REVIEW OF ICDM 2017

Dinh Huu Nguyen, 2017

Abstract: review of ICDM 2017.

Contents

1. Adaptive Laplace Mechanism: Differential Privacy Preservation in Deep Learning

This paper [1] develops the adaptive Laplace mechanism to preserve differential privacy in neural networks such that

- consumption of privacy budget ϵ is independent of number of training steps
- it can adaptively add noise to features based on their contributions to the output
- it applies to a variety of neural networks

To achieve this, the mechanism perturbs the preprocessing affine transformation and the loss function in the network.

Let X be a general dataset of n samples $x_1, \ldots, x_n$ and m features $X_1, \ldots, X_m$

$$\begin{array}{c|ccc} \text{X} & X_1 & \ldots & X_m \\ \hline x_1 & x_{11} & \ldots & x_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ x_n & x_{n1} & \ldots & x_{nm} \end{array}$$

and let X' be a neighboring dataset that differs from X by one sample.

Let $y_j = (0, \ldots, 1, \ldots, 0) \in \mathbb{R}^c$ be the multiclass label of sample x_j.

Let

$$x \xrightarrow{W_1} h_1 \quad \ldots \quad h_{s-1} \xrightarrow{W_s} h_s \xrightarrow{W_{s+1}} y$$

be a general neural network of s hidden layers $h_1, \dots, h_s$ that optimizes loss function L on t batches $B_1, \dots, B_t$ by stochastic gradient descent.

Definition 1.1. (ϵ-differential privacy) A function $\mathcal{X} \xrightarrow{F} \mathbb{R}^c, x \mapsto F(x)$ fulfills ϵ-differential privacy if

$$P(F(X) = S) \leq e^{\epsilon} P(F(X') = S)$$

for all neighboring X, X' and $S \subset \mathbb{R}^c$

The privacy budget ϵ controls the amount by which $F(X), F(X')$ may differ. A smaller ϵ enforces better privacy for F.

Definition 1.2. Laplace mechanism is the popular method of adding noise of Laplace distribution to output $F(x)$ to give it ϵ-differential privacy.

Definition 1.3. Layer-wise relevance propagation is a popular algorithm to compute the relevance R_{ji} of each input feature x_{ji} of sample x_j to output $F_{x_j}(\theta)$.

With these ingredients, the adaptive Laplace mechanism follows five steps.

1. (private relevance) Obtain the average relevance of each feature X_i over all samples x_j

$$R_i = \frac{1}{n} \sum_j R_{ji}$$

   by applying layer-wise relevance propagation to a neural network trained on X. Then add Laplace noise to R_i to get $\bar{R}_i$. Privacy budget for this step is ϵ_1.
2. (private affine transformation layer with adaptive noise) Add Laplace noise n_0 to each hidden neuron of an affine transformation W_0 of the input. Based on $\bar{R}_i$, "more noise" is added to features which are "less relevant" to the model output and vice versa. Privacy budget for this step is ϵ_2.

$$\begin{tikzcd}
& h_0 \arrow[dr, "n_0"] & & & & & & \\
x \arrow[ur, "W_0"] \arrow[rr] & & \bar{h}_0 \arrow[r, "W_1"] & h_1 & \cdots & h_{s-1} \arrow[r, "W_s"] & h_s \arrow[r, "W_{s+1}"] & y
\end{tikzcd}$$

3. (local response normalization) Apply normalization n_p to each layer to bound nonlinear activation functions

$$\begin{tikzcd}
& h_0 \arrow[dr, "n_0"] & & h_1 \arrow[dr, "n_1"] & & & & h_s \arrow[dr, "n_s"] & & \\
x \arrow[ur, "W_0"] \arrow[rr] & & \bar{h}_0 \arrow[ur, "W_1"] \arrow[rr] & & \bar{h}_1 & \cdots & \bar{h}_{s-1} \arrow[ur, "W_s"] \arrow[rr] & & \bar{h}_s \arrow[r, "W_{s+1}"] & y
\end{tikzcd}$$

4. (perturbation of loss function) Derive a polynomial approximation to loss function F. Then add Laplace noise to $F_{B_q}(\theta)$ to get $\bar{F}_{B_q}(\theta)$ for each batch B_q. Privacy budget for this step is ϵ_3.
5. (training) Update θ_q for loss function $\bar{F}_{B_q}(\theta_q)$ for each batch B_q.

The paper shows that total privacy budget is $\epsilon_1 + \epsilon_2 + \epsilon_3$. It also shows theoretical results for sensitivities and error bounds.

2. Supervised Belief Propagation: Scalable Supervised Inference on Attributed Networks

This paper [3] develops the supervised belief propagation algorithm to compute beliefs $b_i(x_i)$ about the state x_i of node i in an attributed network such that

- it learns optimal propagation strength ϵ_{ij} for each edge (i, j)
- it applies to all attributed networks

Let $X = \{X_i\}_{i \in V}$ be a pairwise Markov random field of discrete random variables whose joint relationships are modeled as an undirected graph (V, E). The joint probability $p(X = x)$ is computed by multiplying all the potentials ϕ and ψ

$$p(X = x) = \frac{1}{Z} \prod_{i \in V} \phi_i(x_i) \prod_{(i,j) \in E} \psi_{ij}(x_i, x_j)$$

where Z is a normalizing constant. Each node potential $\phi_i(x_i)$ represents an unnormalized probability of node i being in state x_i without consideration of influences by other nodes. Each edge potential $\psi_{ij}(x_i, x_j)$ represents an unnormalized joint probability of nodes i and j being in states x_i and x_j.

Definition 2.1. An attributed network is a graph G whose edges E and vertices V have attributes such as sign, weight or feature vector.

In this paper the edges (i, j) in E have feature vector θ_{ij} while the nodes in V include negative nodes N with sign s_n and positive nodes P with sign s_p.

Definition 2.2. A belief $b_i(x_i)$ is an approximate marginal probability of node i being in state x_i.

Definition 2.3. A message $m^*_{ij}(x_j)$ is an unnormalized opinion of node i about the probability of node j being in state x_j.

Definition 2.4. Loopy belief propagation is another algorithm to compute beliefs $b_j(x_j)$ by passing messages $m^*_{ij}(x_j)$ between the variables X_i.

Loopy belief propagation uniformly initializes all messages and updates them through iterations until they converge. For this it heuristically chooses a propagation strength ϵ to model edge potentials $\psi_{ij}(x_i, x_j)$.

With these ingredients, the supervised belief propagation algorithm follows these steps.

1. split N into observed negative nodes N_{obs} and training negative nodes N_{trn}.
2. split P into observed positive nodes P_{obs} and training positive nodes P_{trn}.
3. initialize weight vector w
4. while convergence criterion is not met:
   - $b, m \longleftarrow$ propagate$(w, N_{obs}, P_{obs}, \phi)$
   - $w \longleftarrow$ update$(w, b, m, N_{trn}, P_{trn})$
5. $b, m \longleftarrow$ propagate(w, P, N, ϕ)
6. return b

The paper provides details about the propagation step, such as how to compute

$$\epsilon_{ij} = \frac{1}{1 + e^{-\theta_{ij}^t w}}$$

and details about the update step, such as how to define a differentiable cost function

$$E(w) = \lambda||w||_2^2 + \sum_{p \in P_{trn}} \sum_{n \in N_{trn}} h(b_n - b_p)$$

where $h(x) = \frac{1}{1+e^{-x/d}}$ to minimize through gradient-based approach.

The space complexity for this algorithm is $O(|\theta||E|)$ where $|\theta|$ is the number of features and $|E|$ is the number of edges.

The time complexity for this algorithm is $O(((T_1 + \nu|\theta|)|E| + |\theta||P_{trn}||N_{trn}|)T_2)$ where T_1 is the number of iterations for the propagation step, ν is the number of derivative updates for the update step, $|P_{trn}|$ is the number of positive training nodes, $|N_{trn}|$ is the number of negative training nodes, and T_2 is the number of weight updates.

The paper applies both supervised belief propagation and loopy belief propagation to classify unlabeled nodes in a partially labeled undirected attribute network for comparison.

3. Linear Time Complexity Time Series Classification with Bag-of-Pattern Features

This paper [2] develops the bag-of-pattern features to classify time series that is

- free of parameters
- competitive to Fast Shapelets, Elastic Ensemble, Bag of SFA Symbols, DTW Features, Shapeless Transform.

Definition 3.1. SAX representation uses piecewise aggregate approximation to map a time series to a word.

Definition 3.2. ANOVA F value is the ratio of mean squared variance of the feature values among different classes and mean squared variance of feature values among same class.

With these ingredients, the method follows these steps.

1. extract subsequences of length l from time series.
2. map each subsequence to a word of length w in an alphabet of size α
3. compute ANOVA F value of each word
4. form feature sets by decreasing ANOVA F value
5. select feature set by cross validation with centroids

The paper explains how subsequence length l, word length w and alphabet size α are initially set by user but later selected as the top 15% combinations during the incremental validation.

The time complexity for this method is $O(mn)$ where m is the length of the longest time series and n is the number of time series in the dataset.

The paper applies bag-of-pattern features to classify time series in the UCR time series classification archive.

References

[1] H. N. Phan, X. Wu, H. Hu, and D. Dou, *Adaptive Laplace Mechanism: Differential Privacy Preservation in Deep Learning*, 2017 ICDM.
[2] X. Li and J. Lin, *Linear Time Complexity Time Series Classification with Bag-of-Pattern Features*, 2017 ICDM.
[3] J. Yoo, S. Jo, and U. Kang, *Supervised Belief Propagation: Scalable Supervised Inference on Attributed Networks*, 2017 ICDM.

prepared by Dinh Huu Nguyen.