北京郵電大學

实验报告

实验题目一：

基于机器学习的数据库基数估计任务

队长： 胡宇杭 **2022212408**

组员： 陈炳璇 **2022211479**

组员： 吴林涛 **2022212036**

学院： 计算机学院 (国家示范性软件学院)

时间： 2024 年 5 月 20 日

目录

1 实验任务背景介绍

基数估计是数据库中一个重要的模块。对于输入的查询，基数估计模块将快速估计其满足不同执行顺序条件下的记录行数，其中记录行数也称为基数。通过对不同执行顺序的基数估计，可以选择出最优的执行顺序，也称为执行计划。基数估计作为数据库关系系统查询优化器的基础和核心，对于查询性能的优化和整体数据库系统的效率至关重要，近年来伴随人工智能技术的发展，其在数据处理，提取数据之间的关系等方面展示出了更优越的性能。

传统的基数估计技术一般采用像直方图等数据结构的统计方法来拟合数据表上的数据分布 [1]。在当前大数据时代下，面对不断膨胀的数据信息、复杂多样的应用场景、异构的硬件架构和参差不齐的用户使用水平，传统方法可能很难适应这些新的场景和变化。而基于机器学习的数据基数估技术因其较强的学习能力，逐渐在数据库领域展现出了潜力和应用前景 [2]，具有很重要的现实意义。

本文的主要贡献包括以下 4 个方面：

1) 针对基于机器学习的数据库基数估计任务，我们应用了 **MLP** 多层感知机模型。通过对输入的查询特征进行学习，模型能够有效预测查询结果的基数，为数据库查询优化提供一种解决思路；

2) 我们通过将查询条件抽象为统一的特征表示，实现了对查询条件的灵活建模，使模型能够处理多种类型的查询，提高了模型的泛化性；

3) 引入了三种启发式方法 **attribute value independence, AVI**、**exponential backOff, EBO**、**minimum selectivity, Min-Sel** 分别计算得到总的选择率，使用这些指标来辅助模型从而得到更准确的选择率，从而提高模型在不充分数据下的学习能力。

4) 通过对 **IMDB** 数据集的实验验证，我们的 **MLP** 模型在基数估计任务上取得了显著的性能优越性，有效提高了数据库基数估计任务的准确度。

2 相关技术

基数估计定义：给定 **SQL** 查询语句，数据库 D，其表示在数据库 D 中执行查询 q 返回的结果行数，记为 $C(q|D) = |D'|$，其中 D' 表示查询结果。

数据库优化查询系统中，基数估计发挥了重要作用。基数估计技术根据是否使用机器学习算法分为了传统的基数估计技术和基于机器学习的基数估计技术两类。传统的基数估计一般采用统计的方法，其核心是使用某种数据结构（例如直方图，数据画像）来拟合表上的数据分布。基于直方图的算法 [3] 根据边界 $[b_1, b_2, \ldots, b_n]$ 将列数据划分成若干份, 并且统计如下信息, 一个是该属性位于 b_{i-1} 和 b_i 之间的元素行数, 另一个是位于此范围不同的元素行数, 这种直方图间隔也被称作是分桶。其中 J.Xu[4] 等人利用直方图的方式对表中的数据分布进行查询，此方法会导致基数估计偏低，任意数据分布可以通过哈希函数得到一个均匀分布。

基于数据画像的统计方法，它的核心思想是使用位图来记录元素的出现情况，从而在降低计算成本的同时提供对不同元素数量的估计。这种方法适用于大规模数据集，其中直接计算精确的基数可能变得非常昂贵。与传统的基数估计算法相比，基于数据画像的方法通常更节省内存，因为它们不需要存储实际的元素，而是使用位图记录元素的出现情况。然而这种方法是一种概率性估计，结果可能受到哈希冲突等因素的影响 [2]。

基于统计的基数估计技术适用于拟合单列的数据分布，而在处理任意多列组合上数据之间的复杂关系，其能力较弱 [1]。

还有一类基于线性映射的基数估计方法, 包括线性计数 [6] 和布隆过滤 [7]. 基本思想是使用线性哈希函数将数据均匀映射到位图上, 根据位图上每个位置被访问到的次数, 利用极大似然对基数进行估计。后者为了在有限的空间中减少哈希结果碰撞, 使用了多个独立的哈希函数, 每个元素可以被映射到固定数量的位置上, 布隆过滤的方法已经被广泛使用 [2]。

采样是一种能够替代基于统计法的基数估计方法. 它不依赖于特定假设, 能够发现一些偶然的关联从而获得更加准确的估计 [2]。基于采样的基数估计技术在大规模数据表的复杂查询场景中需要消耗大量空间存储采样的元组，同时有效采样元组会随着多表复杂的连接而减少，损失估计的性能。

综上所述，传统的基数估计技术计算通常需要存储直方图，位图，采样的元组等信息，这会占用较大的存储空间，并且可能难以适应数据的动态变化。而基于机器学习的基数估计技术则是利用机器学习或深度学习的方法来学习数据的分布和查询的特征，从而预测基数，需要存储学习映射函数 $f(\cdot)$ 的模型 [1]。相较于传统基数估计基数，其模型占用空间小，这种技术的优点是可以更好地拟合数据的复杂分布和查询的复杂关系，从而提高基数估计的准确度。本文采取的基于机器学习的数据库基数估计，成功引入了多层感知机模型，可进一步提升基数估计精度，减少空间占用，加强拟合复杂数据关系能力。

3 机器学习建模

3.1 单层感知机模型

感知机（**perceptron**）是二分类的线性分类模型，属于监督学习算法。输入为实例的特征向量，输出为实例的类别（取 $+1$ 和 -1）。感知机旨在求出将输入空间中的实例划分为两类的分离超平面。为求得超平面，感知机导入了基于误分类的损失函数，利用梯度下降法对损失函数进行最优化求解。如果训练数据集是线性可分的，则感知机一定能求得分离超平面。如果是非线性可分的数据，则无法获得超平面。给定输入 $\boldsymbol{x}$，权重 $\boldsymbol{w}$ 和偏移 $\boldsymbol{b}$，感知机输出：

$$o = \sigma(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + \boldsymbol{b}) \qquad \sigma(x) = \begin{cases} 1 & if\ x > 0 \\ -1 & otherwise \end{cases}$$

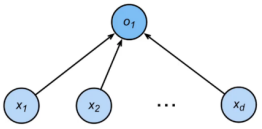

Fig.1 Illustation of SLP

图 1 SLP 示意图

3.2 多层感知机模型

多层感知机（**multilayer perceptron**，**MLP**）是人工神经网络（**ANN**）的一种。神经网络是对生物神经元的模拟和简化，包括三层：输入层，隐藏层和输出层。输入层用于接受外界向网络内传入的信息，并将这些信息传递给隐藏层。隐藏层中含有隐藏单元，这些单元将从上一层获取的信息进行计算，并将信息传递给输出层。输出层向外界传输已经处理好的信息 [5]。

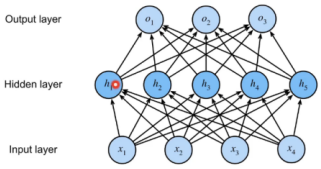

Fig.2 Illustation of MLP

图 2 MLP 示意图

对于一个多层感知机，第 l 层的输出应为：

$$X^l = \sigma(W^{l-1}X^{l-1} + b^{l-1})$$

其中，X^l 表示第 l 层的输出结果，W^{l-1} 是第 l 层与第 $l-1$ 层之间的权重矩阵，b^{l-1} 是第 $l-1$ 层的偏置，σ 是非线性激活函数。

3.3 非线性激活函数

激活函数（**activation function**）通过计算加权和并加上偏置来确定神经元是否应该被激活，它们将输入信号转换为输出的可微运算。如果不使用激活函数，每一层输出都是上层输入的线性函数，无论神经网络有多少层，输出都是输入的线性组合。使用激活函数后，能够给神经元引入非线性因素，使得神经网络可以任意逼近任何非线性函数，这样神经网络就可以利用到更多的非线性模型中。

我们使用的激活函数必须要为非线性激活函数，以单隐藏层模型为例，假设激活函数 $\sigma(\boldsymbol{x}) = \boldsymbol{x}$ 为本身，有

$$\boldsymbol{h} = \sigma(\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{b}_1)$$
$$o = \boldsymbol{w}_2^T\boldsymbol{h} + b_2$$

则输出仍为线性函数，等价于一个单层感知机。因此我们应选用非线性激活函数。常用的非线性激活函数有 **sigmoid** 函数，**tanh** 函数或 **ReLU** 函数等。

ReLU 函数：**ReLU** 函数实现简单，同时在各种预测任务中表现良好 [9]。其提供了一种非常简单的非线性变换。给定变量 x，**ReLU** 函数被定义为该元素与 0 的最大值：

$$ReLU(\boldsymbol{x}) = \begin{cases} x & x > 0 \\ 0 & x <= 0 \end{cases}$$

相较于 **Sigmoid** 和 **tanh**，**ReLU** 有以下优点：

1) 相比 **Sigmoid** 和 **tanh**，**ReLU** 摒弃了复杂的计算幂指数运算，而是采用了简单的截断，提高了运算速度；

2) 对于复杂的网络而言，**Sigmoid** 和 **tanh** 函数反向传播的过程中，饱和区域非常平缓，接近于 0，容易出现梯度消失的问题，减缓收敛速度。**Relu** 的梯度大多数情况下是常数，有助于解决复杂网络的收敛问题；

3) **ReLU** 会使一部分神经元的输出为 0，这样使得网络变得更稀疏，并且减少了参数的相互依存关系，缓解了过拟合问题的发生；

4) **ReLU** 的另一个优势是在生物上的合理性，它是单边的，相比 **sigmoid** 和 **tanh**，更符合生物神经元的特征。**ReLU** 更容易学习优化。因为其分段线性性质，导致其前传，后传，求导都是分段线性。而传统的 **Sigmoid** 函数，由于两端饱和，在传播过程中容易丢弃信息 [9]。

基于阅读相关文献获得这些原因，我们选取 **ReLU** 作为我们的激活函数。

3.4 损失函数

损失函数（**loss function**）用来度量模型的预测值 $f(x)$ 与真实值 Y 的差异程度的运算函数，它是一个非负实值函数，通常使用 $L(Y, f(x))$ 来表示，损失函数越小，模型的鲁棒性就越好。

均方误差（**mean square error,MSE**）是回归损失函数中最常用的误差，它是预测值 $f(x)$ 与目标值 Y 之间差值平方和的均值，其公式如下：

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f(x) - Y)^2$$

对于数据库基数估计任务，由于基数可能会很大，所以采用对数形式进行运算，公式变为：

$$\begin{aligned} MSE &= \frac{1}{n}\sum_{i=1}^{n}[log(act\_rows) - log(est\_rows)] \\ &= \frac{1}{n}\sum_{i=1}^{n} log(\frac{act\_rows}{est\_rows}) = MSE(q\_error) \end{aligned}$$

其中，$est\_rows = f(x), act\_rows = Y$，$q\_error = max(\frac{act\_rows}{est\_rows}, \frac{est\_rows}{act\_rows})$

3.5 反向传播

反向传播（**backpropagation，BP**）是对多层人工神经网络进行梯度下降的算法，即用链式法则以网络每层的权重为变量计算损失函数的梯度，以更新权重来最小化损失函数。

该算法存储了计算某些参数梯度时所需的任何中间变量（偏导数）。假设我们有函数 $Y = f(X)$ 和 $Z = g(Y)$，其中输入和输出 X, Y, Z 是任意形状的张量。利用链式法则，我们可以计算 Z 关于 X 的导数 [10]

$$\frac{\partial Z}{\partial X} = prod(\frac{\partial Z}{\partial Y}, \frac{\partial Y}{\partial X})$$

多层感知机的训练通常使用反向传播算法来更新权重和偏置，以最小化预测结果与真实标签之间的误差。反向传播算法通过计算误差的梯度来调整每个神经元的权重和偏置，从而逐步优化网络的性能。在训练过程中，可以使用不同的优化器进一步改善网络的训练效果。

3.6 优化器

常见的优化器有 **Adam**、**SGD**、**Momentum** 等，本文中选取了前两者作为模型的优化器。

Adam 优化器：

自适应矩估计（**adaptive moment estimation，Adam**）[11] 是一种自适应学习率的算法，对每一个参数都计算自适应学习率。其在存储一个指数衰减的历史平方梯度的平均的同时，还保存一个历史梯度的指数衰减均值 m_t，类似于动量。

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

其中，m_t 和 v_t 分别是对梯度一阶矩和二阶矩的估计。而当 m_t 和 v_t 初始化为 0 向量，尤其当衰减率很小时，它们都偏向于 0。这可以通过计算偏差校正后的 m_t 和 v_t 的来抵消 [12]

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

利用上述的公式更新参数，由此生成了 **Adam** 的更新公式。

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon}\hat{m}_t$$

SGD 优化器：

随机梯度下降（**stochastic gradient descent, SGD**）[12] 是一种常见的优化器，也是

机器学习模型中最基础的优化算法之一。它是梯度下降算法（**GD**）的一种实现方式，常被用于神经网络中的权重更新（文献引用）。

SGD 优化算法在每次参数更新时，仅选取一个样本 $x^{(i)}, y^{(i)}$ 计算其梯度，参数更新公式为

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta J_i(\theta; x^{(i)}; y^{(i)})$$

SGD 通过一次迭代进行一次更新消除了在处理大型数据集时的冗余计算，训练速度快，但由于每次迭代并不是向整体最优化的方向，**SGD** 更容易得到一个局部最优解，导致准确度下降。

Adam 和 **SGD** 作为机器学习领域中常用的两种优化算法，各自具有一些独特的特性。**Adam** 算法通过一阶和二阶梯度矩估计，以及自适应学习率，可以在数据量大或者参数复杂的场景下快速收敛。相比之下，**SGD** 算法虽然简单，但在数据量小或者对精度要求高的场景下，仍然能够找到更平坦的最小值，从而提高泛化能力。为更好的提升模型性能，我们在实际训练中分别测试了这两种优化器对模型性能的影响。

3.7 三种特征值提取方法 [13]

1) **AVI：** 方法假设不同查询条件的选择率 (**selectivity**) 之间相互独立，即满足乘法原则。其中，选择率代表某个查询查询出的记录行数占总行数的百分比。在这个假设下，总选择率为 $Pi_{k=1}^{d}(s_k)$。通过将总的选择率乘上数据总行数，便得出 **AVI** 方法估计出来的基数

2) **EBO：** 此方法不同于 **AVI** 方法考虑所有的列，其认为应选出 4 个最小的选择率组成总的选择率，为 $s_{(1)} \times s_{(2)}^{0.5} \times s_{(3)}^{0.25} \times s_{(4)}^{0.125}$，其中，$s_{(k)}$ 代表第 k 小的选择率。通过将总的选择率乘上数据总行数，便得出 **EBO** 方法估计出来的基数

3) **minimumMinSel：** 该方法较为简单，其认为总的选择率便是最小的选择率。通过将总的选择率乘上数据总行数，便得出 **MinSel** 方法估计出来的基数

4 模型优化

4.1 学习率

学习率是优化过程中最重要的超参数之一。它决定了每次参数更新的步长，对模型收敛速度和质量有显著影响。通过分别使用 **Adam** 和 **SGD** 优化器中分别测试了不同的学习率设置，目的是找到一个既能快速收敛又能避免过拟合的最佳值。由于 **Adam** 自身具有调整学习率的机制，因此对学习率的选择稍微宽容一些。相反，**SGD** 对学习率非常敏感，需要更加谨慎地选择。

4.2 迭代次数

迭代次数决定了整个数据集被遍历的次数。太少的迭代次数可能导致模型未能充分学习数据，而过多的迭代则可能导致过拟合。通过测试不同的迭代次数，以确保模型既能学习到足够的特征，又能在合适的时候停止学习。

4.3 隐藏层层数和大小

隐藏层对模型的性能、复杂度和学习能力有直接影响。增加隐藏层数量和大小，可以学习更复杂、更抽象的特征表示，使得模型能够捕捉更复杂的关系，但同时也会提高计算成本和时间；过度拟合训练数据，从而降低其在新数据上的泛化能力；可能会引起梯度消失或梯度爆炸问题，使训练变得困难。

4.4 网格搜索与 K 折交叉验证

为了较精确的确定我们模型的超参数，我们采用了 **K** 折交叉验证和网格搜索方法 [14]。**K** 折交叉验证主要步骤是：

1) 将含有 N 个样本的数据集，分出 K 份，每份含有 $\frac{N}{K}$ 个样本。选择其中一份作为验证集，另外 $K-1$ 份作为训练集，此时验证集有 K 种情况。

2) 在每种情况中，用训练集训练模型，用验证集测试模型，计算模型的泛化误差。

3) 交叉验证重复验证 K 次，平均 K 次的结果作为模型最终的泛化误差。

网格搜索是在指定的参数范围内，按步长依次调整参数，利用调整的参数训练我们的模型，从而在所有的参数中找到在验证集上精度最高的参数。

这些方法的综合应用能够使我们在多个超参数配置下全面评估模型的性能，并最终锁定最佳的学习率，迭代次数等关键参数。

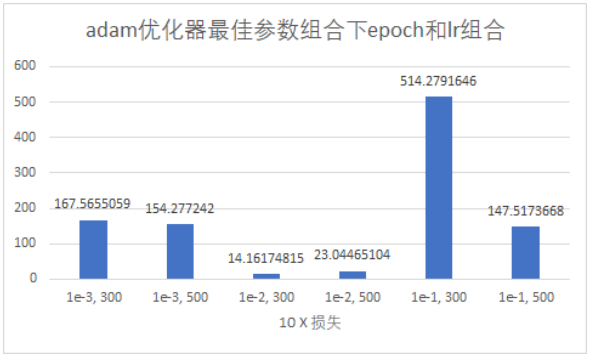

Fig.3 Loss under the different epoch and lr

图 3 不同 epoch 和 lr 组合下的损失

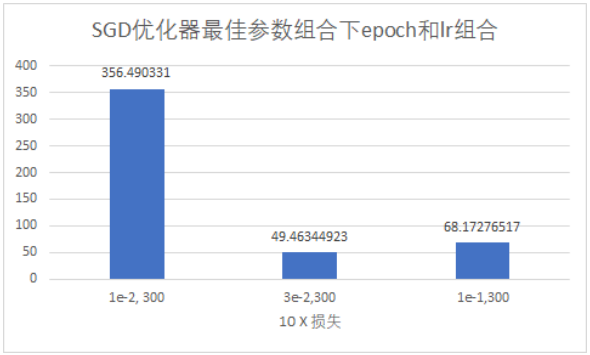

Fig.4 Loss under the different layers number

图 4 不同 epoch 和 lr 组合下的损失

5 实验测试

5.1 学习率和迭代次数选择

在实验中，我们选择了 $1e-1$ 和 $1e-2$ 作为 **Adam** 优化器的学习率，以及 $3e-2$ 和 $1e-2$ 作为 **SGD** 优化器的学习率，同时，**epoch** 设置为 300。这些选择基于对模型在不同学习率下的性能影响的分析。较高的学习率能加速初期收敛，但可能引起稳定性问题，而较低的学习率虽然训练过程缓慢，却能提供更稳定的收敛。

5.2 隐藏层层数

经过测试，如下图，隐藏层层数为 2 时，模型的拟合小效果远小于其他情况，隐藏层层数为 3, 4 时拟合效果相似，但考虑到增加隐藏层层数所带来的成本上升，我们最后确定隐藏层层数为 3 层。

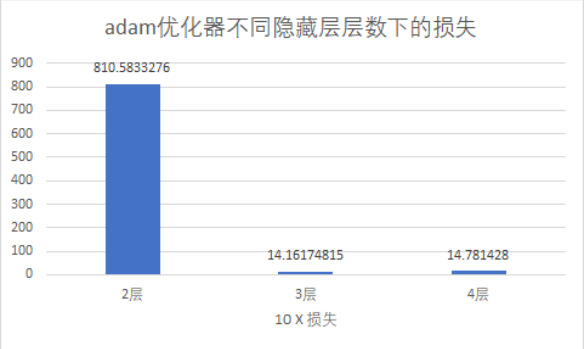

Fig.5 Loss under the different layers number

图 5 不同隐藏层层数下的损失

5.3 隐藏层大小

我们通过一系列实验和分析，确定了最佳的超参数配置。对于 **Adam** 优化器，我们设置三个隐藏层的大小分别为 100, 30, 50；而对于 SGD，参数配置为 100, 50, 50。

5.4 网格搜索与 K 折交叉验证

最终，我们通过网格搜索确定了与手动调整相似的最佳超参数，验证了我们的调整策略的有效性。此外，引入 K 折交叉验证后，模型在多样化数据上的泛化能力得到了提升，从而降低了过拟合的风险，并提高了模型的整体性能。

参考文献

[1] Yue Wenjing, Qu Wenwen, Lin Kuan, et al. A Review of Machine Learning-Based Cardinality Estimation Techniques. Journal of Computer Research and Development, 1-14 (in chinese)

（岳文静, 屈稳稳, 林宽，等. 基于机器学习的基数估计技术综述. 计算机研究与发展 1-14）

[2] Li Guoliang, Zhou Xuanhe, Sun Jie, et al. A Review of Database Technologies Based on Machine Learning. Journal of Computer Science and Technology, 2020, 43 (11): 2019-2049 (in chinese)

（李国良, 周煊赫, 孙佶，等. 基于机器学习的数据库技术综述 [J]. 计算机学报, 2020, 43 (11): 2019-2049）

[3] Zhang J,Xiao Z,Yang X,et al. Winslett, M. Differentially private histogram publication. Proceedings of the VLDB Endowment, 2013, 22: 797-822

[4] J.Xu, Z.Zhang, X.Xiao, et al. Differentially private histo-gram publication[J]. The VLDB Journal, 2019, 22(6): 797-821.

[5] Yin Xiangbing. Research on Database Query Optimization Based on AI Technology [J]. Journal of Anhui Vocational and Technical College, 2023, 22(02): 15-20+26 (in chinese)

（尹向兵. 基于 AI 技术数据库查询优化的研究 [J]. 安徽职业技术学院学报, 2023, 22 (02): 15-20+26）

[6] Alon N, Matias Y, Szegedy M. The space complexity ofapproximating the frequency moments. Journal of Computerand System Sciences, 1999, 58(1): 137-147

[7] Papapetrou O, Siberski W, Nejdl W. Cardinality estimationand dynamic length adaptation for Bloom filters. Distributedand Parallel Databases, 2010, 28(2): 119-156

[8] Anshuman Dutt, Chi Wang, Azade Nazi, Srikanth Kandula, Vivek R. Narasayya, Surajit Chaudhuri: Selectivity Estimation for Range Predicates using Lightweight Models. Proc. VLDB Endow. 12(9): 1044-1057 (2019)

[9] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks[C]//Proceedings of the fourteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 2011: 315-323.

[10] Aston Zhang, Li Mu, Zachary C. Lipton, et al. Hands-On Machine Learning with PyTorch [M]. Beijing: Posts & Telecom Press, 2023.

[11] Diederik P. Kingma , Jimmy Lei Ba. Adam: a Method for Stochastic Optimization. Interna tional Conference on Learning Representations, pages 1–13, 2015.

[12] Ruder, Sebastian. "An overview of gradient descent optimization algorithms." arXiv preprint arXiv:1609.04747 (2016).

[13] Anshuman Dutt, Chi Wang, Azade Nazi, Srikanth Kandula, Vivek R. Narasayya, Surajit Chaudhuri: Selectivity Estimation for Range Predicates using Lightweight Models. Proc. VLDB Endow. 12(9): 1044-1057 (2019)

[14] CSDN, Machine Learning: Detailed Explanation of K-Fold Cross-Validation (Deep Understanding Edition) (Most Detailed on the Internet) [2023-12-10] https://blog.csdn.net/Rocky6688/article/details/107296546 (in chinese)

CSDN, 机器学习 _K 折交叉验证知识详解（深刻理解版）（全网最详细）[2023-12-10] https://blog.csdn.net/Rocky6688/article/details/107296546