analysisV6

August 20, 2025

1 HybridNet Evaluation

Version: 2.1.0 **Date:** Mon Aug 18th 2025

[23]:
```
import pandas as pd
import matplotlib.pyplot as plt

CSV_FILE = r'/Users/jbalkovec/Desktop/DR/experiments/model_logs/
 ↪metrics_expanded.csv'

BANDS = {
    "train_loss": [(0.00, 0.60, "green"), (0.60, 0.90, "yellow"), (0.90, 2.00,␣
 ↪"red")],
    "val_loss":   [(0.00, 0.60, "green"), (0.60, 0.90, "yellow"), (0.90, 2.00,␣
 ↪"red")],

    "val_dice_mean": [(0.00, 0.45, "red"), (0.45, 0.60, "yellow"), (0.60, 1.00,␣
 ↪"green")],
    "val_iou_mean":  [(0.00, 0.30, "red"), (0.30, 0.45, "yellow"), (0.45, 1.00,␣
 ↪"green")],

    "dice_c0": [(0.00, 0.25, "red"), (0.25, 0.45, "yellow"), (0.45, 1.00,␣
 ↪"green")],  # MA
    "dice_c1": [(0.00, 0.40, "red"), (0.40, 0.60, "yellow"), (0.60, 1.00,␣
 ↪"green")],  # HE
    "dice_c2": [(0.00, 0.40, "red"), (0.40, 0.60, "yellow"), (0.60, 1.00,␣
 ↪"green")],  # EX
    "dice_c3": [(0.00, 0.40, "red"), (0.40, 0.60, "yellow"), (0.60, 1.00,␣
 ↪"green")],  # SE

    "iou_c0": [(0.00, 0.15, "red"), (0.15, 0.35, "yellow"), (0.35, 1.00,␣
 ↪"green")],   # MA
    "iou_c1": [(0.00, 0.25, "red"), (0.25, 0.40, "yellow"), (0.40, 1.00,␣
 ↪"green")],   # HE
    "iou_c2": [(0.00, 0.25, "red"), (0.25, 0.40, "yellow"), (0.40, 1.00,␣
 ↪"green")],   # EX
```

```
    "iou_c3": [(0.00, 0.25, "red"), (0.25, 0.40, "yellow"), (0.40, 1.00,␣
 ↪"green")],   # SE
}

YLIMS = {
    "loss": (0.0, 1.2),
    "dice": (0.0, 1.0),
    "iou":  (0.0, 1.0),
}

def _apply_bands(ax, metric_key):
    if metric_key not in BANDS:
        return
    for low, high, color in BANDS[metric_key]:
        ax.axhspan(low, high, facecolor=color, alpha=0.15)

def _infer_ylim(metric_key):
    if "loss" in metric_key:
        return YLIMS["loss"]
    if "dice" in metric_key:
        return YLIMS["dice"]
    if "iou" in metric_key:
        return YLIMS["iou"]
    return None

def plot_with_bands(ax, df, metrics, title=None, labels=None, markers=True):
    if isinstance(metrics, str):
        metrics = [metrics]
    base_key = metrics[0]
    _apply_bands(ax, base_key)

    for i, m in enumerate(metrics):
        if m not in df.columns:
            raise ValueError(f"Metric '{m}' not found in df columns.")
        lbl = (labels[i] if labels and i < len(labels) else m)
        if markers:
            ax.plot(df["epoch"], df[m], marker="o", label=lbl)
        else:
            ax.plot(df["epoch"], df[m], label=lbl)

    ax.set_xlabel("Epoch")
    ax.set_ylabel(base_key)
    if title:
        ax.set_title(title)
    ax.grid(True, alpha=0.3)
    ax.legend()
```

```
    ylim = _infer_ylim(base_key)
    if ylim is not None:
        ax.set_ylim(*ylim)

def plot_two_panel_report(df, loss_title="Loss (Focal Tversky)",␣
 ↪dice_title="Val Dice (mean)"):
    fig, axes = plt.subplots(1, 2, figsize=(14, 5))
    plot_with_bands(
        axes[0],
        df,
        metrics=["train_loss", "val_loss"],
        title=loss_title,
        labels=["Train", "Val"],
    )
    plot_with_bands(
        axes[1],
        df,
        metrics=["val_dice_mean"],
        title=dice_title,
        labels=["Val"],
    )
    plt.tight_layout()
    plt.show()

def plot_two_panel_iou_and_dice(df, iou_title="Val IoU (mean)", dice_title="Val␣
 ↪Dice (mean)"):
    fig, axes = plt.subplots(1, 2, figsize=(14, 5))
    plot_with_bands(axes[0], df, metrics=["val_iou_mean"], title=iou_title,␣
 ↪labels=["Val"])
    plot_with_bands(axes[1], df, metrics=["val_dice_mean"], title=dice_title,␣
 ↪labels=["Val"])
    plt.tight_layout()
    plt.show()

def plot_per_class_grid(df, base="dice"):
    assert base in ("dice", "iou")
    cols = [f"{base}_c0", f"{base}_c1", f"{base}_c2", f"{base}_c3"]
    titles = ["MA", "HE", "EX", "SE"]

    fig, axes = plt.subplots(2, 2, figsize=(14, 10))
    for i, (col, t) in enumerate(zip(cols, titles)):
        r, c = divmod(i, 2)
        plot_with_bands(axes[r, c], df, metrics=[col], title=f"{base.upper()} -␣
 ↪{t}", labels=[t])
    plt.tight_layout()
    plt.show()
```

[24]:
```
df = pd.read_csv(CSV_FILE)
```

1.1 Best Epoch

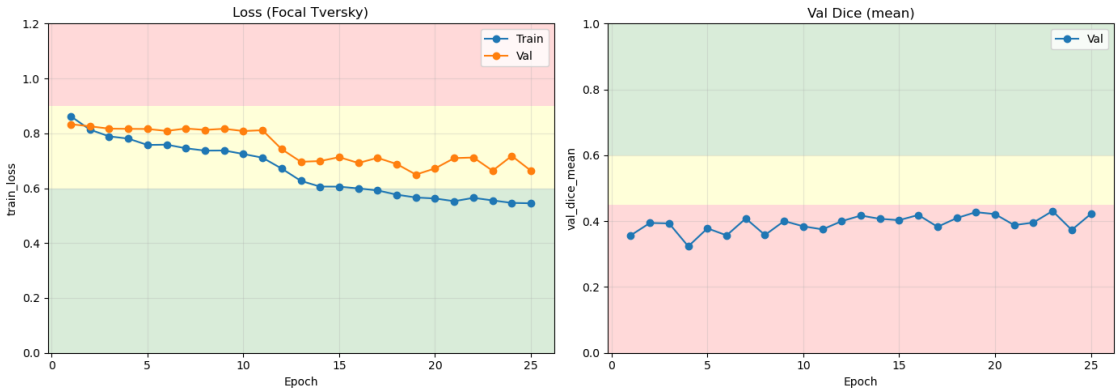
[25]:
```
best_idx = df['val_dice_mean'].idxmax()
best_row = df.loc[best_idx]
print(f"Best epoch: {best_row['epoch']}")
print(best_row[['train_loss','val_loss','val_dice_mean','val_iou_mean']])
print("\nPer-class Dice:", [best_row[f'dice_c{i}'] for i in range(4)])
print("Per-class IoU :", [best_row[f'iou_c{i}'] for i in range(4)])
```

```
Best epoch: 23.0
train_loss       0.555480
val_loss         0.663624
val_dice_mean    0.430738
val_iou_mean     0.380597
Name: 22, dtype: float64

Per-class Dice: [0.1881998032331466, 0.4289990067481994, 0.440678983926773,
0.6650723218917847]
Per-class IoU : [0.1752179712057113, 0.3427709937095642, 0.3655824959278106,
0.6388179063796997]
```

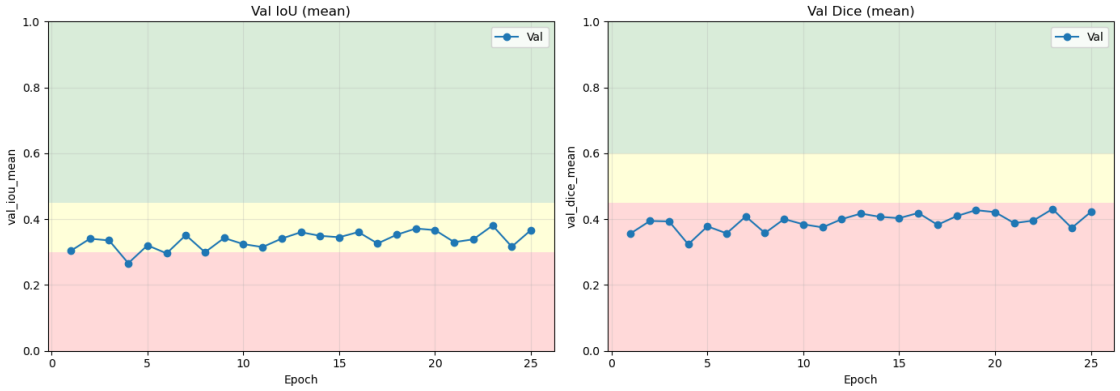
1.2 Loss Curves

[26]:
```
plot_two_panel_report(df, loss_title="Loss (Focal Tversky)", dice_title="Val␣
↪Dice (mean)")
```



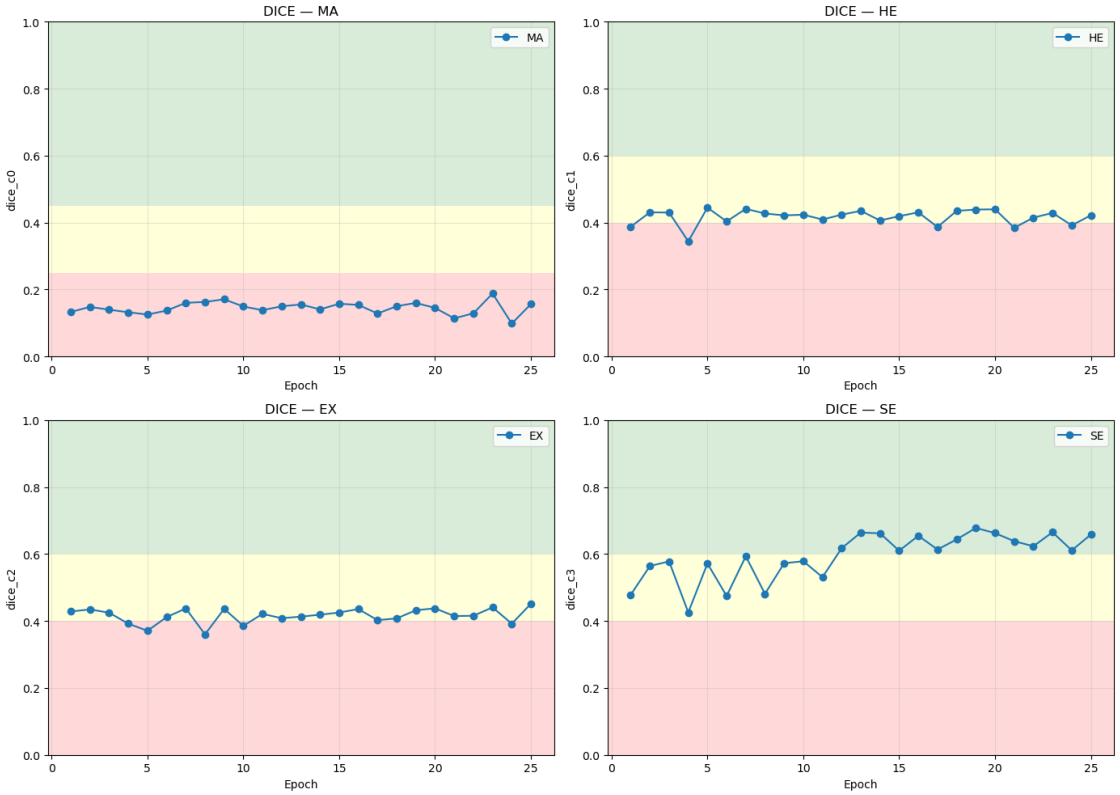
1.3 IoU vs Dice

[27]:
```
plot_two_panel_iou_and_dice(df, iou_title="Val IoU (mean)", dice_title="Val␣
↪Dice (mean)")
```



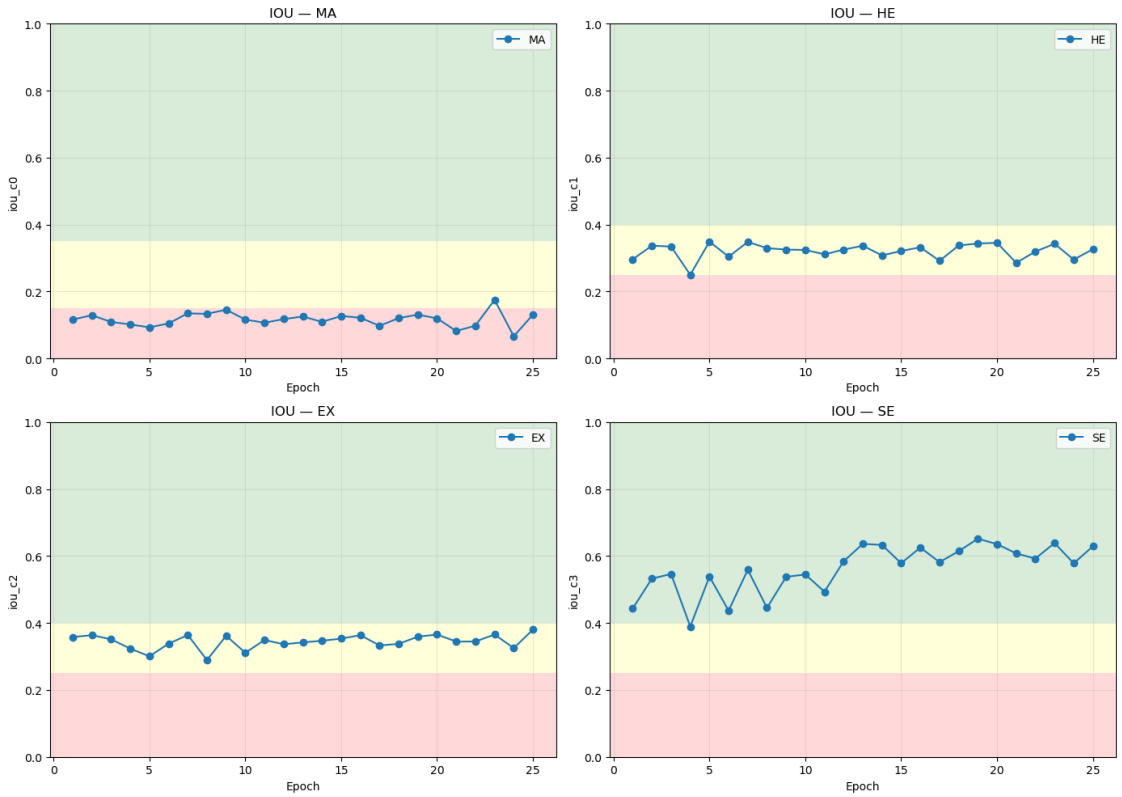
1.4 Per-class Dice

[28]:
```
plot_per_class_grid(df, base="dice")
```



1.5 Per-class IoU

[29]:
```
plot_per_class_grid(df, base="iou")
```



1.6 Summary Table (Last + Best)

[30]:
```
summary = pd.DataFrame({
    "Epoch": [int(best_row['epoch']), int(df['epoch'].iloc[-1])],
    "Val Dice Mean": [best_row['val_dice_mean'], df['val_dice_mean'].iloc[-1]],
    "Val IoU Mean": [best_row['val_iou_mean'], df['val_iou_mean'].iloc[-1]],
    "Loss": [best_row['val_loss'], df['val_loss'].iloc[-1]]
})
print("\nSummary (best vs last):\n", summary)
```

```
Summary (best vs last):
    Epoch  Val Dice Mean  Val IoU Mean      Loss
0      23       0.430738      0.380597  0.663624
1      25       0.421981      0.366671  0.663712
```