📚 This guide explains how to use **Weights & Biases** (W&B) with YOLOv5 🚀. UPDATED 29 September 2021.

About Weights & Biases

Think of W&B like GitHub for machine learning models. With a few lines of code, save everything you need to debug, compare and reproduce your models — architecture, hyperparameters, git commits, model weights, GPU usage, and even datasets and predictions.

Used by top researchers including teams at OpenAI, Lyft, Github, and MILA, W&B is part of the new standard of best practices for machine learning. How W&B can help you optimize your machine learning workflows:

- Debug model performance in real time
- GPU usage visualized automatically
- Custom charts for powerful, extensible visualization
- Share insights interactively with collaborators
- Optimize hyperparameters efficiently
- Track datasets, pipelines, and production models

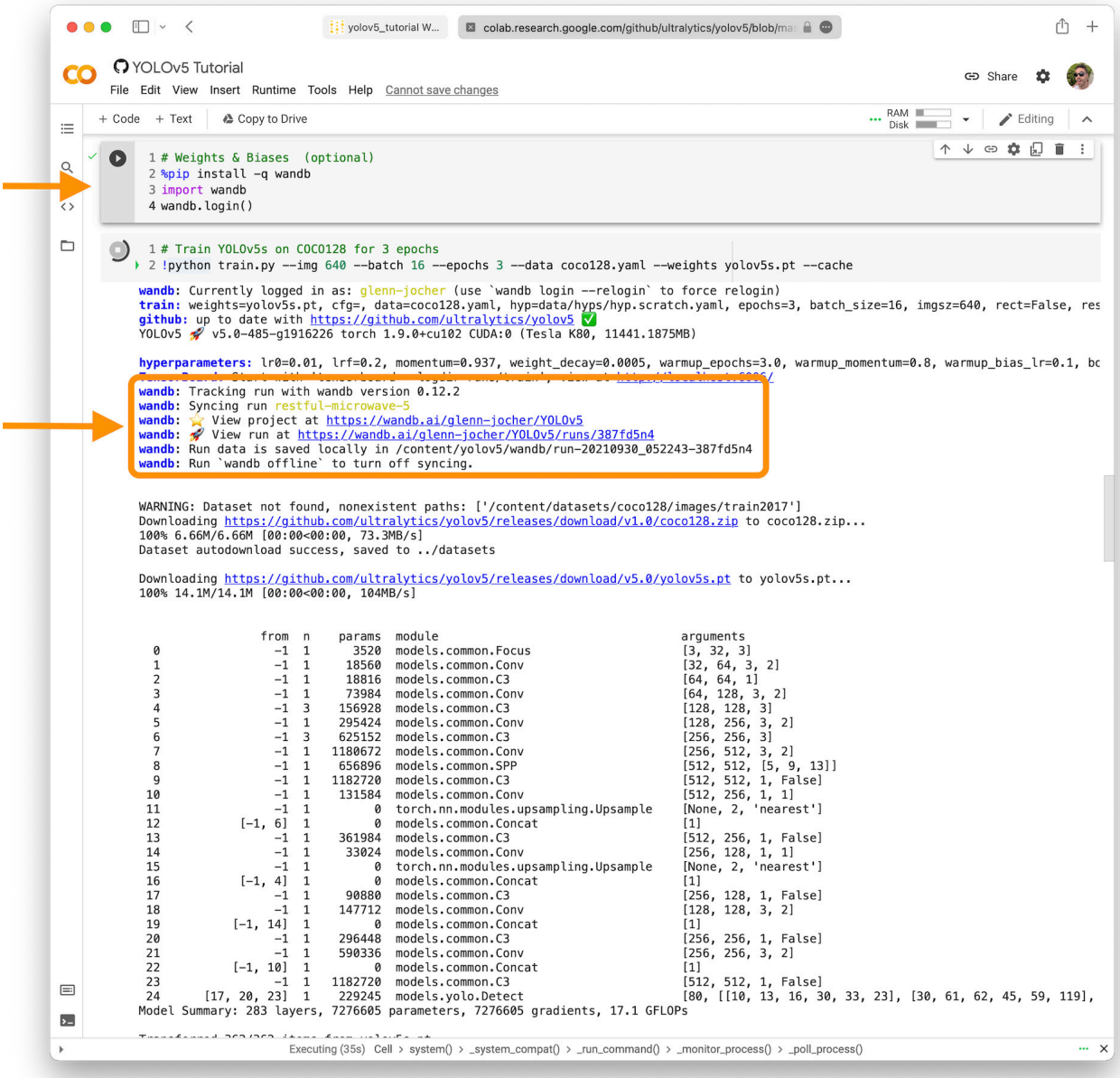
First-Time Setup

▼ Toggle Details
When you first train, W&B will prompt you to create a new account and will generate an **API key** for you. If you are an existing user you can retrieve your key from https://wandb.ai/authorize. This key is used to tell W&B where to log your data. You only need to supply your key once, and then it is remembered on the same device. W&B will create a cloud **project** (default is 'YOLOv5') for your training runs, and each new training run will be provided a unique run **name** within that project as project/name. You can also manually set your project and run name as:

```
$ python train\.py --project ... --name ...
```

Open in Colab Open in Kaggle

YOLOv5 notebook example:



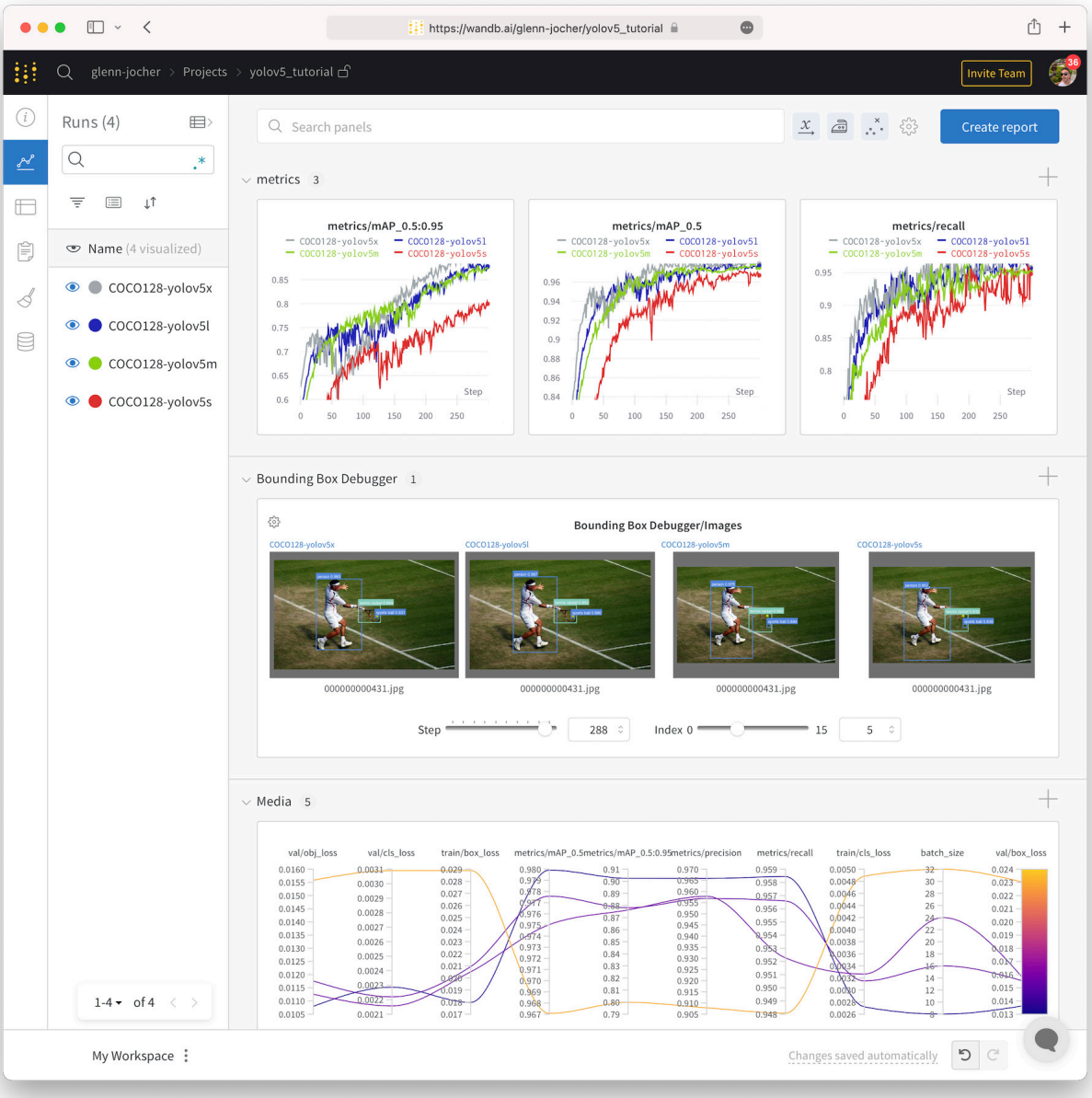
Viewing Runs

▼ Toggle Details

Run information streams from your environment to the W&B cloud console as you train. This allows you to monitor and even cancel runs in **realtime** . All important information is logged:

- Training & Validation losses

- Metrics: Precision, Recall, mAP@0.5, mAP@0.5:0.95
- Learning Rate over time
- A bounding box debugging panel, showing the training progress over time
- GPU: Type, **GPU Utilization**, power, temperature, **CUDA memory usage**
- System: Disk I/0, CPU utilization, RAM memory usage
- Your trained model as W&B Artifact
- Environment: OS and Python types, Git repository and state, **training command**



Advanced Usage

You can leverage W&B artifacts and Tables integration to easily visualize and manage your datasets, models and training evaluations. Here are some quick examples to get

you started.

▼ Details

1. Visualize and Version Datasets

Log, visualize, dynamically query, and understand your data with W&B Tables. You can use the following command to log your dataset as a W&B Table. This will generate a `{dataset}_wandb.yaml` file which can be used to train from dataset artifact.

▶ **Usage**

2: Train and Log Evaluation simultaneousy

This is an extension of the previous section, but it'll also training after uploading the dataset. **This also evaluation Table** Evaluation table compares your predictions and ground truths across the validation set for each epoch. It uses the references to the already uploaded datasets, so no images will be uploaded from your system more than once.

▶ **Usage**

3: Train using dataset artifact

When you upload a dataset as described in the first section, you get a new config file with an added `_wandb` to its name. This file contains the information that can be used to train a model directly from the dataset artifact. **This also logs evaluation**

▶ **Usage**

4: Save model checkpoints as artifacts

To enable saving and versioning checkpoints of your experiment, pass `--save_period n` with the base cammand, where `n` represents checkpoint interval. You can also log both the dataset and model checkpoints simultaneously. If not passed, only the final model will be logged

▶ **Usage**

5: Resume runs from checkpoint artifacts.

Any run can be resumed using artifacts if the `--resume` argument starts with `wandb-artifact://` prefix followed by the run path, i.e, `wandb-artifact://username/project/runid`. This doesn't require the model checkpoint to be present on the local system.

▶ Usage

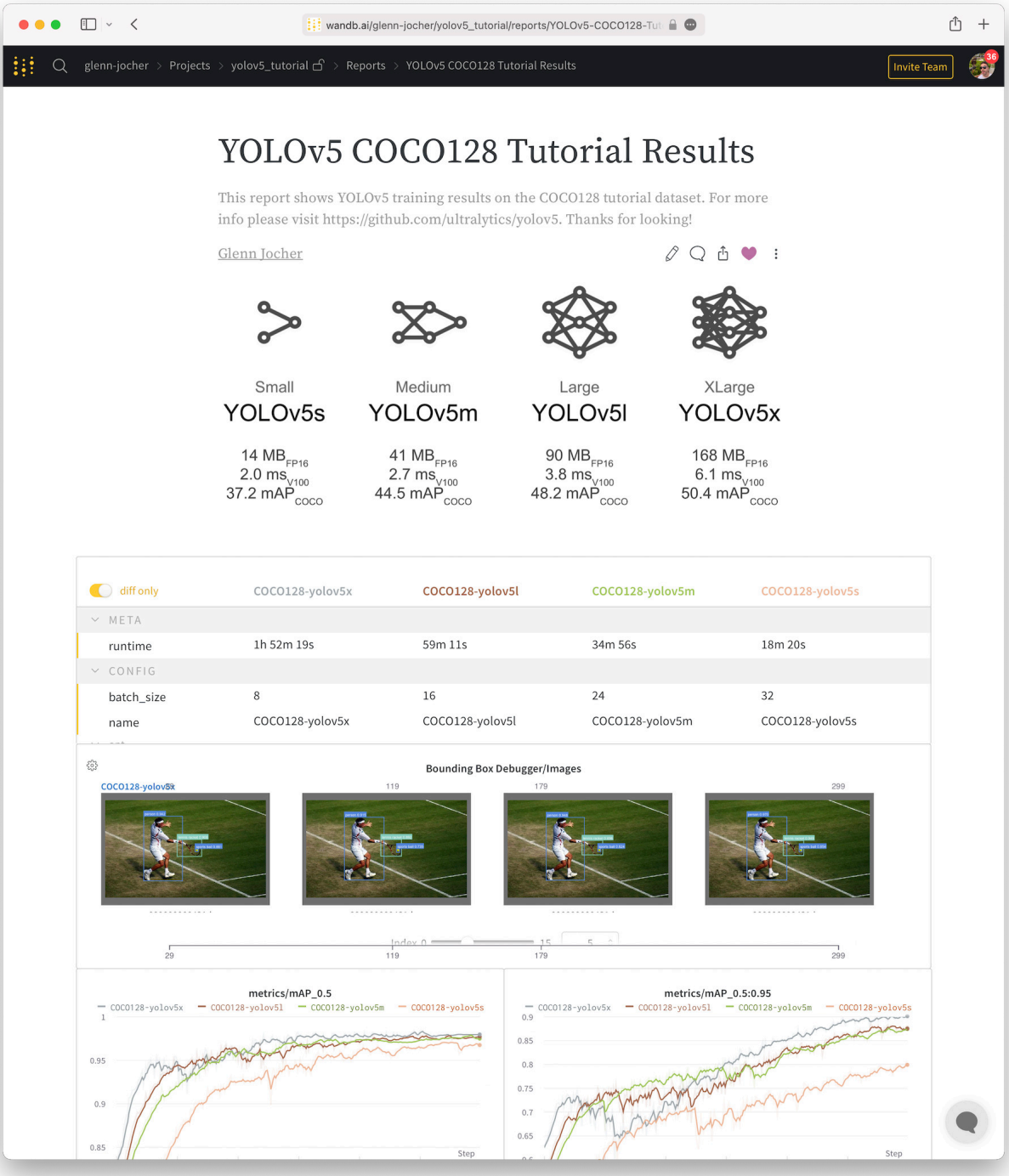
6: Resume runs from dataset artifact & checkpoint artifacts.

Local dataset or model checkpoints are not required. This can be used to resume runs directly on a different device The syntax is same as the previous section, but you'll need to lof both the dataset and model checkpoints as artifacts, i.e, set bot `--upload_dataset` or train from `_wandb.yaml` file and set `--save_period`

▶ Usage

Reports

W&B Reports can be created from your saved runs for sharing online. Once a report is created you will receive a link you can use to publically share your results. Here is an example report created from the COCO128 tutorial trainings of all four YOLOv5 models ([link](https://wandb.ai/glenn-jocher/yolov5_tutorial/reports/YOLOv5-COCO128-Tutorial-Results--VmlldzozMDI5OTY)).



Environments

YOLOv5 may be run in any of the following up-to-date verified environments (with all dependencies including CUDA/CUDNN, Python and PyTorch preinstalled):

- **Google Colab and Kaggle** notebooks with free GPU:

- **Google Cloud** Deep Learning VM. See GCP Quickstart Guide
- **Amazon** Deep Learning AMI. See AWS Quickstart Guide
- **Docker Image**. See Docker Quickstart Guide 

Status

If this badge is green, all YOLOv5 GitHub Actions Continuous Integration (CI) tests are currently passing. CI tests verify correct operation of YOLOv5 training (train.py), validation (val.py), inference (detect.py) and export (export.py) on MacOS, Windows, and Ubuntu every 24 hours and on every commit.