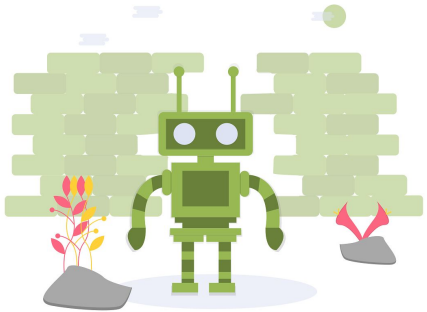
Training mit ML-Agents und Auswertung
- Praktikum -

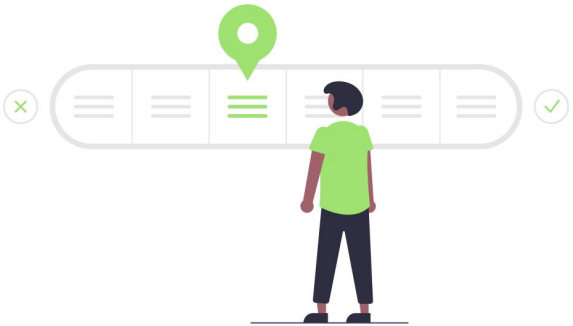
Projektgruppe 642 SoSe 2021



Marco Pleines & Roman Kalkreuth | Dortmund, 23.04.2021

Agenda

1. Hyperparameter
2. Experimente
3. Ergebnisse visualisieren



1. Hyperparameter

```yaml
environment:
  type: "Unity"
  name: "Hallway"
  frame_skip: 1
  last_action_to_obs: False
  last_reward_to_obs: False
  obs_stacks: 1
  grayscale: False
  resize_vis_obs: [84, 84]
  reset_params:
    start-seed: 0
    num-seeds: 100
```

```yaml
trainer:
  algorithm: "PPO"
  gamma: 0.99
  lamda: 0.95
  updates: 2000
  epochs: 3
  n_workers: 16
  worker_steps: 512
  n_mini_batch: 8
  resume_at: 0
```

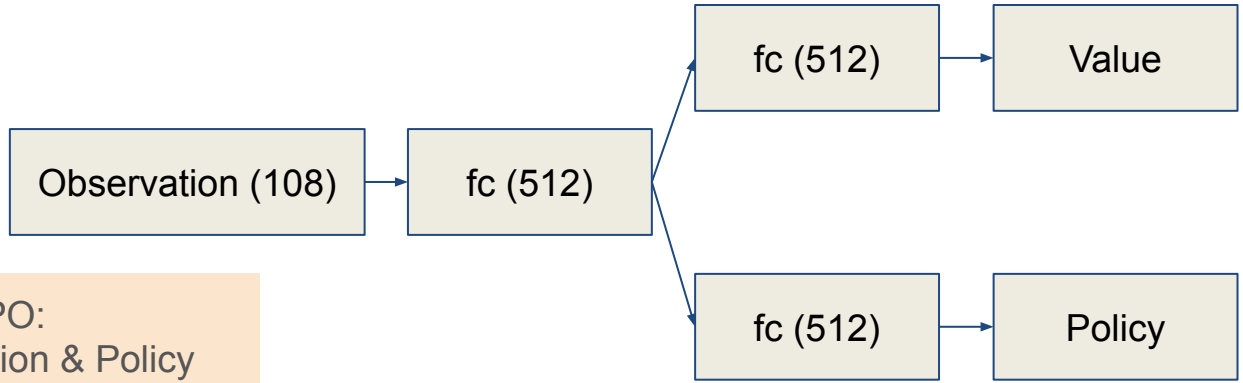
```yaml
learning_rate_schedule:
    initial: 3.0e-4
    final: 3.0e-6
    power: 1.0
    max_decay_steps: 2000
  beta_schedule:
    initial: 0.01
    final: 0.01
    power: 1.0
    max_decay_steps: 2000
  clip_range_schedule:
    initial: 0.2
    final: 0.01
    power: 1.0
    max_decay_steps: 2000
```

Hyperparameter

- Hyperparameter sind jene Parameter, welche der Entwickler vor dem Training einstellt (z.B. learning rate)
- Hyperparameter beeinflussen das Verhalten des Trainingsalgorithmus
- Sehr viele andere Parameter werden innerhalb eines Deep Learning Lernprozesses optimiert
  (z.B. Gewichtungen eines neuronalen Netzes)

Hyperparameter

- Hyperparameter und Modellarchitekturen können je nach Trainingsaufgabe sehr unterschiedlich ausfallen



Case PPO:
Shall Value Function & Policy share parameters?

Hyperparameter Tuning

Ansätze zur Optimierung der Hyperparameter:

- Vergleichbare Trainingsaufgaben
- Trial-and-error
- Grid Search
- Random Search
- Population-based training
- ...

2. Experimente

Das A und O von DRL Experimenten

- Einzelne Durchläufe eines Experiments können sehr unterschiedlich ausfallen
  - z.B. extrem gut oder extrem schlecht (Outlier?)

- Daher: Experimente **X-Mal Wiederholen**
  - Je nach verfügbaren Ressourcen

Evaluierung des trainierten Agenten

Testen auf Generalization und Overfitting eindämmen:

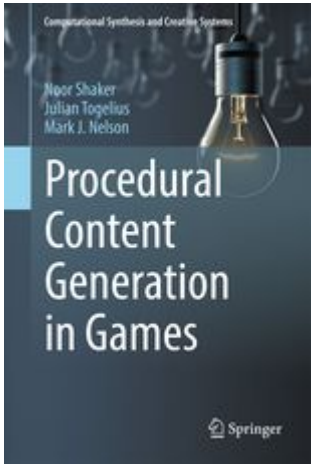
- Den trainierten Agenten in Umweltinstanzen agieren lassen, welche er nicht im Training gesehen hat.

- Trainings Set: Menge an Seeds nur für das Training
- Test Set: Menge an Seeds nur für die Evaluation

- Nützlich: Procedural Content Generation 

PCG Literatur

Procedural Content Generation in Games
Shaker & Togelius & Nelson
Springer



Deep Learning for Procedural Content Generation
Liu et al. 2020
https://arxiv.org/abs/2010.04548

Reproduzierbarkeit

Manchmal nicht ganz so einfach, aber setzt einen festen Seed

Python: random.seed(seed)

Numpy: np.random.seed(seed)

PyTorch: torch.manual_seed(seed)

Tensorflow: tf.random.set_seed(seed)

Gym: env.seed(seed)

torch.use_deterministic_algorithms(True)

Während des Trainings Checkpoints anlegen!

- Implementiert stets ein Checkpoint-System um Trainingszwischenstände zu speichern
- Trainingsvorgänge können abbrechen
- Checkpoints können für die Evaluierung gut verwendet werden

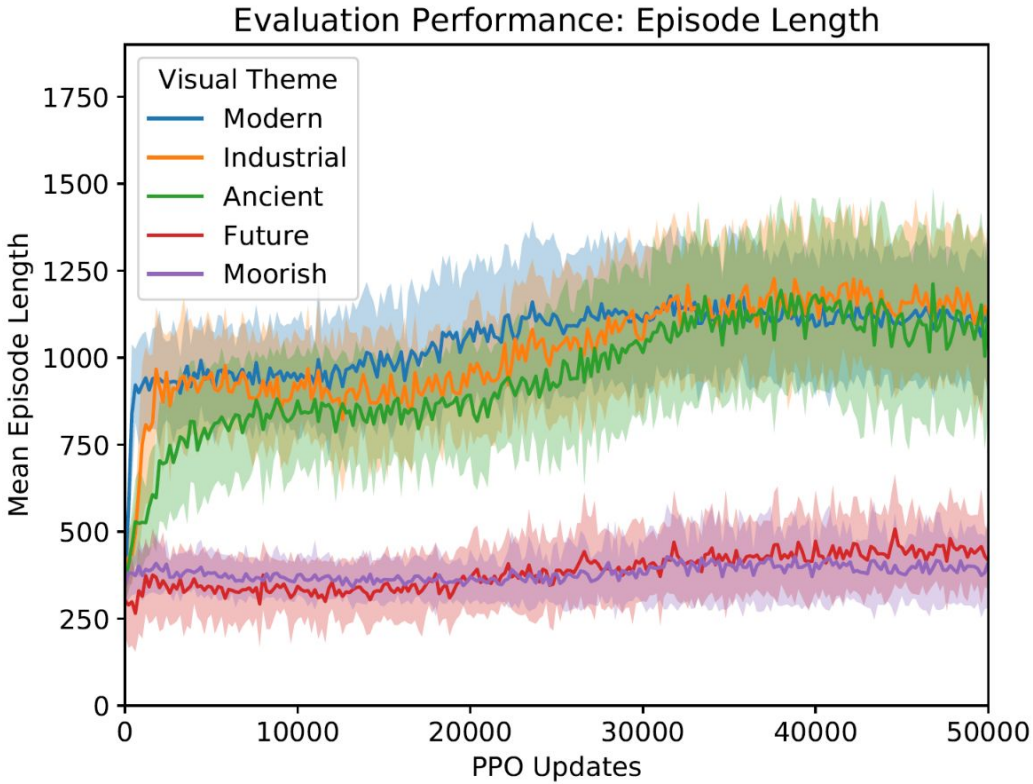
Ab wann lohnt sich eine GPU?

- Wann ist die GPU (besser gesagt die Schnittstelle) ein Flaschenhals?
  - Geringes Aufkommen von Trainingsdaten
  - Kleine Observationsräume
  - Kleine Batches

3. Ergebnisse Visualisieren

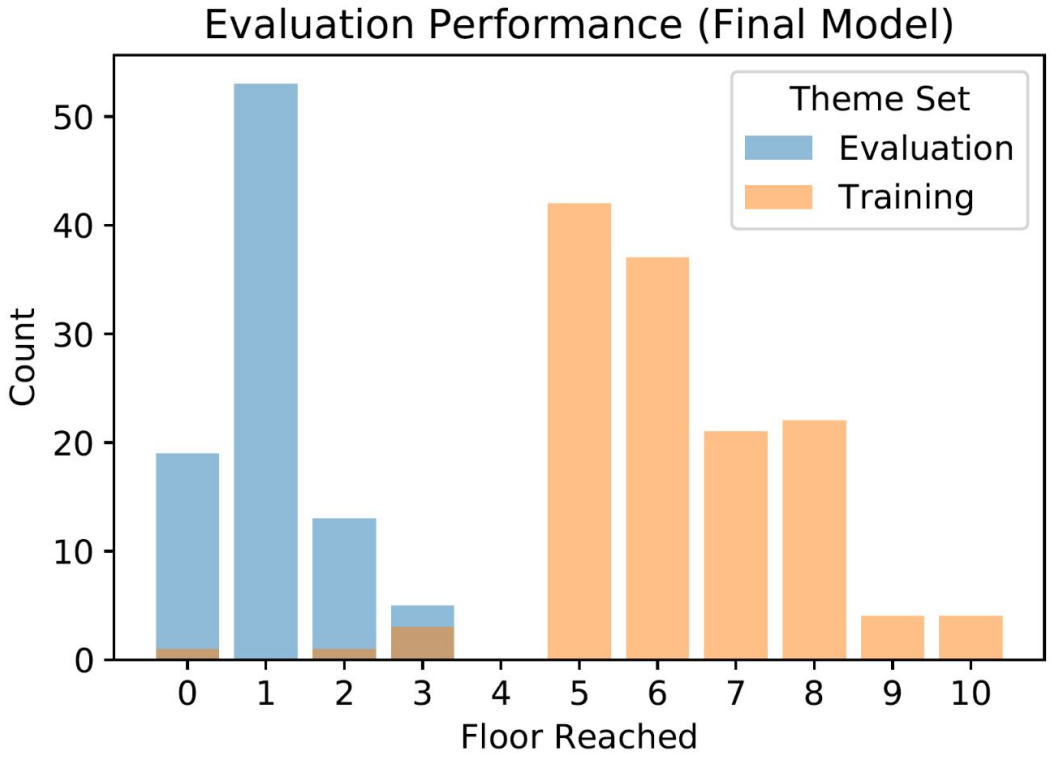
Plots

- Was plotten?
  - Cumulative Reward
  - Episode Length
  - Loss
  - KL-Divergence
  - ...



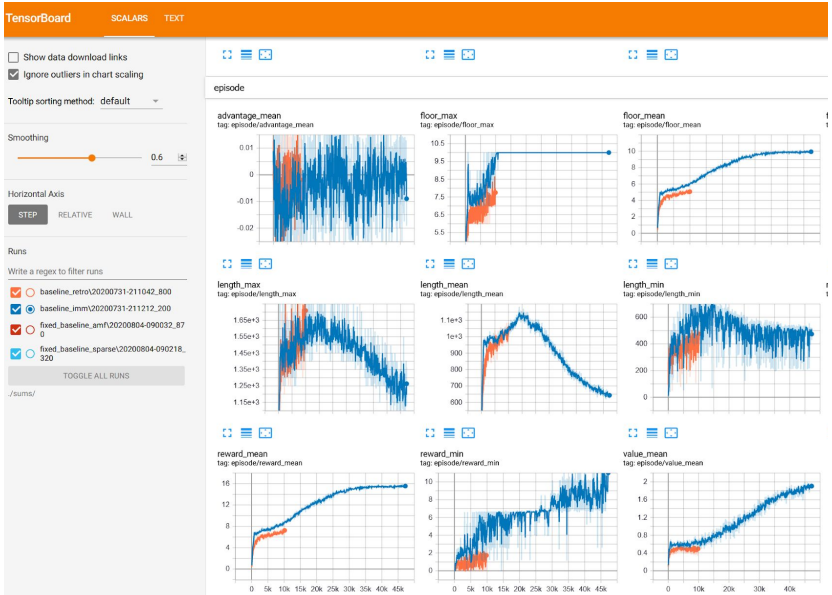
Wie plotten?

- Line plot
- Box plot
- Bar plot
- Scatter plot
- Histogram
- …



Tensorboard

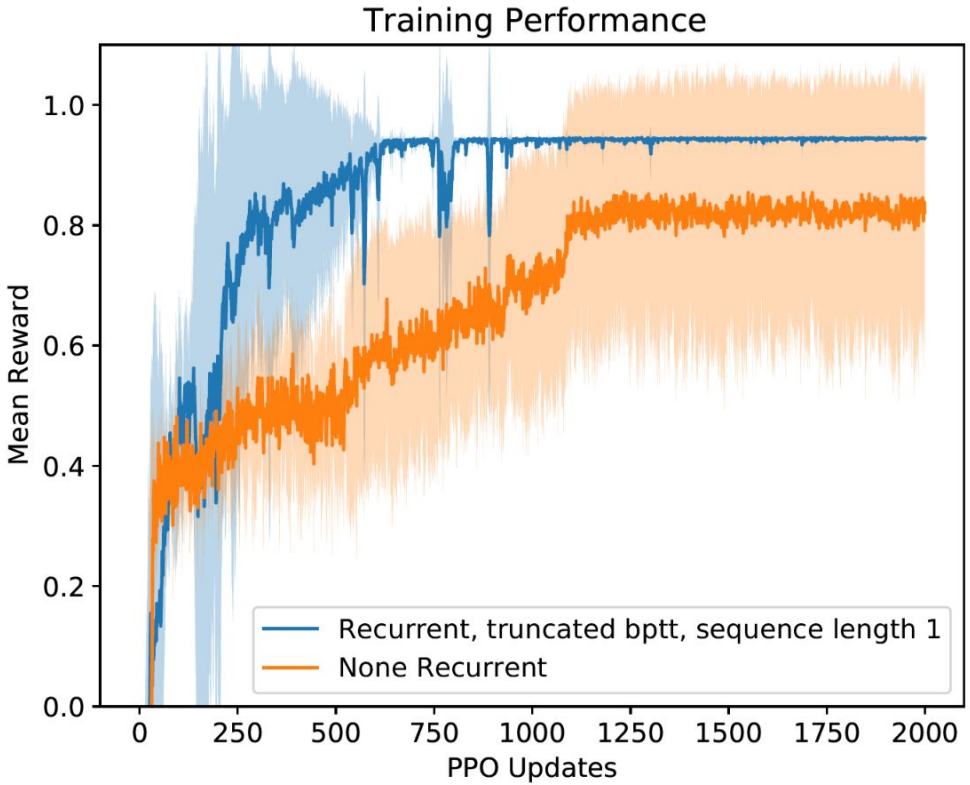
- Gut zur Beobachtung von Trainingsdurchläufen
- Nicht gut geeignet um Resultate zu teilen



Line Plots

- Achsenbeschriftung
- Legende
- Titel
- ggf. Bildunterschrift

- Durchschnitt || >1 Runs ?
- Varianz plotten!



Varianzen und Abweichungen

- Min & Max
- Standard Deviation
- Asymmetric Standard Deviation
- Confidence Interval

Asymmetric Deviation

- Fast wie Standardabweichung, aber...
- … welche Indices liegen unter und über dem Durchschnitt?

```
def asymmetric_std(data):
  mean = np.mean(data)

  x_up = np.where(data >= mean)[0] # returns indices
  x_up = data[x_up.tolist()]
  k = x_up.shape[0]

  x_down = np.where(data <= mean)[0] # returns indices
  x_down = data[x_down.tolist()]
  l = x_down.shape[0]

  std_up = np.sqrt((1/(k-1)) * np.sum((x_up-mean)**2))
  std_down = np.sqrt((1/(l-1)) * np.sum((x_down-mean)**2))

  return std_up, std_down
```

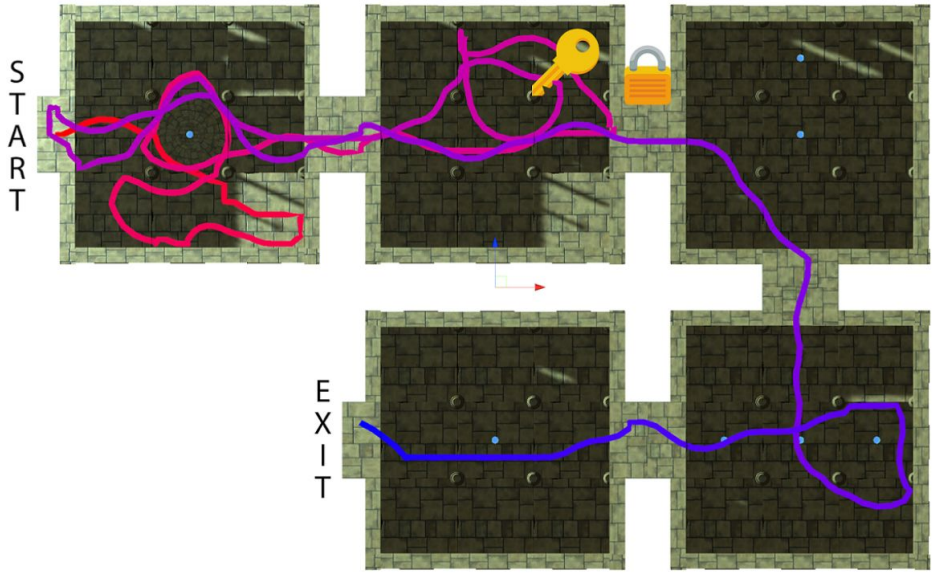
Confidence Interval

- In welchem Bereich liegt der wahre Mittelwert einer Stichprobe?
- Mit was für einer Wahrscheinlichkeit?
- Confidence: 95% oder 99%

```
import scipy.stats
scipy.stats.t.interval(0.95, len(data)-1, loc=np.mean(data),
scale=st.sem(data))
```

Verhalten des Agenten beobachten

- Viele Paper reduzieren ihre Ergebnisse nur auf Zahlen
- Das tatsächliche Verhalten bzw. die gelernte Strategie kommt zu kurz!
- Immer angucken oder sogar Videos aufnehmen



Fragen?

Kontakt

Roman Kalkreuth
roman.kalkreuth@tu-dortmund.de

Marco Pleines
marco.pleines@tu-dortmund.de