# Deep Learning

*A course about theory & practice*

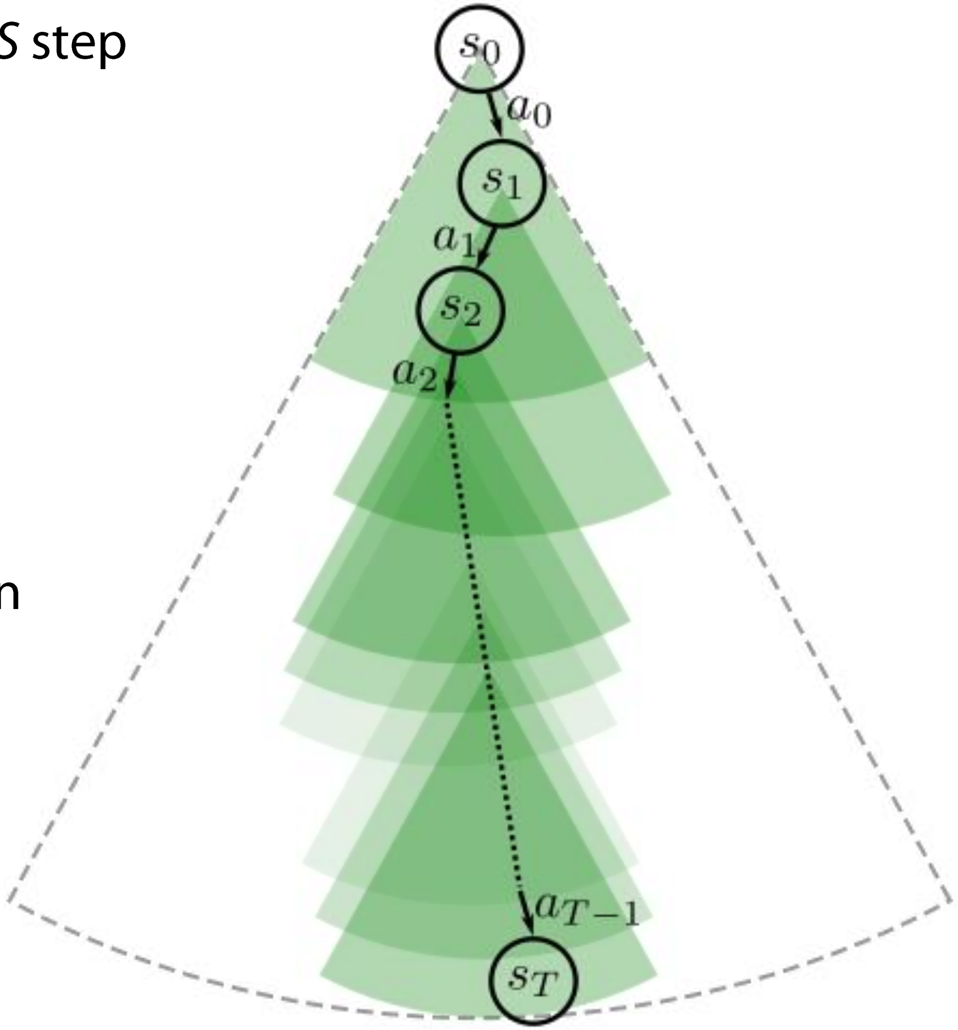
## AlphaZero

Marco Piastra

# AlphaZero =MCTS + DNN

# Monte Carlo Tree Search (MCTS) method

- ***MCTS** method:*
  - *memory* of past playouts in a single *MCTS* step
    *(collected in the tree statistics)*
  - *knowledge transfer* between *MCTS* steps
    *(by reusing subtrees already explored)*
  - optimal policy only *partially* defined
    *(on actually computed states)*
  - *intrinsically stochastic* policy optimization
    *(the same initial state*
    *can give rise to different optimizations)*
  - What about *knowledge transfer*
    between *MCTS episodes?*
    *transferring the entire MCTS tree*
    *would rapidly cause its explosive growth…*

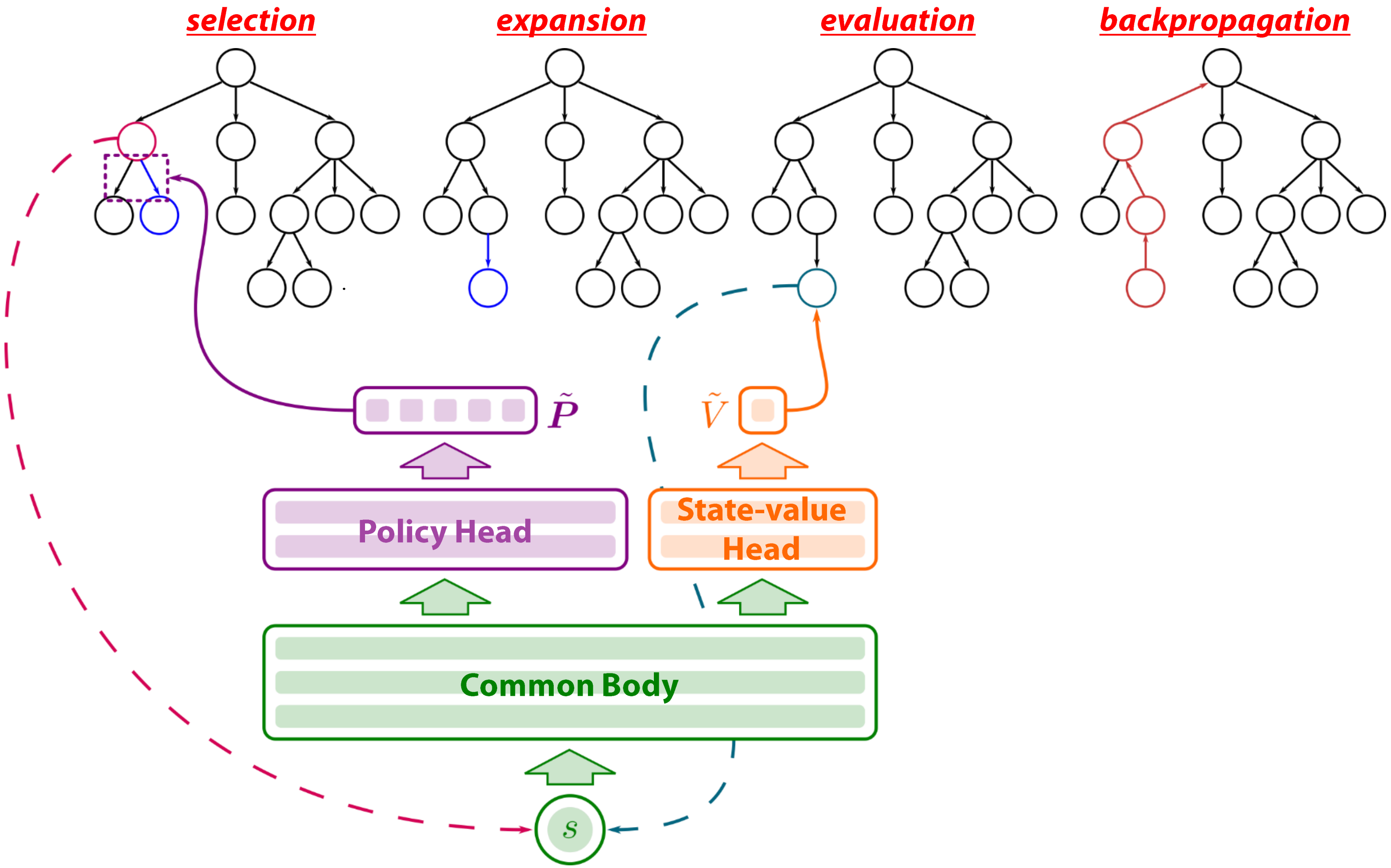


# Knowledge transfer between MCTS episodes

- ***AlphaZero*** [Silver et al. 2017]
  - *Monte Carlo Tree Search (MCTS):*
    improves the policy by focusing on the most promising actions
  - *Deep Neural Network (DNN):*
    learns the improved policy and transfers it between MCTS episodes

# AlphaZero

- ***AlphaZero = MCTS + DNN***



# DNN in AlphaZero

- **DNN in AlphaZero**
  - <u>*input:*</u> a *state* $s$
  - <u>*output:*</u> a *probability distribution* $\tilde{\boldsymbol{P}}(s) := [\tilde{P}(a \mid s)]_{a \in \mathcal{A}(S)}$

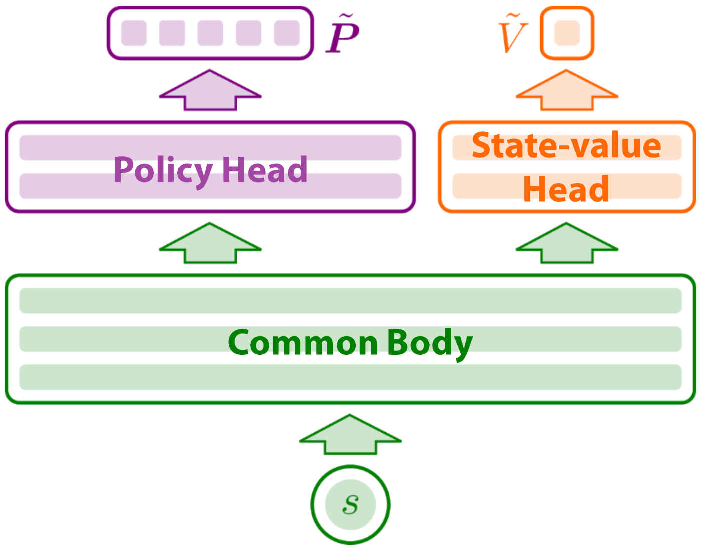
stochastic policy (a <u>vector</u> of probabilities)

and a *state-value* $\tilde{V}(s)$

predicts the expected reward for state $s$

acts as an <u>***actor-critic***</u> in the <u>*training*</u> of ***parameters*** $\vartheta$ of the net

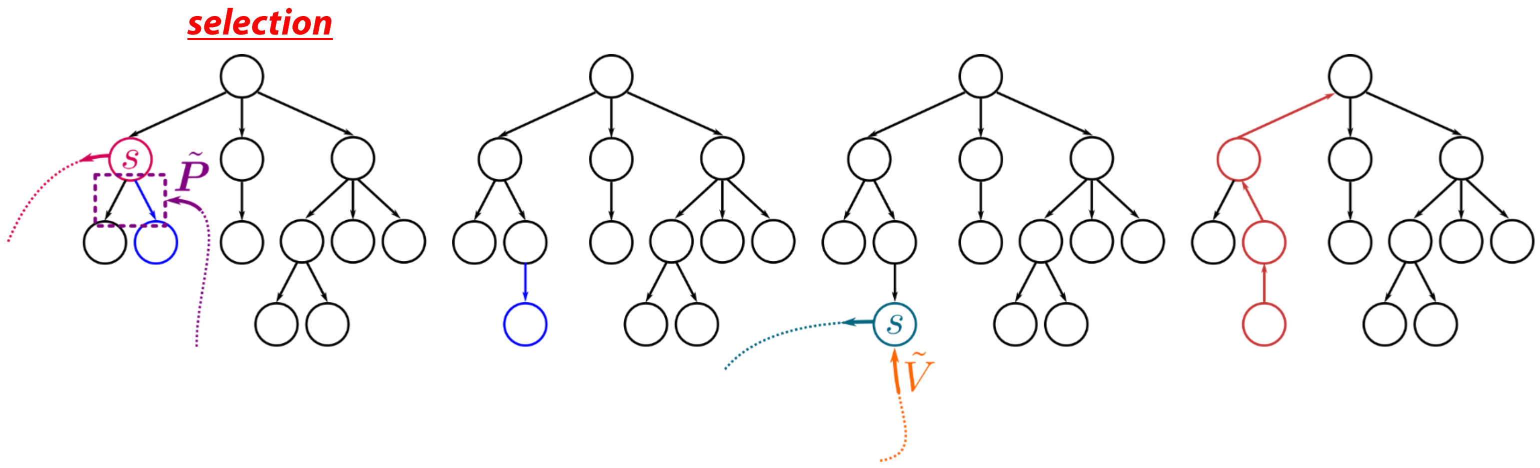
$\tilde{V}$ is compared with the *actual* reward $r$, which also impacts on training $\tilde{\boldsymbol{P}}$ by <u>*backpropagating*</u> through the *Common Body*




**"Y" shape**

# MCTS step in AlphaZero

- ***MCTS step* in *AlphaZero***




- <u>*selection:*</u> *UCT* policy is replaced with ***PUCT*** *("Predictor" + UCT)*

*MCTS* estimation of $Q(s,a)$ for *DNN* policy

*DNN* policy

$$\pi^{\mathrm{PUCT}}(s) := \operatorname*{argmax}_a \left\{ \hat{Q}(s,a) + c(s)\tilde{P}(a \mid s) \frac{\sqrt{N(s)}}{N(s,a) + 1} \right\}$$

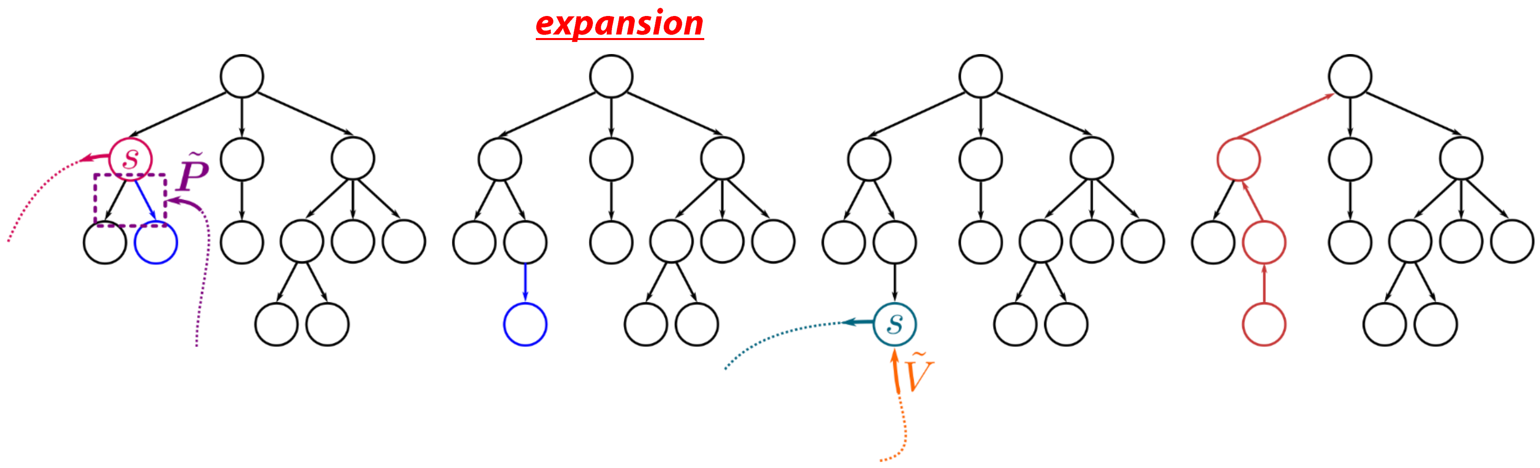
***exploration rate*** $c(s) := \log \frac{1 + N(s) + c_{\mathrm{base}}}{c_{\mathrm{base}}} + c_{\mathrm{init}}$

(slowly grows with search time)

avoids division by 0

# MCTS step in AlphaZero

- ***MCTS step*** **in** ***AlphaZero***

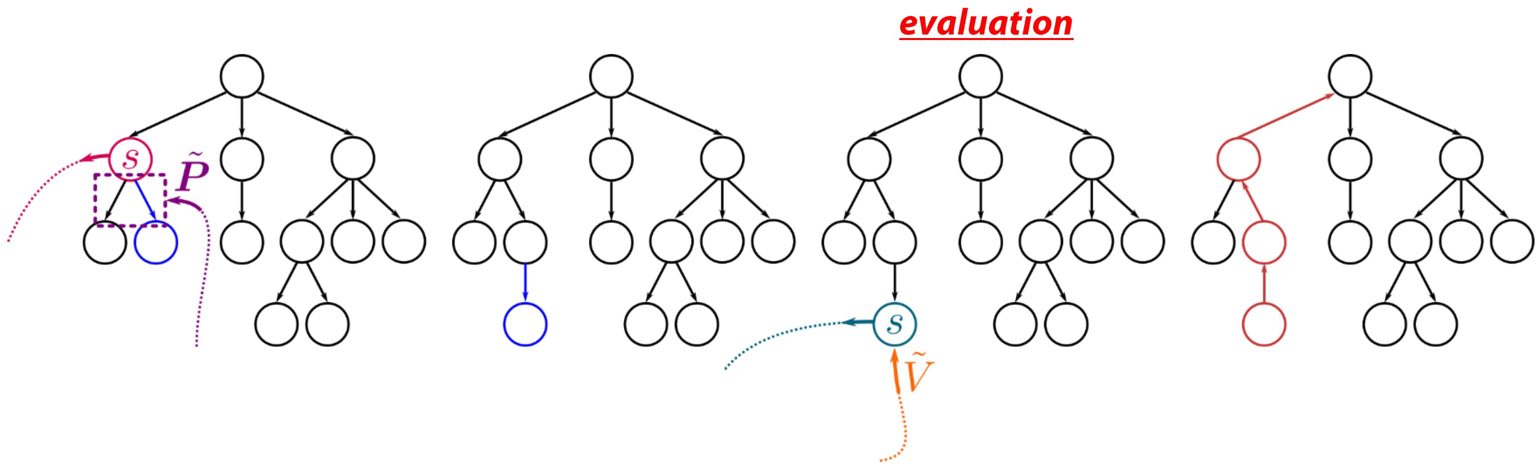



- *expansion:* initialization of the leaf new node $s_L$:

$$N(s_L) := 0 \quad \text{and} \quad \forall\, a \in \mathcal{A}(s_L) \qquad N(s_L, a_L) := 0, \quad \hat{Q}(s_L, a_L) := -\infty$$

# MCTS step in AlphaZero

- ***MCTS step** in **AlphaZero***



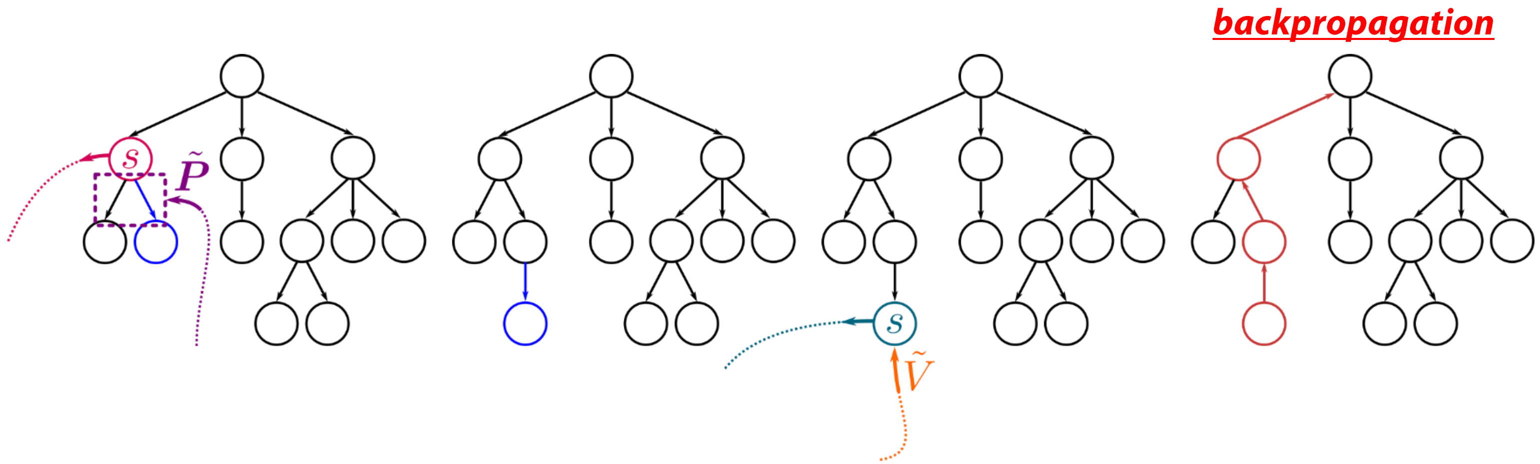

- *expansion:* initialization of the leaf new node $s_L$:

$$N(s_L) := 0 \quad \text{and} \quad \forall\, a \in \mathcal{A}(s_L) \qquad N(s_L, a_L) := 0, \quad \hat{Q}(s_L, a_L) := -\infty$$

- *evaluation* (in place of *simulation*): expected reward is $\tilde{V}(s_L)$

# MCTS step in AlphaZero

- ***MCTS step*** **in** ***AlphaZero***




- *expansion:* initialization of the leaf new node $s_L$:

$$N(s_L) := 0 \quad \text{and} \quad \forall\, a \in \mathcal{A}(s_L) \qquad N(s_L, a_L) := 0, \quad \hat{Q}(s_L, a_L) := -\infty$$

- *evaluation* (in place of *simulation*): expected reward is $\tilde{V}(s_L)$
- *backpropagation:* for each state $s$ and action $a$ visited in selection/expansion:

$$\begin{array}{c} N(s) := N(s) + 1, \\ N(s,a) := N(s,a) + 1 \end{array} \quad \text{and} \quad \hat{Q}(s,a) := \hat{Q}(s,a) + \frac{\boxed{\tilde{V}(s_L)} - \hat{Q}(s,a)}{N(s,a)}$$

# MCTS step in AlphaZero: policies

- <u>Selection</u> policy: **PUCT**

$$\pi^{\text{sel}}(s) := \pi^{\text{PUCT}}(s) := \operatorname*{argmax}_a \left\{ \hat{Q}(s,a) + c(s)\tilde{P}(a \mid s)\frac{\sqrt{N(s)}}{N(s,a)+1} \right\}$$

- <u>Output</u> policy:

$$\pi^{\text{out}}(s) \sim \left[ \hat{P}(a \mid s) := \frac{N(s,a)}{N(s)} \right]_{a \in \mathcal{A}(s)}$$

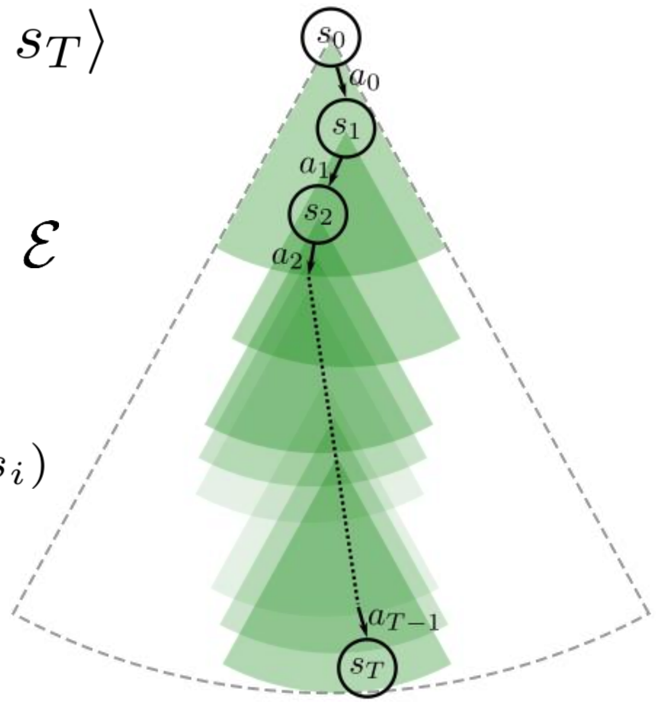
taking frequencies as probabilities
(in place of their argmax as output action)
ensures ***exploration***

*(the <u>simulation</u> policy does not exist anymore)*

# DNN training in AlphaZero

- ***Data items*** from a <u>single</u> *MCTS episode:*

After an *MCTS episode* $\mathcal{E} := \langle s_0, a_0, s_1, \ldots, a_{T-1}, s_T \rangle$
with actual reward $\hat{V}^{\mathcal{E}} := r(s_T)$:




- for each <u>*non-terminal*</u> state $s_i$ $(i = 0 \ldots T-1)$ in $\mathcal{E}$

$$\hat{\boldsymbol{P}}(s_i) := \left[ \hat{P}(a \mid s_i) := \frac{N(s_i, a)}{N(s_i)} \right]_{a \in \mathcal{A}(s_i)}$$

<u>vector</u> of frequencies

- the ***output*** of $\mathcal{E}$ is

$$D^{\mathcal{E}} := \Big\{ \underbrace{\langle s_i, \hat{\boldsymbol{P}}(s_i), \hat{V}^{\mathcal{E}} \rangle}_{\text{data item}} \Big\}_{i=0 \ldots T-1}$$

# DNN training in AlphaZero

- ***Iteration:***

$K$ times $\begin{cases} \text{1) play one } \textit{MCTS episode } \mathcal{E}_j \\ \text{2) collect data items } D^{\mathcal{E}_j} \end{cases}$

3) train the parameters of the *DNN* by using as ***dataset***

$$D := \bigcup_{j=1}^{K} D^{\mathcal{E}_j}$$

- In the limit of *infinite* iterations:

$$\pi^{\mathrm{DNN}}(s) := \operatorname*{argmax}_{a \in \mathcal{A}(s)} \tilde{P}(a \mid s) \to \pi^*(s) \quad \forall\, s$$

# AlphaZero

- ***AlphaZero*:**
  - *memory* of past playouts in a single *MCTS* step
    *(collected in the tree statistics)*
  - *knowledge transfer* between *MCTS* steps
    *(by reusing subtrees already explored)*
  - *knowledge transfer* between *MCTS* episodes
    *(provided by DNN)*

  - *deterministic* policy optimization
    with policy defined for all states $s$ :

$$\pi^{\mathrm{DNN}}(s) := \underset{a \in \mathcal{A}(s)}{\operatorname{argmax}} \tilde{P}(a \mid s)$$