision (AP50) results for Faster R-CNN X101 FPN and YOLOv5 models that were trained on 300 objects and 150 objects.)

| Model                 | BCCD (100) | Mask (100) | Vehicle (100) | BCCD (50) | Mask (50) | Vehicle (50) |
|-----------------------|------------|------------|---------------|-----------|-----------|--------------|
| Faster R-CNN X101 FPN | 79.24      | 82.11      | 49.43         | 77.89     | 71.03     | 47.95        |
| YOLOv5                | 8.99       | 26.9       | 4.3           | 8.2       | 13.9      | 6.4          |

Table 3. Average precision (AP50) results for Faster R-CNN X101 FPN and YOLOv5 models that were trained on 100 images and 50 images.

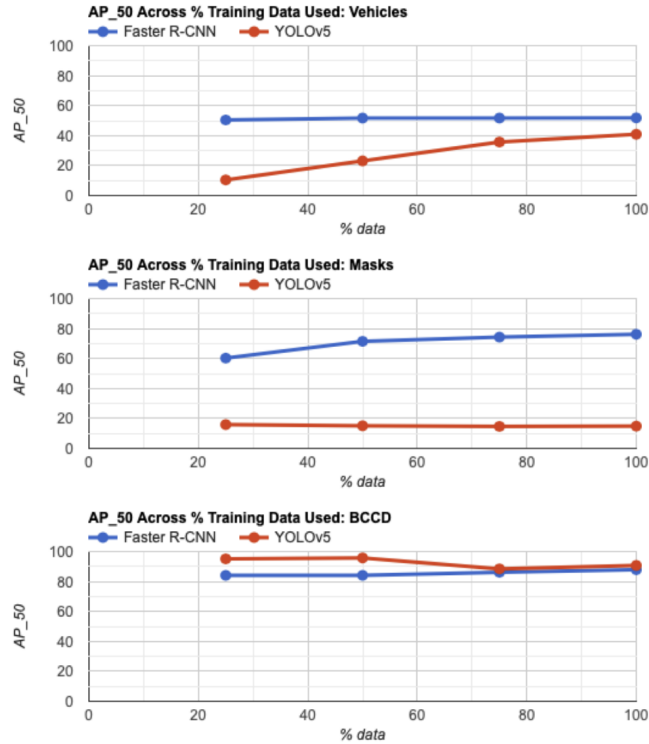


Figure 5. AP performance of both models across varying training data set sizes from 25% to 100% for the three data sets.

## 6. Sample Model Results

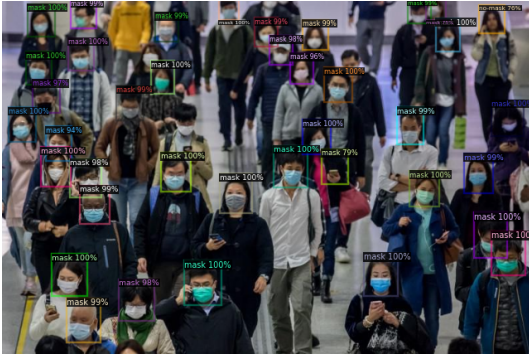


Figure 6. Faster R-CNN X101 FPN bounding box results on the full Mask Wearing training dataset.

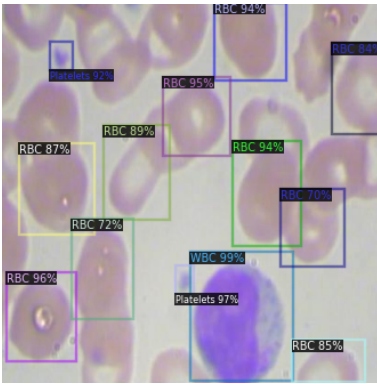


Figure 7. YOLOv5 bounding box results on the full BCCD training dataset.

## 7. Discussion

A more detailed analysis of the results are provided as follows:

### 1. How important is the in or out-of-distribution nature of the transfer learning data set?

Our results suggest that both YOLOv5 and Faster R-CNN X101 FPN perform relatively well in transfer learning for both in and out-of-distribution (OOD) cases. Specifically, when standardizing the number of objects across the training datasets, Faster R-CNN X101 FPN performed best on the OOD (Mask) and very OOD (BCCD) data, while YOLOv5 also performed best on the BCCD and Mask data. When standardizing the raw image counts across the training data sets, we saw similar results, where both Faster R-CNN X101 FPN and YOLOv5 performed better on the OOD data sets than the in-distribution dataset.

We propose a potential explanation for the models' performance on the in-distribution data set: compared to both

the OOD data sets (especially the BCCD data), the in-distribution Vehicles data set proved to be much more heterogeneous since it contained a higher number of classes and variation in the objects. Conversely, while the BCCD dataset was extremely OOD and unseen during the pretraining process, the dataset was much more homogenous (microscopic blood cells tend to appear the same), leading to higher accuracy scores overall. Thus, we predict that the results may vary (and accuracies would be higher) if we were to perform this same experiment with a more homogenous in-distribution dataset.

### 2. How important is the size of the fine-tuning data set in in-distribution cases?

Our results indicate that the size of the fine-tuning data set matters more for the YOLOv5 model compared to Faster R-CNN X101 FPN. While YOLOv5 saw a loss of nearly 30.5 AP when using the full Vehicles-OpenImage data set vs. 25% of the dataset, the Faster R-CNN X101 FPN model observed barely any difference, with a decrease of only 1.37 AP between the two trials. Thus, Faster R-CNN X101 FPN delivers more consistent results regardless of variations in training data size, while YOLOv5 suffers more in performance when the data set is sparse. These results reaffirm the findings from previous studies that Faster R-CNN X101 FPN is better suited for smaller training data sets since its performance is less severely impacted by the availability of training data.

### 3. How important is the size of the fine-tuning data set in out-of-distribution cases?

For the OOD Mask dataset, both models performed consistently well regardless of variations in the size of the training dataset, with the YOLOv5 model remaining more consistent across all variations. While the Faster R-CNN X101 FPN model decreased 15.89 in AP when comparing the 100% and 25% training sets, YOLOv5 actually increased by 1.0. Unlike the in-distribution cases, our results show that the size of the OOD data set does not affect the performance of the YOLOv5 model, while impacting the performance of the Faster R-CNN X101 FPN model slightly more. In regards to the actual accuracy metrics between the two models, Faster R-CNN X101 FPN performed significantly better with an AP50 of 76.18 when trained on the full dataset compared to an AP50 of 14.9 achieved by the YOLOv5 model; therefore, our results indicate that Faster R-CNN X101 FPN may be a better candidate for transfer learning to OOD datasets that are not too far from the original distribution.

### 4. How important is the size of the fine-tuning data in very out-of-distribution cases?

For the BCCD data set, both models performed consistently well regardless of variations in the size of the training data set, with the YOLOv5 model remaining more consistent across all variations. Between the 100% and

25% training sets, the Faster R-CNN X101 FPN model saw a decrease of 3.77 in AP, while YOLOv5 saw an increase in 4.4 AP. These results show that neither of the models were severely impacted in performance due to variations in training data size. Additionally, both models showed significantly higher accuracies in performance compared to the other two data sets, with Faster R-CNN X101 FPN achieving 87.86 AP and YOLOv5 achieving 90.7 AP when trained on the full available training set. Thus, our metrics show that both models are capable of transfer learning to relatively homogeneous OOD data sets with high success. Overall, our results indicate that while both models were able to perform reasonably consistently on both OOD data sets, Faster R-CNN X101 FPN is more ideal for application in clinical settings due to its consistency and high accuracy across all data sets. Not only is it minimally impacted by small training data sets, which is optimal in situations where medical data is less accessible, but it also proved to perform relatively better on both the in-distribution and OOD data sets. One downside regarding Faster R-CNN X101 FPN is its training time since YOLOv5 has a significantly faster training time, which could prove to be necessary for clinics that have less resources available for training.

Lastly, we can also be reasonably certain that our results did not suffer from overfitting as we made efforts to monitor the loss and accuracies on the validation set. To prevent either model from overfitting, we plotted each of the results from the validation set and found that none of the results exhibited characteristic overfitting.

## 8. Conclusion

This study aimed to support the efforts of the clinical efforts by PAC by observing the performances of two state-of-the-art object detection models on varied training data set sizes and both in and out-of-distribution data. While the data sets used are not perfectly representative of those encountered in clinical settings, we were able to determine that both models can be fine-tuned on OOD datasets to perform object detection tasks. By varying amounts of training data used in this fine-tuning process, we observed that even partial datasets as small as 25 percent of the overall size can achieve reasonable performance, which validates the path for clinical transfer learning. Medical image data often requires extensive domain knowledge and time to annotate, so these results indicate that even sparse representation of each clinical class may lead to worthwhile results.

### 8.1. Future Work

One of the most immediate areas to expand on this project would be further characterizing the differences in which OOD data affects transfer learning. While the BCCD

dataset was the furthest OOD, it had a much higher degree of homogeneity in its image samples, so controlling for discrepancies such as these in further experiments is key. In addition, once PAC reaches a critical mass of clinical ICU data, findings from this project can be used to effectively transfer learn for real clinical use; our results support that even partial datasets can reasonably perform object detection tasks.

## 9. Contributions and Acknowledgements

Ally Nakamura, Kachachan Chotitamanvee, and Roshan Swaroop contributed equally to building the models, performing the experiments, and writing the paper.

\*We are deeply thankful for the help of Zane Durante, who helped scope the original project, the milestone, and helped the CS 231N team pivot and scope a new project after the proposed project no longer became feasible; Ehsan Adeli, who provided consideration on identifying relevant research areas that spanned both PAC's work and CS 231N; and Fei Fei Li, who approved the original project and oversees PAC's ICU research subgroup.

## 10. References

1. Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015 Figure copyright 2015, Ross Girshick
2. He et al, "Mask R-CNN", arXiv 2017
3. Morid, Mohammad Amin, Alireza Borjali, and Guilherme Del Fiol. "A Scoping Review of Transfer Learning Research on Medical Image Analysis Using ImageNet." *Computers in Biology and Medicine* 128 (2021): 104115. <https://doi.org/10.1016/j.compbimed.2020.104115>.
4. [1]Huh, M., Agrawal, P., and Efros, A. A., "What makes ImageNet good for transfer learning?", *arXiv – prints*, 2016.
5. Z. Luo et. al, 'MOMA: Multi-Object Multi-Actor Activity Parsing', *Advances in Neural Information Processing Systems*, 2021, 34, 17939–17955.
6. Malhotra, K. R., Davoudi, A., Siegel, S., Bihorac, A., Rashidi, P. (2018). Autonomous detection of disruptions in the intensive care unit using deep mask RCNN. *Conference on Computer Vision and Pattern Recognition Workshops. IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Workshops*, 2018, 1944–1946. <https://doi.org/10.1109/CVPRW.2018.00241>
7. P. Agrawal, R. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition.
8. Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M., "YOLOv4: Optimal Speed and Accuracy of Object Detection", *arXiv – prints*, 2020.
9. Yosinski, J., Clune, J., Bengio, Y., and Lipson, H., "How transferable are features in deep neural networks?", *arXiv e-prints*, 2014.

10. H. Azizpour, A. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 36–45, 2015.
11. “Vehicles-OpenImages Dataset” Roboflow. [Online]. Available: <https://roboflow.com/>.
12. “BCCD Dataset” Roboflow. [Online]. Available: <https://roboflow.com/>.
13. “Mask Wearing Dataset” Roboflow. [Online]. Available: <https://roboflow.com/>.
14. “YOLOv5” Roboflow. [Online]. Available: <https://models.roboflow.com/object-detection/yolov5>
15. “Detectron 2” Roboflow. [Online]. Available: <https://models.roboflow.com/object-detection/detectron2>