Weights & Biases Tracking

Module 5 of 7

Presenter: Vương Cường — Research Assistant, BAI Lab

What is Weights & Biases?

W&B = **Experiment tracking and ML operations platform**

Purpose

- **Hyperparameter logging** - Track all experiment configurations
- **Metric visualization** - Real-time training progress monitoring
- **Model comparison** - Compare multiple experiments systematically
- **Collaborative experiment management** - Team-wide experiment sharing

Why Essential for Research?

- **Professional experiment tracking** - No more spreadsheets or print statements
- **Reproducible research** - Complete experiment provenance
- **Efficient hyperparameter optimization** - Systematic parameter sweeps
- **Publication-ready visualizations** - High-quality plots and tables

Comparison with Traditional Tracking

Method	Visualization	Comparison	Collaboration	Reproducibility	Autom
Print statements	None	Manual	None	Poor	None
Spreadsheets	Basic	Manual	Limited	Poor	None
TensorBoard	Good	Limited	Poor	Good	Partial
MLflow	Good	Good	Good	Good	Good
W&B	**Excellent**	**Excellent**	**Excellent**	**Excellent**	**Excelle**

Professional Research Benefits

- **Automatic visualization** - No manual plot creation
- **Experiment comparison** - Side-by-side metrics and hyperparameters
- **Team collaboration** - Shared project dashboards
- **Hyperparameter sweeps** - Automated optimization runs
- **Model artifacts** - Version controlled model storage

Key Advantage: W&B transforms ad-hoc experimentation into systematic research methodology.

Basic Tracking Setup

```
# Core usage pattern covers most research needs
import wandb

# Initialize experiment (start of training script)
wandb.init(
    project="bailab-research",              # Project name
    name="transformer-experiment-1",        # Experiment name
    config={                                # Hyperparameters
        "learning_rate": 1e-3,
        "batch_size": 32,
        "epochs": 100,
        "model": "transformer",
        "dataset": "bailab-dataset-v1"
    }
)
```

Basic Tracking Setup (continued)

```
# Log metrics during training (most important)
for epoch in range(config.epochs):
    train_loss = train_epoch(model, train_loader)
    val_acc = validate(model, val_loader)

    wandb.log({
        "epoch": epoch,
        "train/loss": train_loss,
        "val/accuracy": val_acc,
        "learning_rate": optimizer.param_groups[0]['lr']
    })

# Save model artifacts
wandb.save("model.pth")
wandb.finish()  # End experiment
```

Advanced Features

```
# Log additional artifacts
wandb.log({
    "predictions": wandb.Table(data=predictions_df),
    "confusion_matrix": wandb.plot.confusion_matrix(y_true, y_pred),
    "sample_images": [wandb.Image(img) for img in sample_batch]
})

# Track gradients and model topology
wandb.watch(model, log="all")
```

Step 1: Setup W&B Account and Integration

```
cd base-research-repo

# Install W&B
uv pip install wandb

# Login to W&B (one-time setup)
wandb login
# Enter your API key from: https://wandb.ai/authorize
```

Step 2: Integrate W&B into Training Script

```python
# Add to src/training/train.py
import wandb
from pathlib import Path

def train_with_wandb():
    # Initialize W&B experiment
    wandb.init(
        project="bailab-base-research",
        name=f"experiment-{Path.cwd().name}",
    )

    # Simulate training loop
    for epoch in range(wandb.config.epochs):
        # Simulate training metrics
        train_loss = 1.0 - (epoch * 0.1)  # Decreasing loss
        val_accuracy = 0.5 + (epoch * 0.05)  # Increasing accuracy

        # Log metrics to W&B
        wandb.log({
            "epoch": epoch,
            "train/loss": train_loss,
            "val/accuracy": val_accuracy,
            "learning_rate": wandb.config.learning_rate
        })

        print(f"Epoch {epoch}: loss={train_loss:.3f}, acc={val_accuracy:.3f}")

    # Save final model
    wandb.save("final_model.pth")
    wandb.finish()

if __name__ == "__main__":
    train_with_wandb()
```

Step 3: Run Experiment and View Results

```
# Run training with W&B tracking
python src/training/train.py

# W&B will output a URL to view results:
# 🚀 View run at https://wandb.ai/bailab/bailab-base-research/runs/abc123

# Open the URL to see:
# - Real-time training curves
# - Hyperparameter values
# - System metrics (GPU, CPU, memory)
# - Model artifacts
```

Step 4: Compare Multiple Experiments

```
# Run multiple experiments with different hyperparameters
configs = [
    {"learning_rate": 1e-3, "batch_size": 32},
    {"learning_rate": 1e-4, "batch_size": 64},
    {"learning_rate": 5e-4, "batch_size": 16}
]

for i, config in enumerate(configs):
    wandb.init(
        project="bailab-base-research",
        name=f"hyperparameter-sweep-{i}",
        config=config,
        reinit=True
    )
    # Train with current config...
    wandb.finish()
```

Experiment Organization

Project Structure Best Practices

```
# Hierarchical organization
wandb.init(
    entity="bailab",                    # Organization name
    project="transformer-research",     # Broad research area
    group="ablation-study",             # Related experiments
    name="attention-heads-8",           # Specific experiment
    tags=["transformer", "attention", "ablation"]
)
```

Configuration Management

```
# Track everything that affects results
config = {
    # Model hyperparameters
    "model_name": "transformer",
    "hidden_size": 768,
    "num_layers": 12,

    # Training hyperparameters
    "learning_rate": 1e-3,
    "batch_size": 32,
    "epochs": 100,

    # Data configuration
    "dataset_version": "v1.2",
    "preprocessing": "standard",

    # Environment
    "gpu_type": "A100",
    "framework_version": "torch==2.1.0"
}
```

Performance and Resource Management

Efficient Logging Strategy

```
# Log at appropriate intervals
if step % 100 == 0:  # Every 100 steps, not every step
    wandb.log({"train/loss": loss}, step=step)

# Batch log multiple metrics
wandb.log({
    "train/loss": train_loss,
    "train/accuracy": train_acc,
    "val/loss": val_loss,
    "val/accuracy": val_acc
}, step=epoch)
```

Storage Considerations

- **Artifact storage limits** - Check W&B plan limits
- **Log frequency** - Don't log every single step
- **Large artifacts** - Use wandb.save() sparingly
- **Cleanup old experiments** - Remove failed/irrelevant runs

What We Covered

✅ **W&B fundamentals** - Professional experiment tracking platform
✅ **Advantages over ad-hoc tracking** - Systematic research methodology
✅ **Essential integration** - Core logging and artifact management
✅ **Lab-specific setup** - Team collaboration and organization
✅ **Best practices** - Performance, organization, and teamwork

Key Takeaways

1. **W&B transforms experimentation** from chaotic to systematic
2. **Automatic visualization** eliminates manual plot creation
3. **Team collaboration** enables shared research progress
4. **Reproducible experiments** through complete provenance tracking
5. **Publication-ready results** with professional visualizations

Impact on Research Workflow

- **Systematic experimentation** - No more lost experiments or forgotten hyperparameters
- **Efficient comparison** - Quickly identify best performing models
- **Team coordination** - Shared visibility into research progress
- **Paper preparation** - High-quality figures and tables ready for publication

Next Steps

➡️ **Module 6: Telegram Notifications** - Real-time experiment monitoring
➡️ Setup W&B for your current research project
➡️ Configure team-wide experiment tracking standards