PTH 论文阅读汇报

夏厚 PB18051031

2021 年 7 月 8 日

汇报分为三节：论文框架梳理，PTH 模型理解，思考

1 论文框架

本小节分析**论文框架**与**大体内容**，以及由该篇论文得到的论文写作的**启示**。

1.1 ABSTRACT 摘要

摘要部分主要说明了该论文在 binary hashing 领域做出的改进。总体性说明论文介绍了什么：模型、算法、实验、性能分析。

1.2 INTRODUCTION 介绍

binary hashing 受到了广泛研究，它的目的在于将高维数据降维并量化为二进制码。binary hashing 大致分为有监督与无监督两类方法，本文基于无监督 hashing。

介绍传统 binary hashing 的各种方法，包括为无监督与有监督的。这些方法都是建立在两个阶段上：投影降维阶段、二值量化阶段。指出传统方法的固有 **neighborhood-error**。并用两个小测试集比较说明 ITQ 的领域误差，与 PTH 对其的改进。

提出本论文的三阶段 hashing 模型：在其他模型已有的两阶段后，加入一个独立的 post-tuning 阶段。并说明新阶段引入的大致实现。列举贡献如下：

- 提出了三阶段模型，并给出了一个 post-tuning 算法。

- 进一步提出了 out-of-sample 算法，使 PTH 得以处理新数据点。
- 在五个著名数据集上测试性能比当前最好的方法高出 13-58%。

Enlightenment: 背景介绍，研究现状介绍，自己所做工作介绍。

1.3 RELATED WORK 相关工作

高维数据索引实用于检索和识别。比较其他方法如何试图提高性能并指出，领域误差的产生，关键在于二阶段模型最后的二值量化部分。
Enlightenment: 调研相关方法，指出不足之处，给出改进方向。

1.4 PTH(模型及算法介绍)

在已有的二阶段模型上引出 PTH，并逐步导出优化方向：二值矩阵 U$\longrightarrow$ 无约束布尔二次规划问题 $\longrightarrow$ 组合优化问题 $\longrightarrow$ 最终给出算法与 out-of-sample 算法的实现，以及复杂度分析。
Enlightenment: 主要是解释建立模型，提出算法。需要逻辑连贯、说明清晰。

1.5 EXPERIMENT 实验

说明了实验如何进行：数据集、基准方法、性能分析策略。分别在五大数据集上，进行了**各种传统方法**与 **ITQ+PTH** 结果比较分析。在大规模数据集，如 ANN-GIST1M，与高纬度数据集，如 CIFAR-10 上，PTH 方法相比传统方法获得了更大的性能提升。在与 ANN 方法比较中，PTH 也取得了很大提升。此外进行了骨架点数目研究，主要是骨架点个数的选择，证明骨架点个数不随数据集规模变化。传统方法与 PTH 的运行时间比较，PTH 需要额外的少量时间。

Enlightenment: 实验的数据集来源，训练、测试集的划分要有清晰说明，参数选择要有理由说明。性能比较要有性能计算的来源，如使用 mAP。考虑多方面的比较如：多数据集，不同 bits，运行时间等等。像运行时间这种要有运行条件的平台说明。

1.6 CONCLUSION 结论

2 PTH 特性分析与模型详解

本节对 PTH 的优点进行分析，对 PTH 模型原理进行理解说明。

2.1 PTH 的特点

- PTH 的第三阶段是独立于前两阶段的，直接基于前两阶段的结果进行后优化。适用于当前大部分的 binary hashing 方法的输出，应用方便。
- PTH 在五大数据集上对已有 binary hashing 方法的性能进行了有效提高，是一个有效的处理高维数据的二值化后优化方法。
- PTH 在大规模数据、高维数据、特征向量值为实数的数据上对已有 binary hashing 方法性能的提高更加显著。
- PTH 的骨架点引入，使得其训练可以快速高效，在运行时间上仍然较短，并且无论是骨架点，还是 query time 都不会随数据集规模增加而增加。
- PTH 恢复了二值量化带来的高维数据的近邻信息损失。

2.2 模型理解说明

两阶段模型将数据点 $\{x_i\}_{i=1}^n \in \mathbb{R}^d$ 分为投影降维 $P(X)$，和二值量化 $sgn(P(X))$ 两个阶段。但二值量化破坏了邻域结构。这种邻域错误表示为：

$$L = \| S - V \|_F^2$$

$S, V \in \mathbb{R}^{n\times n}$,$S_{ij}$ 度量原高维空间数据点 x_i 与 x_j 是否为邻域点，V_{ij} 度量 x_i 与 x_j 对应的二值码是否相似。$S \in \{-1, 1\}^{n\times n}$。

$$S_{i,j} = \begin{cases} 1 & d(x_i, x_j) < \varepsilon \text{ 以欧式距离度量原空间的相邻性关系} \\ -1 & \text{其他.} \end{cases}$$

$$V_{ij} = (bi \cdot b_j)/m$$

b_i 与 b_j 是 x_i 与 x_j 对应的二值码。以 b_i 为列写成矩阵 $B \in \{-1,1\}^{m\times n}$，于是：

$$L = \| S - \frac{1}{m} B^T B \|_F^2$$

PTH 是要后优化已有 binary code$Z = H(X) \in \{-1,1\}^{m\times n}$。定义二值矩阵 $U \in \{-1,1\}^{m\times n}$，优化问题如下：

$$\begin{aligned} min \quad & Q(U) = \| S - \frac{1}{m}(U \circ Z)^T (U \circ Z) \|_F^2 \\ s.t. \quad & u_{ij} \in \{-1,1\} \end{aligned}$$

其中 $\circ$ 表示矩阵逐元素相乘。最终优化编码为 $B = U \circ Z$。若直接对 U 进行优化，计算难度太高。观察优化目标函数特性，**我的理解是：**因为元素都为 $\{-1,1\}$，$(-1)^2 = (1)^2 = 1$，所以目标函数中的二次项为常数，只需要优化线性项。重写优化目标函数：

$$\Rightarrow \sum_{ij}^{n} [S_{ij} - \frac{1}{m}(\sum_{k=1}^{m} u_{ki} z_{ki} u_{kj} z_{kj})]^2$$

$$\Rightarrow \sum_{ij}^{n} [S_{ij}^2 - \frac{2S_{ij}}{m}(\sum_{k=1}^{m} u_{ki} z_{ki} u_{kj} z_{kj}) + \frac{1}{m^2}(\sum_{k=1}^{m} u_{ki} z_{ki} u_{kj} z_{kj})^2]$$，取 U 的第 p 行对应线性项

$$\Rightarrow -2 \sum_{ij}^{n} u_{pi} z_{pi} u_{pj} z_{pj} [\frac{1}{m} S_{ij} - \frac{1}{m^2}(\sum_{k=1,k\neq p}^{m} u_{ki} z_{ki} u_{kj} z_{kj})]$$

$$\Rightarrow -\frac{2}{m} \sum_{ij}^{n} u_{pi} Q_{ij} (S_{ij} - \frac{1}{m} O_{ij}) u_{pj}$$ ，取 p 行第 q 个元素对应线性项

$$\Rightarrow -\frac{4}{m}(\sum_{k=1,k\neq q}^{m} u_{pk} c_{qk}) u_{pq}$$

值得说明的是，在 $(\sum_{k=1}^{m} u_{ki} z_{ki} u_{kj} z_{kj})^2$ 中取对应 p 行的线性项，为 $2(\sum_{k=1,k\neq p}^{m} u_{ki} z_{ki} u_{kj} z_{kj})$，p 行中取第 q 项的线性项同理。另外，$z_p$ 列向量其值为 Z 的第 p 行，$Q = z_p \cdot z_p^T$，所以有 $Q_{ij} = z_{pi} z_{pj}$。$O = [(U \circ Z)_{\backslash P}]^T \cdot [(U \circ Z)_{\backslash P}]$，$(U \circ Z)_{\backslash P}$ 表示去掉矩阵 $U \circ Z$ 的第 p 行。$C = Q \circ (S - \frac{1}{m} O)$，$Q, S, O$ 都是向量与其转置相乘得来的矩阵，为对称阵，所以 C 也为对称阵。到此对应 u_{pq} 最小化 $Q(U)$，就是最小化 $-\frac{4}{m}(\sum_{k=1,k\neq q}^{m} u_{pk} c_{qk}) u_{pq}$。为使其最下化，应满足如下关系：

$$u_{pq} = \begin{cases} 1 & -\frac{4}{m}(\sum_{k=1,k\neq q}^{m} u_{pk} c_{qk}) \leqslant 0 \\ -1 & -\frac{4}{m}(\sum_{k=1,k\neq q}^{m} u_{pk} c_{qk}) > 0. \end{cases}$$

这也是之后的算法中取 $u_{pq} \leftarrow sgn(\sum_k u_{pk}c_{qk})$ 的原因。接下来上述优化问题转化为组合优化问题，并进行了两步改进，以降低算法复杂度：

- 原本迭代更新系数绝对值最大的项，改为一次迭代对满足系数大于阈值 η 的项进行更新。η 一般取为所有系数的均值。如此很大程度上提升了迭代更新的速度。
- 剪枝策略，只对于投影结果接近于 0 的元素进行更新。以门限 δ 度量接近于 0 的概念，δ 一般取为投影项的绝对值均值。**我的理解是：**因为二值量化的门限为 0，投影值接近于 0 的项在二值量化时更容易引入错误。

数据 X 的后优化完成，考虑样本外的点 $q \notin X$，引入骨架点的概念，定义 X 作为骨架点。**我的理解是：**以骨架点 X 的高维到二值码的映射为基准，仿照 q 在高维空间与 X 的领域关系，去构建 q 的二值码中与 X 二值码的相似关系，从而得到 q 的二值码。之后的实验也能看到，骨架点不随数据规模增加而增加，所以这一措施对于降低复杂度至关重要。

3 思考

- PTH 能在高维和大规模数据上取得更好的性能提升，以及在原始数据以实数值为特征值的情况下取得更好提高。我认为是因为高维提供了数据更多的领域信息，比如一维线性不可分的数据在升维到二维变得线性可分，所以 PTH 能更好的依照高维邻域信息恢复二值码中的相似关系。但是大规模数据上能更有效提高准确性，暂时我还没想通，因为骨架点是恢复二值化误差的关键，但骨架点并未随数据规模变化而变化。
- PTH 在五大数据集上的表现，说明了它适用不同规模，不同维度，不同标签数目的数据，并且骨架点个数相差不远。我觉得骨架点或许与数据集在高维空间中特征值取值范围有关，更大的空间范围的邻域关系到二值码的相似性关系的映射，应该需要更多的骨架点来支撑。
- 一个不太成熟的想法，试图将 PTH 二值码作为一种有损压缩编码：如果将骨架点 $\{x_i\}$ 的高维空间点与其二值码 $\{bi\}$ 记录为索引表，对于某串二值码 b_k，在骨架点 $\{b_i\}$ 中索引相似性最高的，并依据二值码的相似，在高维空间 x_i 的邻域内寻找一个点 x_k 作为 b_k 的高维恢复。