IeorIITB2048

Prof. S. Kalyanakrishnan and Devanand

Policy encoding as 2-ply search with evaluation function: We considered each configuration of the tiles on a board as a state and the available moves from the state as an action set of that state. At any time step t, given a state S_t and a set of possible actions A_{S_t}, the controller has to take an action $a_t^* \in A_{S_t}$ such that the total reward i.e. total score achieved on completion of the game should be maximized. In other words, we must have a "good"policy to achieve this total reward. To obtain this policy, we have come up with an evaluation based technique which uses 2-ply search to obtain the evaluation-value of the given state. Given a state S_t and an action set A_{S_t} as an input, for each time step t, from start till termination of the game, we obtain the following action:

$$a_t^* = \arg\max_{a_i \in A_{S_t}} \{\text{Exp}(\text{Reward}(S_t, a_i))\}.$$

Here (S_t, a_i) is a state obtained by taking an action a_i on state S_t.
a_t^* can be extended as,

$$a_t^* = \arg\max_{a_i \in A_{S_t}} \sum_{\substack{S_{j,i} \in \text{ set of possible states generated} \\ \text{after addition of tile 2 or 4 in } (S_t, a_i)}} Prob((S_t, a_i), S_{j,i}) \times \text{Reward}(S_{j,i}).$$

where $Prob((S_t, a_i), S_{j,i})$ is the transition probability from (S_t, a_i) to $S_{j,i}$ and $\text{Reward}(S_{j,i})$ is maximum evaluation-value of the possible next state, i.e., $\text{Reward}(S_{j,i}) = \max_{a_q \in A_{S_{j,i}}} \{Eval(S_{j,i}, a_q)\}$.

Evaluation function: The Evaluation function has the form :

$$Eval = w_1 \times f_1 + w_2 \times f_2 + w_3 \times f_3 \cdots + w_n \times f_n$$

where $Eval$, gives evaluation-values of the given configuration of tiles (state) in the board. It is a linear combinations of the features $(f_1, f_2, f_3, \cdots, f_n,)$ weighted by coefficients $(w_1, w_2, w_3, \cdots, w_n)$. Note that each feature defines "goodness" of the board position and corresponding weights.

List of features: Feature vector F contains 9 features of the 2048 board. These are as follows:-

$\mathbf{f_1}$: Feature f_1 is 1, if tile with maximum value is on one of the corners in a given state. Otherwise, it is 0.

$\mathbf{f_2}$: Feature f_2 is 1, if second or third highest value tile is next to highest value tile. Otherwise, it is 0.

$\mathbf{f_3}$: Immediate reward received after making a move from current the state to

next state of the board. Here, the reward is defined as sum of total changes in the tiles values of each of the cells when the board changes from one state to the other.
f_4 : Number of non-empty cells in the given state of the board.
f_5 : how many out of selected cells in the given state have equal neighbors. $f_5 = EN(0,0)+EN(1,1)+EN(0,3)+EN(3,0)+EN(2,2)$. Where $EN(x,y)$ is 1 if cell (x,y) has equal neighbors otherwise, it is 0.
f_6 : It is similar to feature f_5. $f_6 = EN(2,3) + EN(3,2) + EN(0,1) + EN(0,2) + EN(1,0) + EN(2,0) + EN(1,3) + EN(3,1)$.
f_7 : Feature f_7 is 1 if the row/column containing highest and 2^{nd} highest value tile does not have empty cell. Otherwise, it is 0.
f_8 : Feature f_8 is 1 if row/column containing highest and 2^{nd} highest value tile is in descending order. Otherwise it is 0.
f_9 : Density around the highest tile cell. Density defines the sum of tiles in the row/column contains highest and 2^{nd} highest tiles + sum of tiles in its adjacent row/column divided by sum of total tiles on the board.

Learning Methodology: Cross Entropy Method
The Algorithm uses the following steps to learn weight vector:

Initialize: Choose initial parameters - mean μ_{0i} and standard deviation σ_{0i}, for individuals w_i corresponding to the weight vector: $W_0 = (w_{01}, w_{02}, w_{03}, \cdots, w_{09})$. Set $k = 0$ and $itrn = 1000$.
Step i: Generate N random sample vectors $X_1 = (X_{11}, X_{12} \cdots, X_{19}), X_2 = (X_{21}, X_{22} \cdots, X_{29}), \cdots, X_N = (X_{N1}, X_{N2} \cdots, X_{N9})$ using normal sample distribution with parameter vectors $(\mu_{k1}, \mu_{k2}, \cdots, \mu_{k9})$ and $(\sigma_{k1}, \sigma_{k2}, \cdots, \sigma_{k9})$.
Step ii: For each generated sample as an input weight vector, use policy discussed above which is based on evaluation function that returns the corresponding output value Out_j, $\forall j = 1, 2, 3, \cdots, N$.
Step iii: Sort sample vectors by generated output values (in descending order). Assign the top output value as $OutTop_k$.
If $k > itrn$ or output value starts converging **Then** Exit with $W_k = OutTop_k$ and corresponding input weight vector.
Else $k = k + 1$.
Step iv: Choose top m sample vectors and evaluate mean and standard deviation as:

$$\mu_{ki} = \frac{1}{m}\sum_{j=1}^{m} X_{ji} \text{ and } \sigma_{ki} = \sqrt{\frac{1}{m}\sum_{j=0}^{m}(\mu_{ki} - X_{ji})^2}. \quad \forall i = 1, 2, 3, \cdots, 9$$

and return to **Step i**
