# Grid search

import torch

import torch.nn as nn

import torch.optim as optim

from hyperopt import fmin, tpe, hp, Trials, STATUS\_OK

import numpy as np

import matplotlib.pyplot as plt

# Определяем целевую функцию для Hyperopt

def objective(params):

# Запускаем pipe1 с параметрами

rmse = pipe1(

data=data, # Ваш DataFrame

net=net,

feat=5,

bs=params["bs"],

lr0=params["lr0"],

lr1=params["lr1"],

eps=150,

device=device

)

return {

"loss": rmse,

"status": STATUS\_OK

}

# Пространство гиперпараметров

space = {

"bs": hp.choice("bs", [0.02, 0.04, 0.08, 0.1, 0.3]),

"lr0": hp.choice("lr0", [0.01, 0.1, 0.5, 0.05, 1]),

"lr1": hp.choice("lr1", [0.0001, 0.001, 0.01, 0.1]),

}

# Запуск оптимизации

trials = Trials() # Для хранения истории испытаний

best = fmin(

fn=objective,

space=space,

algo=tpe.suggest, # Байесовская оптимизация с TPE

max\_evals=20, # Количество испытаний (можно увеличить)

trials=trials

)

# Преобразование индексов в значения

param\_values = {

"bs": [0.02, 0.08, 0.1, 0.3],

"lr0": [0.01, 0.1, 0.5, 0.05, 1],

"lr1": [0.0001, 0.001, 0.01, 0.1],

}

best\_params = {key: param\_values[key][best[key]] for key in best}

print("\nЛучшие параметры:")

print(best\_params)

# Лучший результат

best\_val\_mse = min([trial["result"]["loss"] for trial in trials.trials])

print(f"Best Validation MSE: {best\_val\_mse:.4f}")

# Визуализация истории оптимизации

losses = [trial["result"]["loss"] for trial in trials.trials]

plt.figure(figsize=(10, 6))

plt.plot(range(len(losses)), losses, marker='o')

plt.title("Validation MSE по испытаниям")

plt.xlabel("Испытание")

plt.ylabel("Validation MSE")

plt.grid(True)

plt.show()