Initialize controller network with random weights;
for *episode = 1,M* **do**
    Initialize memory matrix M_0;
    Set attention distribution for each head as uniform random over M_0;
    Calculate read vector $r_0 = \sum_i w_0^r(i)M_0(i)$;
    for t *= 1,T* **do**
        Use input x_t and read vector r_{t-1} to compute the interface vector ξ_t and output y_t of the controller net;
        Subdivide ξ_t and process the parameters as shown above;
        Calculate the attention distribution for the write head w_t^w using w_{t-1}^w, M_{t-1} and the corresponding parameters;
        Calculate the new memory matrix M_t using w_t^w and the add and erase vectors as in (1) and (2);
        Calculate the attention distribution for the read head w_t^r using w_{t-1}^r, M_t and the corresponding parameters as in (3);
        Calculate the read vector r_t using w_t^r and M_t;
        Perform a gradient descent step on the loss (between output and target) to update the weights;
    end
end

Algorithm 1: Neural Turing Machine