Yaml Configuration File

```yaml
attention_dropout: 0.1
batch_size: 128
betas:
- 0.9
- 0.95
checkpoints_path: ./models/dt
clip_grad: 100.0
embedding_dim: 128
embedding_dropout: 0.1
env_name: hopper-medium-v2
episode_len: 1000
eval_episodes: 10
eval_every: 1
eval_seed: 42
group: DT-D4RL
learning_rate: 0.0003
max_action: 1.0
name: DT
num_epochs: 50
num_heads: 8
num_layers: 6
num_workers: 4
project: CORL
residual_dropout: 0.1
reward_scale: 0.001
seq_len: 20
target_returns:
- 3600.0
- 1800.0
train_seed: 10
warmup_ratio: 0.001
weight_decay: 0.0001
```

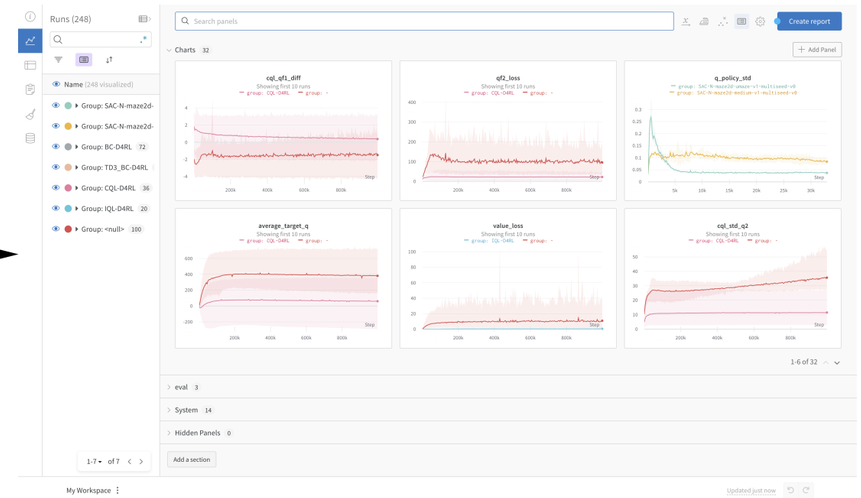
environment params
algorithm params

Single-File Implementation

```python
import gym
import torch


@dataclass
class TrainConfig:
    # ...


# general utils
def set_seed(seed: int, env: gym.Env, deterministic: bool):
    # ...


def load_trajectories(env: str) -> Tuple[List[Dict[str, np.ndarray]], Dict[str, Any]]:
    # ...
    return trajectories, info


# Decision Transformer implementation
class DecisionTransformer(nn.Module):
    def forward(
        self, states: Tensor, actions: Tensor, returns_to_go: Tensor, time_steps: Tensor,
    ) -> Tensor:
        # ...
        return predicted_actions


# Training and evaluation logic
@torch.no_grad()
def eval_rollout(model: DecisionTransformer, env: gym.Env) -> Tuple[float, float]:
    # ...
    return episode_return, episode_len


def train(config: TrainConfig):
    set_seed(...)
    dataset = Dataset(load_trajectories(...))
    model = DecisionTransformer(...)
    for epoch in range(config.num_epochs):
        for batch in dataset:
            # train model on batch
        if epoch % config.eval_every == 0:
            performance = eval_rollout(...)

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

AWAC / BC / CQL / DT
EDAC / IQL / SAC-N / TD3+BC

Experiment Tracking Log



Wandb logs
Wandb reports

```
python dt.py --config=cfg/dt-hopper.yaml --logdir=logs/dt-hopper --num-epochs=50
```