Hide-and-Seek Privacy Challenge

Data and Baseline description

October 16, 2020

1 Data

Access to the data requires following the steps set out at: `https://www.vanderschaar-lab.com/privacy-challenge/`.

The Amsterdam UMC data released as part of this challenge is a *sparse* csv file. The first column is an index, which is removed by `data_preprocess.py`. The second row, admissionid, is used to identify the patient to which the entry belongs. All rows consisting of the same admissionid make up a single patient. The second column, time, identifies the time at which the measurements in that entry were taken. Each further column corresponds to a feature, which may or may not have been measured at the given time. Please refer to the data dictionary provided alongside the data for a breakdown of the specific IDs for each feature column.

For the sake of discussing baselines below, let there be N patients identified by $i \in \{1, ..., N\}$. Each patient consists of $T_i \in \mathbb{N}$ tuples corresponding to T_i time-steps at which (some) features of the patient were measured, i.e. $(t^i_1, \mathbf{x}^i_1), ..., (t^i_{T_i}, \mathbf{x}^i_{T_i})$, where $\mathbf{x}^i_j = (x^{i,j}_1, ..., x^{i,j}_d) \in \mathbb{R}^{*d}$ denotes the feature vector and $\mathbb{R}^* = \mathbb{R} \cup \{*\}$ denotes the real-line together with a point $\{*\}$ denoting that a feature was not measured.

2 Hider baselines

2.1 Add-noise

The add-noise baseline simply adds Gaussian noise, $N(0, \sigma^2)$, to each element of the feature vector. The jth synthetic tuple for patient i is given by

$$(\hat{t}^i_j, \hat{\mathbf{x}}^i_j) = (t^i_j + N^{i,j}_1, (x^{i,j}_1 + N^{i,j}_2, ..., x^{i,j}_d + N^{i,j}_{d+1})) \tag{1}$$

where each $N^{i,j}_k$ is i.i.d. $N(0, \sigma^2)$ with σ needed to be specified as input.

2.2 Time-GAN

The Time-GAN baseline is explained in full in Yoon et al. [2019].

3 Seeker baselines

Denote the true dataset by $\mathcal{D}$ (of size N), the enlarged dataset by $\hat{\mathcal{D}}$ (of size $2N$), and the generated dataset by $\tilde{\mathcal{D}}$ (of size N).

3.1 Nearest Neighbour

For each element $(t^i_j, \mathbf{x}^i_j)_{j=1}^{T_i}$ of the enlarged dataset, we compute its distance to the nearest neighbour in the generated data (where each datapoint is padded with 0s to ensure each patient series is of equal length for comparison):

$$d(i) = \min_{i' \in \tilde{\mathcal{D}}} ||(t^i_j, \mathbf{x}^i_j)_{j=1}^{T_i} - (\tilde{t}^{i'}_{j'}, \tilde{\mathbf{x}}^{i'}_{j'})_{j'=1}^{T_{i'}}||_2 \tag{2}$$

The predicted dataset, $\bar{\mathcal{D}} \subset \hat{\mathcal{D}}$, is given by the N elements for which $d(i)$ is smallest.

3.2 Binary predictor

A binary classifier, f, (an RNN) is trained using the union of $\hat{\mathcal{D}}$ and $\tilde{\mathcal{D}}$, where it is trained, using binary cross-entropy, to label elements from $\hat{\mathcal{D}}$ as 1 and $\tilde{\mathcal{D}}$ as 0. The predicted dataset, $\bar{\mathcal{D}} \subset \hat{\mathcal{D}}$, is given by the N elements for which $f((t^i_j, \mathbf{x}^i_j)_{j=1}^{T_i})$ is smallest.

References

Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 5509–5519, 2019.