Algorithm 1 Private training of HRNet

Input: D : trajectory dataset, b : minibatch size, d : the number of resolutions, h_0 : initial state, σ : noise parameter, C : clipping parameter, η : learning rate, T : number of iterations

for $t \in [T]$ **do**
  $B_t \leftarrow$ take a random minibatch with probability $b/|D|$
  for $\mathbf{v} \in B_t$ **do**
    for $i_{\text{seq}} \in [|\mathbf{v}|]$ **do**
      $v'_{i_{\text{seq}}} \leftarrow$ encode $v_{i_{\text{seq}}}$ with the hierarchical location encoding component
      $h_{i_{\text{seq}}} \leftarrow$ encode prefix with $h_{i_{\text{seq}}-1}$ and $v'_{i_{\text{seq}}}$
      $\text{query}_{i_{\text{seq}}} \leftarrow$ convert $h_{i_{\text{seq}}}$ to a query vector by a feed-forward neural network.
      for $i_{\text{res}} \in [d]$ **do**
        for $l \in L_{i_{\text{res}}}$ **do**
          $l' \leftarrow$ encode the grid l at resolution i_{res} with the hierarchical location encoding component
          $\text{key}_l \leftarrow$ convert l' to a key vector by a feed-forward neural network
          $\text{score}_l \leftarrow$ derive the score by dot production between $\text{query}_{i_{\text{seq}}}$ and key_l
        $\Pr(l^{i_{\text{res}}}|h_{i_{\text{seq}}}) \leftarrow$ derive the probability distribution at resolution i_{res} with the softmax function to $\text{score}_{[1,\dots,|L_{i_{\text{res}}}|]}$
        $\text{loss}^{i_{\text{res}}}_{i_{\text{seq}}} \leftarrow$ cross entropy loss at resolution i_{res} with $\Pr(l^{i_{\text{res}}}|h_i)$ and the ground truth l at the resolution i_{res}
    $\text{loss} \leftarrow \sum_{i_{\text{res}}, i_{\text{seq}}} \text{loss}^{i_{\text{res}}}_{i_{\text{seq}}}$
    $g \leftarrow$ add computed and clipped gradient with loss by C to g
  $\tilde{\mathbf{g}}_t \leftarrow \frac{1}{|B_t|}(g + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}))$
  $\theta_{t+1} \leftarrow \theta_t - \eta \tilde{\mathbf{g}}_t$

Output: θ_T

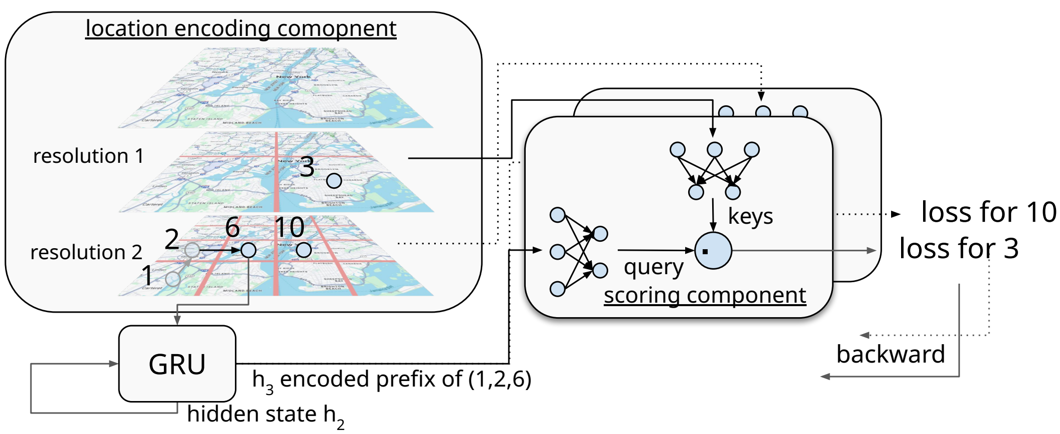

Figure 1: Overview of multi-task training with the hierarchical location encoding component. In this example, we assume that $w = 4$, so we have sixteen POIs. This illustrates the training of grid 10 from prefix $(1, 2, 6)$. In addition to the task inferring grid 10 at resolution 2, HRNet learns grid 3 at resolution 1 using the hierarchical location encoding component.

1 HRNET

In this section, we introduce the Hierarchical and Multi-Resolution Network (HRNet). HRNet's core novelty is the integration of a hierarchical location encoding component, which supersedes the traditional embedding matrix and scoring component. This design reduces the number of parameters, effectively alleviating the first bottleneck. Moreover, it facilitates multi-resolution interpretation, enabling multi-task learning. This approach allows HRNet to process and learn from data at multiple resolutions simultaneously, adeptly handling the difficulty of learning at the finest resolution, which alleviates the second bottleneck. Together, they contribute to HRNet's robustness to the number of POIs n_{POI} when optimized with DP-SGD.

We begin by outlining HRNet, using a simple example illustrated in Figure 1 and the pseudocode. Subsequently, we delve into detailed explanations of each component.

1.1 Overview

We provide the overview of multi-task training of HRNet, illustrated in Figure 1 and detailed in Algorithm 1. HRNet builds upon the baseline model that models conditional probabilities. Note that we simplify our explanation, sidelining the training of time information which is the same as the baseline, to emphasize the principal distinctions.

Consider a scenario depicted in Figure 1. The example involves two resolutions, featuring 4 and 16 grids, and trajectory $(1, 2, 6, 10)$ at resolution 2. In the learning phase for $v = (1, 2, 6, 10)$, particularly at step 3 (step 1 is learning 2 from (1), step 2 is learning 6 from $(1, 2)$, and step 3 is learning 10 from $(1, 2, 6)$), HRNet operates as follows:

The hierarchical location encoding component encodes the location 6 (Line 5). The GRU cell computes h_3, embedding of the prefix $(1, 2, 6)$, using h_2 and the encoded vector of 6 (Line 6). h_3 forms a *query vector* with the feed-forward neural networks (Line 7) for subsequent steps. Unlike the baseline model that focuses exclusively on resolution 2, HRNet concurrently considers resolution 1, that is, learning 3 at resolution 1 from $(1, 2, 6)$ (Line 8). At resolution 1, the model uses the hierarchical location encoding component to encode the four grids. For each resolution, it encodes all grids and convert them to *key vectors* with feed forward neural networks (Line 9-10). It then calculates scores as the next grid of the prefix $(1, 2, 6)$ by performing a dot product with the query vector (Line 12). This operation generates a probability distribution for each resolution (i.e., over $[4]$ and $[16]$), using the softmax function (Lines 13). The model computes the cross-entropy loss based on this probability distribution and the actual value 3 at resolution 1 (Line 14). A similar process unfolds at resolution 2, where HRNet processes the sixteen grids, with the actual value being 10 at resolution 2.

Finally, the model calculates the gradients of the parameters θ from the summed cross-entropy losses. These gradients are clipped to a predefined threshold C to moderate sensitivity, and Gaussian noise is introduced during the parameter update phase (Lines 16-18). This procedure, which involves minibatch random sampling, enhances privacy through subsampling amplification (Line 2).