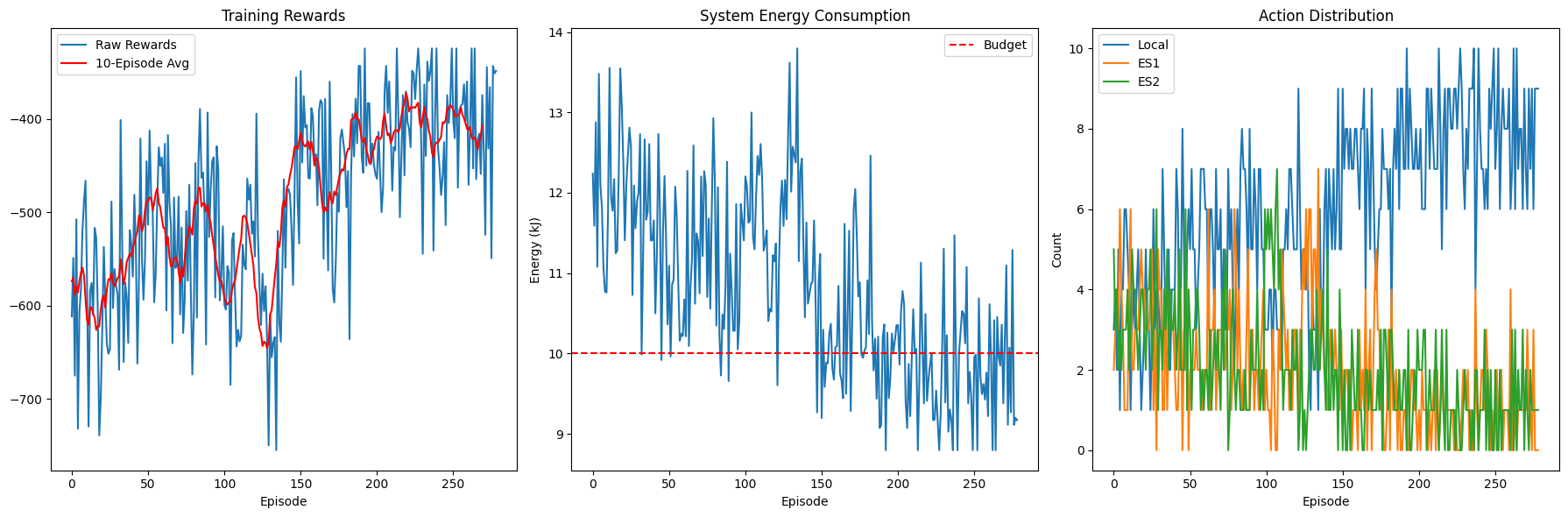
**DQN Project Graphs**

**New reward and improved parameters**

****

**Let's analyze the energy costs using the formulas from the MECEnvironment class:**

**For Local Processing:**

**energy\_needed = (md\_energy\_local \* task\_size \* 1e9) / (md\_compute \* 1e9)**

**For Edge Processing:**

**tx\_energy = md\_energy\_tx \* task\_size \* 2.5**

**process\_energy = (es\_energy\_process \* task\_size \* 1e9) / (es\_compute \* 1e9)**

**total\_energy = tx\_energy + process\_energy**

**250 MB Task Calculations**

| **Parameter** | **Local Processing Range** | **Edge Processing Range** |
| --- | --- | --- |
| **MD Energy Local** | **4.0-10.0 J/GB** | **-** |
| **MD Compute** | **1.5-3.5 GHz** | **-** |
| **MD Energy TX** | **-** | **0.6-1.2 J/MB** |
| **ES Energy Process** | **-** | **35.0-65.0 J/GB** |
| **ES Compute** | **-** | **25.0-35.0 GHz** |

**Local Energy:**

**Min: (4.0 \* 250) / 3.5 = 285.7 J**

**Max: (10.0 \* 250) / 1.5 = 1666.7 J**

**Edge Energy:**

**TX Min: 0.6 \* 250 \* 2.5 = 375 J**

**TX Max: 1.2 \* 250 \* 2.5 = 750 J**

**Processing Min: (35.0 \* 250) / 35.0 = 250 J**

**Processing Max: (65.0 \* 250) / 25.0 = 650 J**

**Total Min: 375 + 250 = 625 J**

**Total Max: 750 + 650 = 1400 J**

**10 MB Task Calculations**

**Local Energy:**

**Min: (4.0 \* 10) / 3.5 = 11.4 J**

**Max: (10.0 \* 10) / 1.5 = 66.7 J**

**Edge Energy:**

**TX Min: 0.6 \* 10 \* 2.5 = 15 J**

**TX Max: 1.2 \* 10 \* 2.5 = 30 J**

**Processing Min: (35.0 \* 10) / 35.0 = 10 J**

**Processing Max: (65.0 \* 10) / 25.0 = 26 J**

**Total Min: 15 + 10 = 25 J**

**Total Max: 30 + 26 = 56 J**

**Energy Comparison Table**

| **Task Size** | **Processing Type** | **Min Energy (J)** | **Max Energy (J)** |
| --- | --- | --- | --- |
| **250 MB** | **Local** | **285.7** | **1666.7** |
| **250 MB** | **Edge** | **625.0** | **1400.0** |
| **10 MB** | **Local** | **11.4** | **66.7** |
| **10 MB** | **Edge** | **25.0** | **56.0** |

**Key Observations:**

1. **Small tasks (10MB) are more energy-efficient to process locally**
2. **Medium tasks (250MB) show complex tradeoffs:**
   * **Edge processing has higher minimum energy but more predictable costs**
   * **Local processing can be more efficient for high-performance MDs**
3. **Energy costs scale super-linearly with task size due to:**
   * **Transmission energy multiplier (2.5×)**
   * **Processing energy's inverse relationship with compute speed**

**This explains why the agent needs to learn context-aware offloading strategies based on device capabilities and current battery states.**

**The hyperparameter sets are as follows:**

**HP\_CONFIGS = {**

**"conservative": {**

**'memory\_size': 50000,**

**'batch\_size': 128,**

**'gamma': 0.95,**

**'epsilon\_decay': 0.98,**

**'epsilon\_min': 0.1,**

**'learning\_rate': 0.001,**

**'tau': 0.1,**

**'network\_layers': [128, 64],**

**'energy\_penalty\_scale': 0.002,**

**'prioritization\_beta': 0.5**

**},**

**"aggressive": {**

**'memory\_size': 200000,**

**'batch\_size': 512,**

**'gamma': 0.99,**

**'epsilon\_decay': 0.99,**

**'epsilon\_min': 0.2,**

**'learning\_rate': 0.0003,**

**'tau': 0.05,**

**'network\_layers': [512, 256],**

**'efficiency\_bonus': 3.0,**

**'task\_reward': 10.0**

**},**

**"balanced": {**

**'memory\_size': 100000,**

**'batch\_size': 256,**

**'gamma': 0.97,**

**'epsilon\_decay': 0.995,**

**'epsilon\_min': 0.05,**

**'learning\_rate': 0.0005,**

**'tau': 0.12,**

**'network\_layers': [256, 128],**

**'energy\_penalty\_scale': 0.0015,**

**'budget\_penalty': 12.0**

**},**

**"hp6": {**

**'memory\_size': 30000,**

**'batch\_size': 64,**

**'gamma': 0.93,**

**'epsilon\_decay': 0.97,**

**'epsilon\_min': 0.2,**

**'learning\_rate': 0.002,**

**'tau': 0.25,**

**'network\_layers': [64, 32],**

**'energy\_penalty\_scale': 0.001,  # Reduced penalty**

**'reward\_components': {**

**'task': 9.0,**

**'efficiency': 1.5,**

**'budget': 8.0**

**}**

**},**

**"hp5": {**

**'memory\_size': 150000,**

**'batch\_size': 384,**

**'gamma': 0.98,**

**'epsilon\_decay': 0.998,  # Slower exploration decay**

**'epsilon\_min': 0.15,**

**'learning\_rate': 0.0002,**

**'tau': 0.08,**

**'network\_layers': [512, 256, 128],**

**'prioritization\_alpha': 0.7,  # More aggressive prioritization**

**'prioritization\_beta': 0.5**

**},**

**"hp4": {**

**'memory\_size': 75000,**

**'batch\_size': 192,**

**'gamma': 0.96,**

**'epsilon\_decay': 0.985,**

**'epsilon\_min': 0.05,**

**'learning\_rate': 0.0007,**

**'tau': 0.15,**

**'task\_reward': 10.0,  # Higher task completion reward**

**'efficiency\_bonus': 3.0,**

**'network\_layers': [384, 192]**

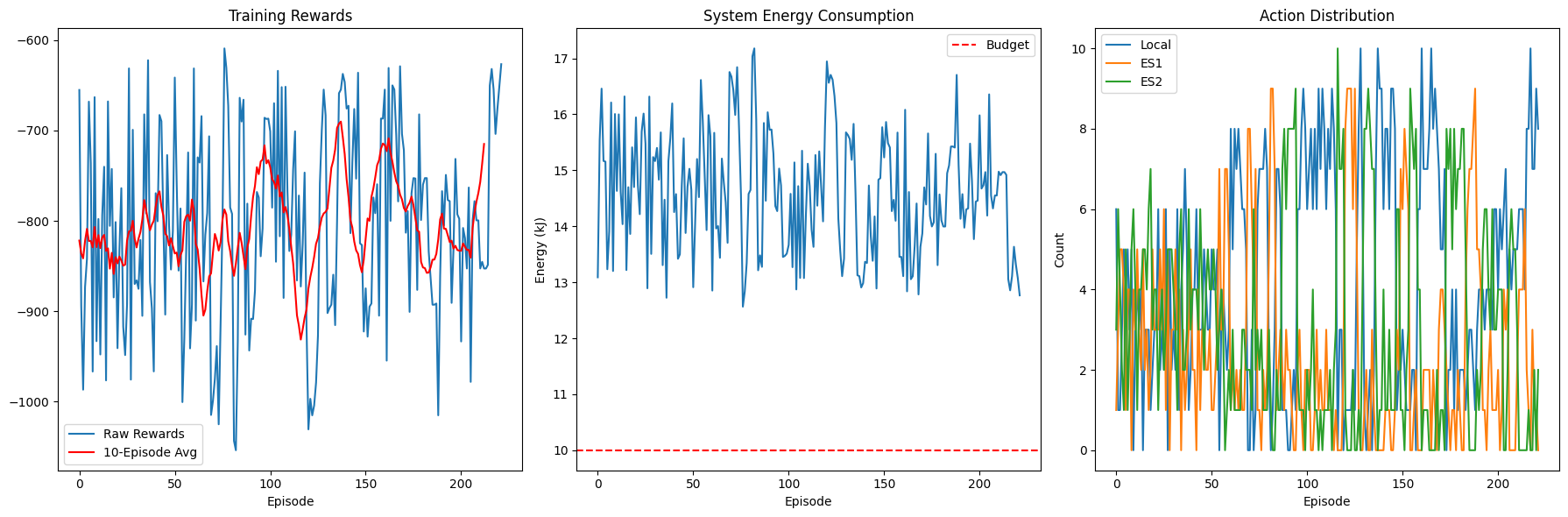
**}**

**}**

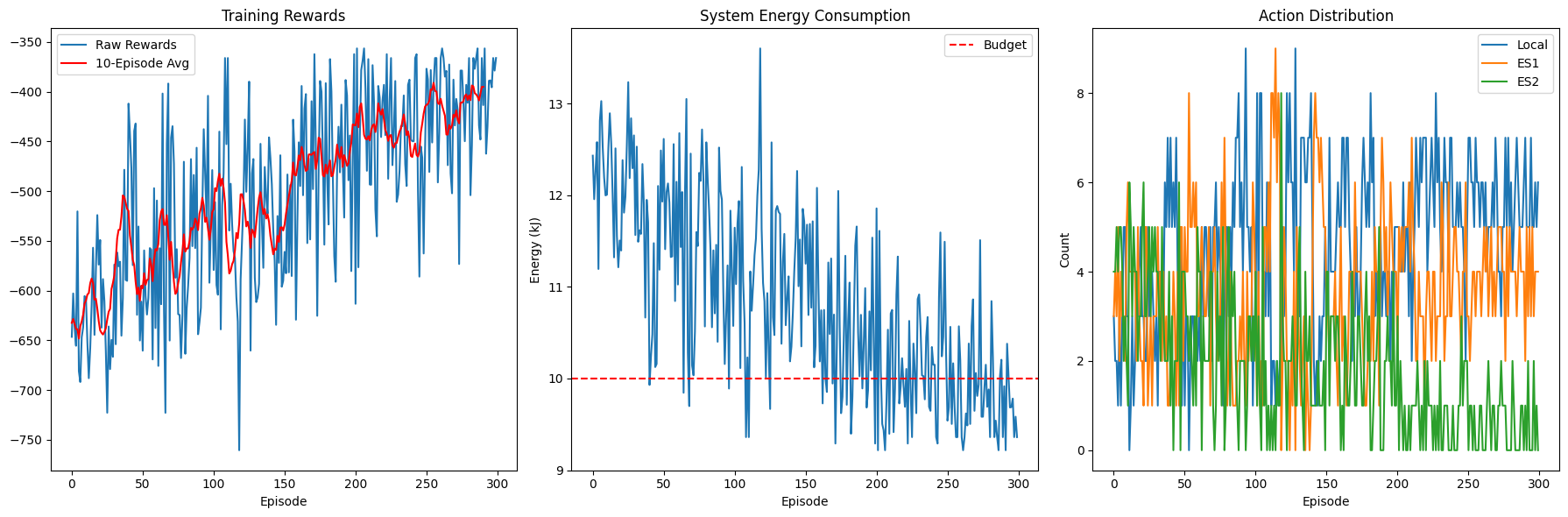
**Hyperparameter Tuning  
  
  
  
  
(i) Set A(Conservative)**

****

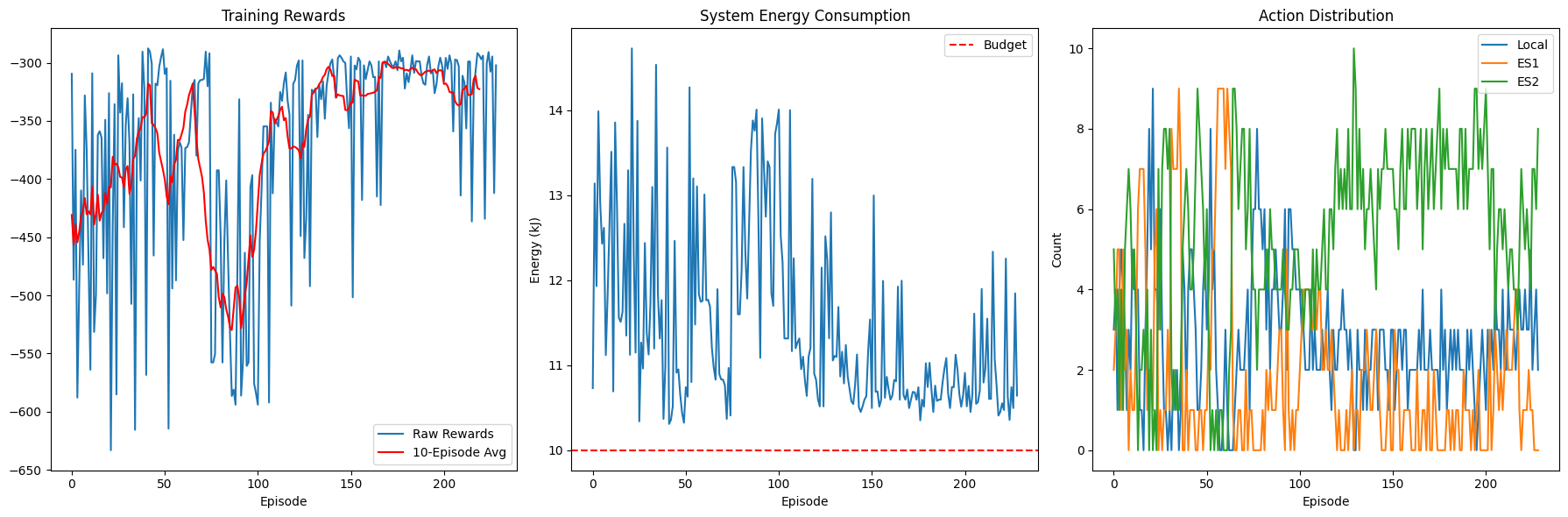
**(ii)Set B(Aggressive)**

****

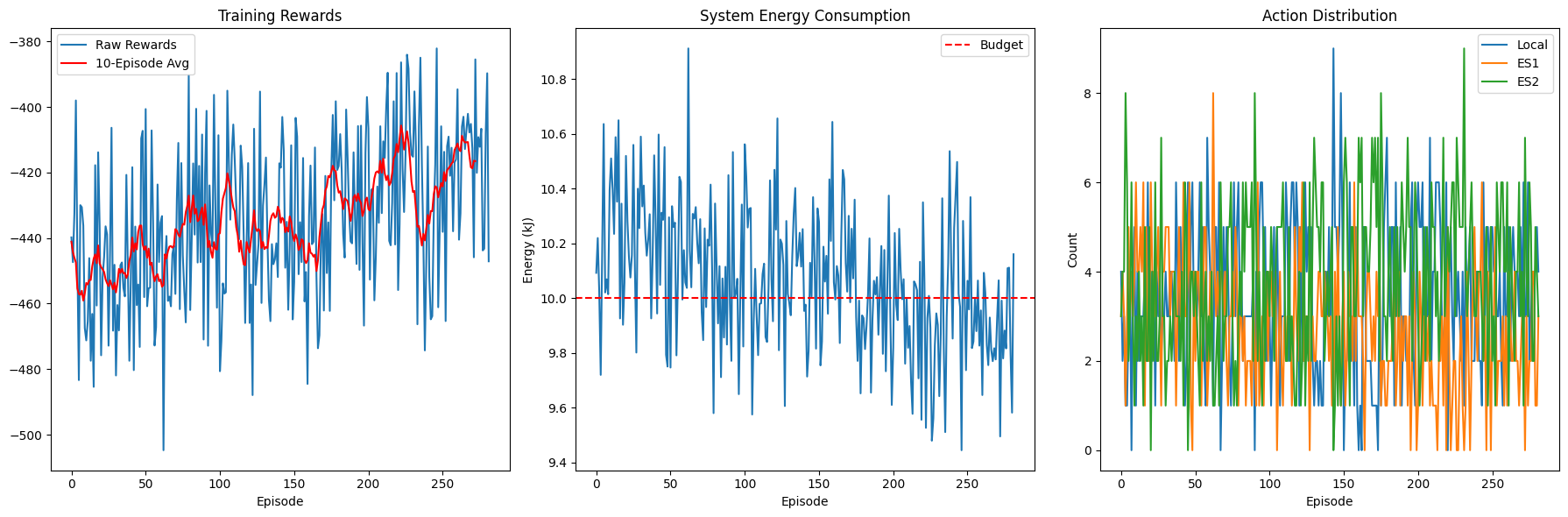
**(iii)Set c(balanced)**

****

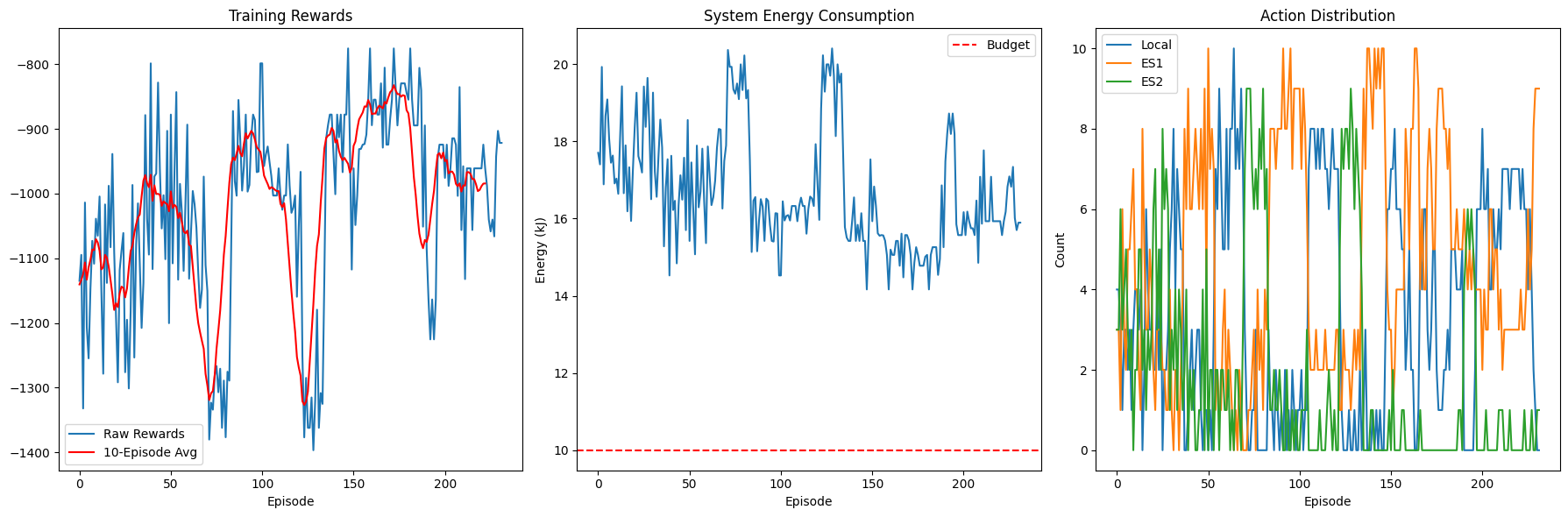
**(iv)Hp 6**

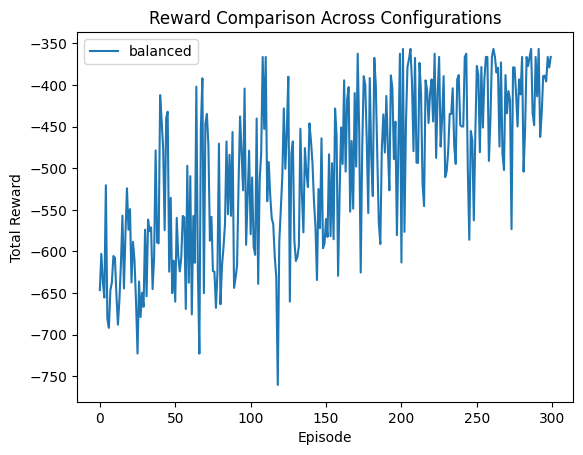
****

**(v)Hp 5**

****

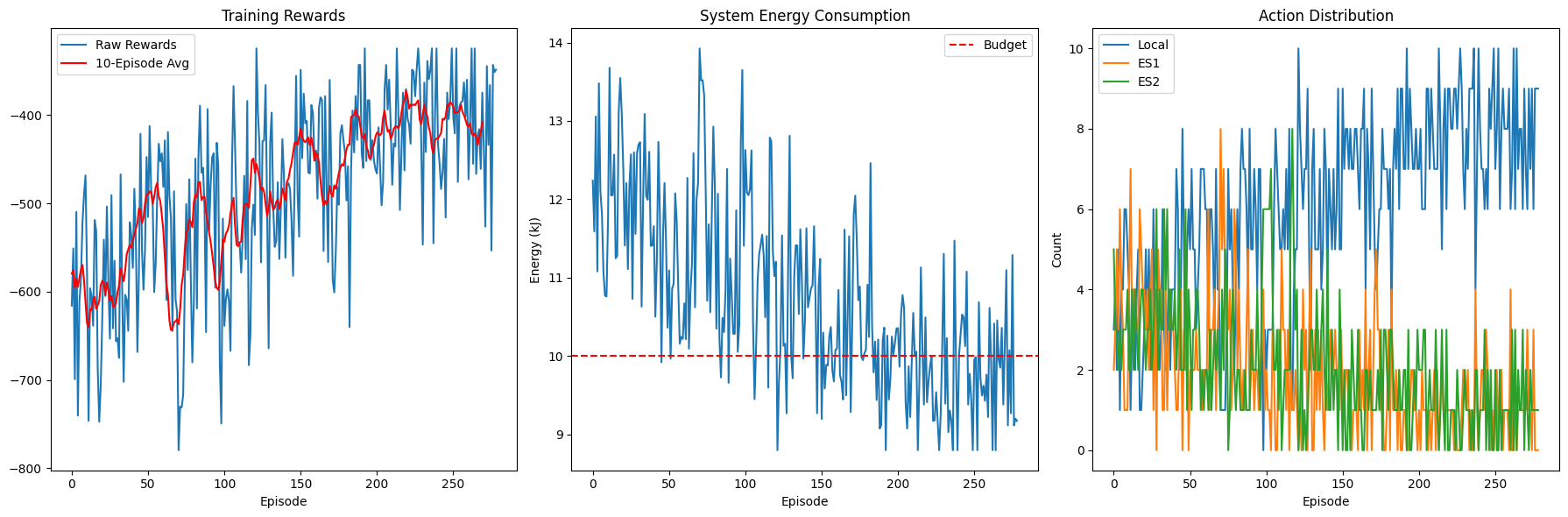
**(vi)Hp 6**

**  
  
  
The best results were seen in the balanced hyperparameter set.**

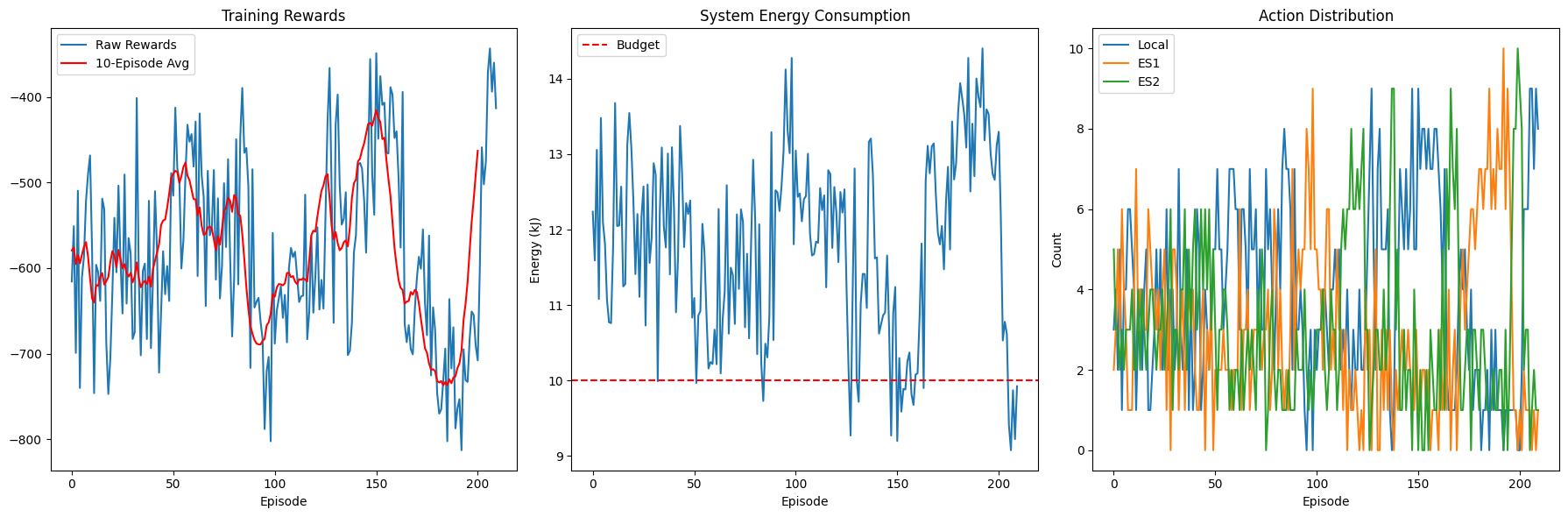
****

**New Model with few files(10)**

**(i)Learning rate =0.0005**

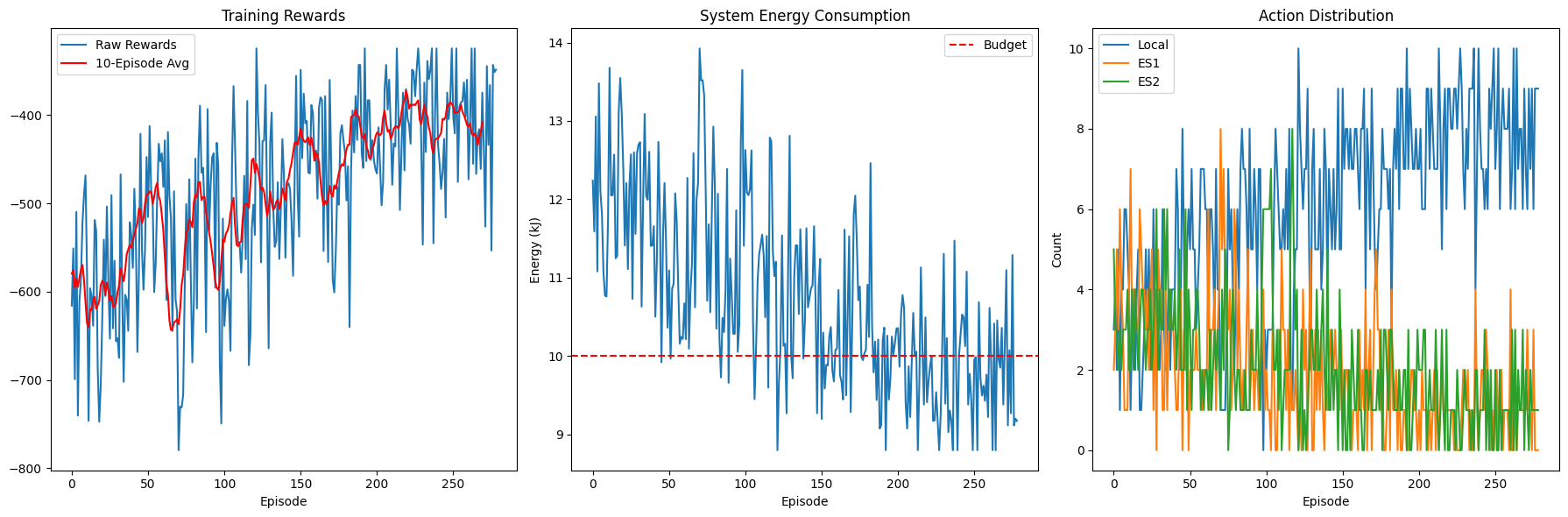
****

**(ii)Learning rate =0.0001**

****

**New model in different Environments**

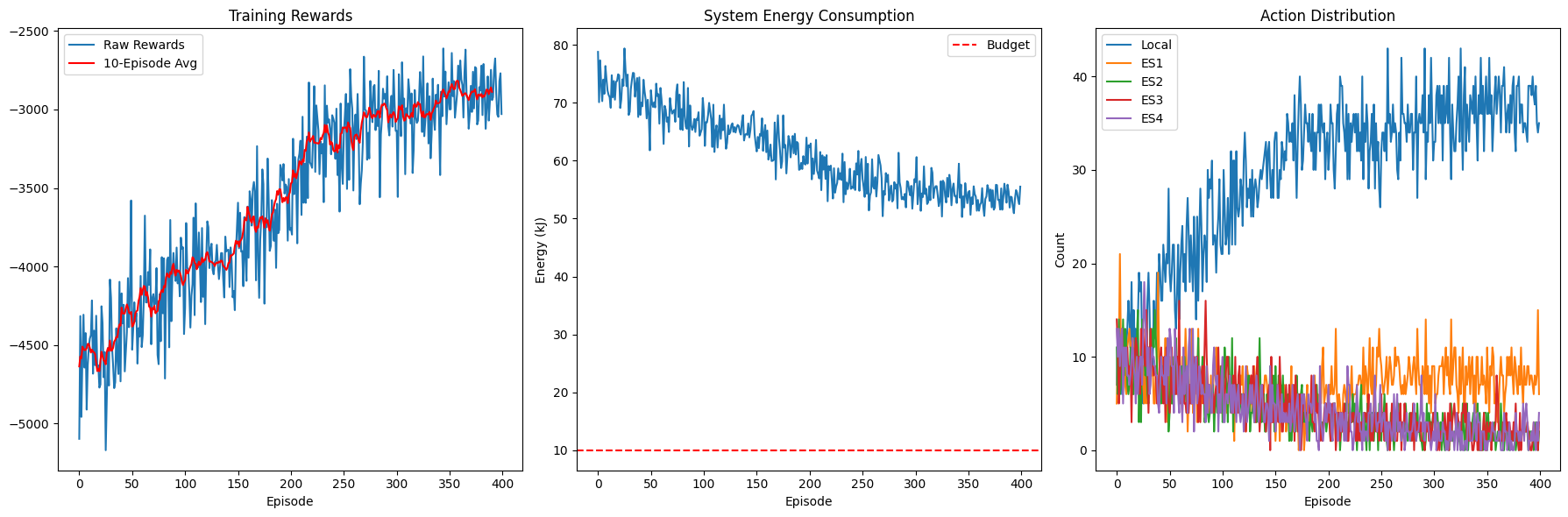
**(i)md=3,es=2,num\_tasks=10**

****

**(ii)md=4,es=1,num\_tasks=20**

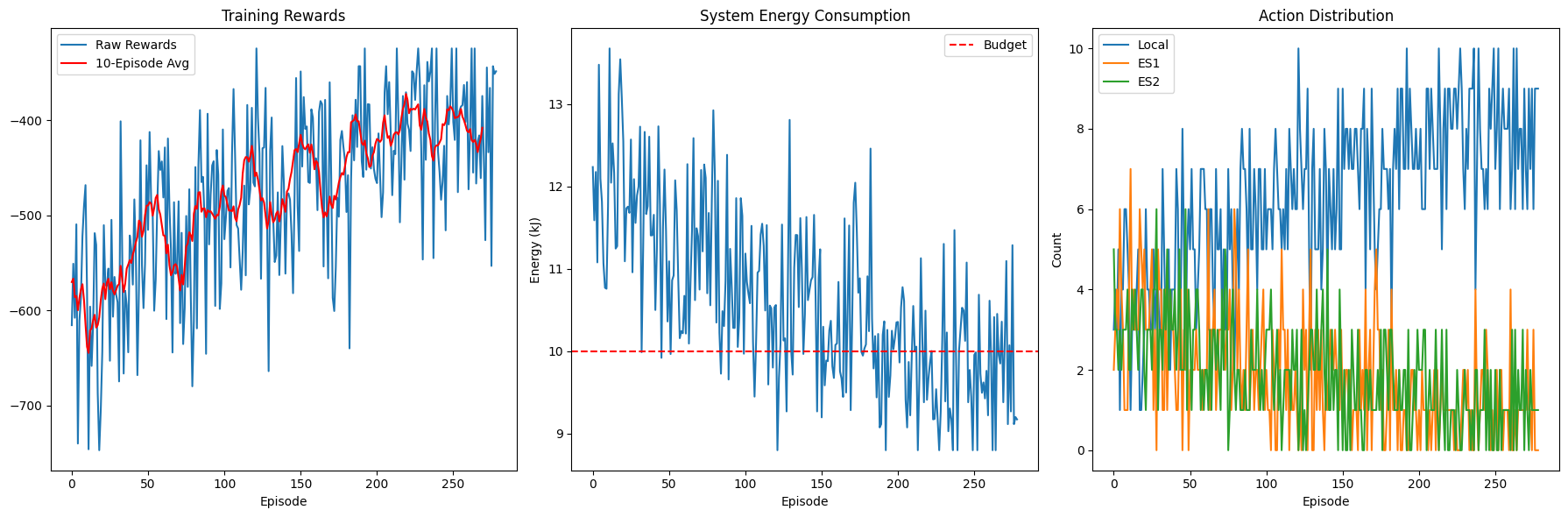
****

(iii)md=10,es=4,num\_tasks=50

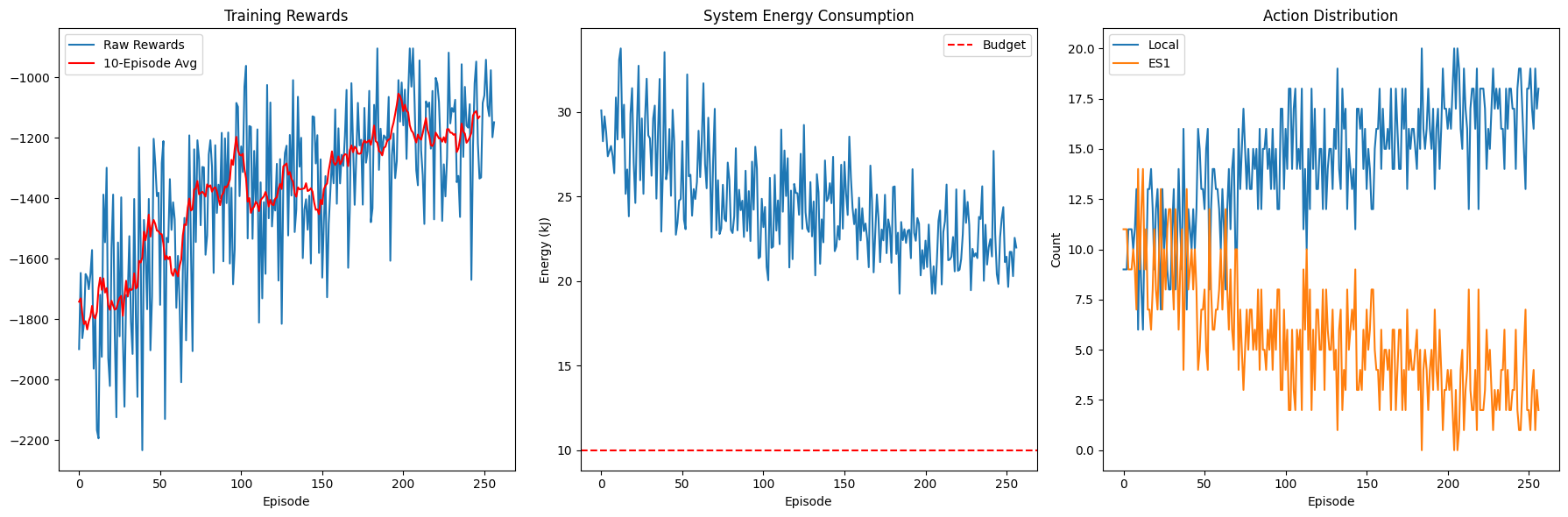


**Traditional Model**

(i)md=3,es=2,num\_tasks=10



(ii)md=4,es=1,num\_tasks=20



(iii)md=10,es=4,num\_tasks=50

