2A-01

非自己回帰モデルによる安定したカーディナリティ推定手法の提案

伊藤 竜一 † 佐々木 勇和 ‡ 肖 川 § 鬼塚 真 ¶
大阪大学 † 大阪大学 ‡ 大阪大学 § 大阪大学 ¶

1 はじめに

データベースシステムにおいて，カーディナリティ推定はクエリ応答性能に大きな影響力を持つ重要な要素技術である [1]．既存のデータベースシステムでは属性間の依存関係を考慮しないため，カーディナリティ推定の性能を悪化させる原因となっている．属性間の依存関係を考慮したカーディナリティ推定を実現するため，機械学習を用いる手法が提案されている．特に自己回帰モデルを利用する手法では従来手法を上回る性能を達成したと報告されている [2,3]．しかしながら，自己回帰モデルは推定できる属性の順序が学習時に固定されてしまうため，推定対象クエリの述語によって推定値の分散が大きくなるという問題がある．

本稿では，非自己回帰モデルによってデータを学習しクエリの述語に基づく動的な順序での推論を利用した，安定したカーディナリティ推定手法の提案を行う．評価実験を行ったところ，属性順によって数十倍の性能悪化が発生する既存手法と比較して，提案手法は安定して既存手法のピークと同等の性能であることが確認された．

2 カーディナリティ推定の定式化

カーディナリティが述語の条件を満たす同時確率から求まることに注目する．リレーション R が属性 $\boldsymbol{A} = \{A_1,..,A_n\}$ と タプル $\boldsymbol{t} = \{t_1,...,t_m\}$ から成り，条件となる述語の演算を $\boldsymbol{\theta}(t) \rightarrow \{0,1\}$ とする．カーディナリティ $\mathcal{C}$ は条件を満たすタプルの数として以下のように表される．

$$\mathcal{C}(\boldsymbol{\theta}) = |\{t \in R \mid \boldsymbol{\theta}(t) = 1\}| \tag{1}$$
$$= |R| \cdot P(\{t \in R \mid \boldsymbol{\theta}(t) = 1\}) \tag{2}$$

タプル t は属性 $\{A_1,...,A_n\}$ の要素 $\{a_1 \in A_1,...,a_n \in A_n\}$ の組み合わせで成り立つことから，

$$\mathcal{C}(\boldsymbol{\theta}) = |R| \cdot \sum_{a_1 \in A_1} ... \sum_{a_n \in A_n} \boldsymbol{\theta}(a_1,...,a_n) P(a_1,...,a_n) \tag{3}$$

とも表せる．ただし式から分かるように，この組み合わせは非常に多くなるため直接扱うことは難しい．そのため，各属性が独立であるという仮定 $P(A_1,...,A_n) \approx \prod_n P(A_i)$ をおくことで，属性ごとのヒストグラムからカーディナリティを求める手法が広く利用されている．しかしながら，この仮定は現実世界データには不適切であり，エラーの主たる原因となっている．

本稿ではそのようなヒューリスティックに基づく仮定を用いず，厳密な変換である乗法定理を利用する．$\boldsymbol{\theta}$ が conjunctive である ($\boldsymbol{\theta} = \theta_1 \otimes \theta_2 \otimes ... \otimes \theta_n$, $\theta_i(a_i) \rightarrow \{0,1\}$) とすると，カーディナリティは以下の形としても導かれる．

$$\mathcal{C}(\boldsymbol{\theta}) = |R| \cdot \sum_{a_1 \in A_1} \theta_1(a_1) P(a_1) \sum_{a_2 \in A_2} \theta_2(a_2) P(a_2 \mid A_1) \ ... \sum_{a_n \in A_n} \theta_n(a_n) P(a_n \mid A_{n-1},...,A_1) \tag{4}$$

3 提案手法

提案手法では非自己回帰モデルをコアとしてカーディナリティ推定を実現する．提案手法はデータの分布のみが学習の対象でクエリを事前に必要としないため，単一モデルで様々なクエリのカーディナリティ推定が可能である．なお対象データベースに複数のリレーションが含まれる場合は， 全てのリレーションを完全外部結合しユニバーサルリレーションとして見なすことで，結合を含むカーディナリティ推定も可能である．

3.1 学習フェーズ

非自己回帰モデルによる密度推定の学習をタプル単位で行う．リレーション内からタプルを取り出し，一部属性をマスクしたものを入力，マスクしていないものをターゲットとする．マスクをランダムに行うことで，任意の属性を条件とした残りの属性の同時確率分布を推論可能なモデルの学習となる．

3.2 推論フェーズ

2 章で述べたように，カーディナリティは述語を考慮した同時確率と総タプル数の積と等価であることから，条件付き確率からカーディナリティを推定する．Naru [2] で提案された Progressive Sampling を非自己回帰モデルに拡張する．乗法定理による同時確率の表現は厳密なものであるが，Progressive Sampling では直前の推論結果を以降の推論に利用すること，また，その際にサンプリングを行うため，乗法定理による式展開の順序が性能に影響する．特に，正確度や精度が低い属性の推論の先行は後続の推論のエラー原因となる．なお，推論性能は述語に影響される点にも注意する．自己回帰モデルでは属性順が学習の段階で固定されるため，$!|A|$ 通りの属性順毎に 1 から学習して適切な属性順を見つけることは困難である．一方提案手法で利用する非自己回帰モデルは任意の属性順で推論できる．そこで，推論時の精度が高い属性の推論を先行させるため，与えられた述語を適用したときに重複しない値が少ない属性順で推論を行う．手順を Algorithm 1 に示す．

Robust Cardinality Estimation by Non-Autoregressive Model
† Ryuichi Ito, Osaka University
‡ Yuya Sasaki, Osaka University
§ Chuan Xiao, Osaka University
¶ Makoto Onizuka, Osaka University

Algorithm 1 非自己回帰モデルを利用したカーディナリティ推定アルゴリズム

Input: $\mathcal{M}, \boldsymbol{\theta}, \boldsymbol{A}, N, |R|$
Output: $\widehat{cardinality}$

1: **procedure** ESTIMATE($\mathcal{M}, \boldsymbol{\theta}, \boldsymbol{AttrsInPreds}$))
2: Initialize *inputs* w/ nulls
3: $prob \leftarrow 1.0$
4: **foreach** $A_i \in \boldsymbol{AttrsInPreds}$ **do**
5: $\widehat{dist}_{A_i} \leftarrow \mathcal{M}(inputs)$ ▷ Forward
6: $\widehat{dist'}_{A_i} \leftarrow \{\theta_i(a_i)\widehat{dist}_{a_i} \mid a_i \in A_i\}$) ▷ Filter by θ
7: $prob \leftarrow prob * \sum_{a_i \in A_i} \widehat{dist'}_{A_i}$
8: $a_i \leftarrow$ SAMPLING($\widehat{dist'}_{A_i}$)
9: $inputs[A_i] \leftarrow$ ENCODE(a_i)
10: **end for**
11: **return** *prob*
12: **end procedure**
13:
14: $AttrsInPreds \leftarrow \{A \in \boldsymbol{A} \mid A$ USED IN $\boldsymbol{\theta}\}$
15: **foreach** $i \in N$ **do** ▷ Batched in practice
16: $probs[i] \leftarrow$ ESTIMATE($\mathcal{M}, \boldsymbol{\theta}$,
17: SORTBYFILTEREDDOMAINSIZE($AttrsInPreds$))
18: **end for**
19: $\widehat{selectivity} \leftarrow$ MEAN($probs$)
20: $\widehat{cardinality} \leftarrow \widehat{selectivity} * |R|$
21: **return** $\widehat{cardinality}$

Progressive Sampling を拡張したこのアルゴリズムは非自己回帰モデル $\mathcal{M}$，述語 $\boldsymbol{\theta}$，属性 $\boldsymbol{A}$，サンプルサイズ N，総タプル数 $|R|$ を受け取る．まず $\sum_{a_i \in A_i} P(a_i \mid \cdot)$ は常に 1 であることに注意すると，述語の対象となっていない属性の推論は不要であるため，述語の対象となっている属性のみを取り出す（14 行目）．次に乗法定理に基づく確率分布の推論と同時確率の計算を N 回行う（15-18 行目）． このとき属性は述語を適用したときに重複しない値の少ない順に並べ変えて与える（17 行目）．推論された確率分布 $\widehat{dist}_{A_i}$ に述語を適用し，その総和と *prob* の積を取り（7 行目），更に確率分布に基づくサンプリングを行い以降の入力となる属性 A_i の要素 a_i を得る（8 行目）．述語の対象となっている全属性でこれらの操作を繰り返すことで *prob* がセレクティビティの推論値となる（11 行目）．最後に，得られた N 個のセレクティビティの推論値の平均と総タプル数 $|R|$ の積をカーディナリティの推論値として返却する（19-21 行目）．

4 評価実験

提案手法として，非自己回帰モデルに多層パーセプトロン（MLP）を利用するものと Transformer (TF) を利用するものを実装した．ベンチマークとして DMV [4] と Join Order Benchmark [1]（JOB-light [5]）を利用し，実際のシステムである PostgreSQL・自己回帰モデルを用いる先行研究の Naru [2]/NeuroCard [3] [*1] と比較を行った．評価指標には推論されたカーディナリティが真の値から何倍離れているかを示す Q-error [1] を用いた．DMV での結果を表 1 に，JOB-light での結果を表 2 に示す．

表 1　DMV での Q-error と平均応答時間（ms）

手法（実装・属性順）	50th	95th	99th	Max	応答時間
PostgreSQL	1.27	57.9	$1.4 \cdot 10^3$	$7.6 \cdot 10^4$	2.93
Naru (MADE・正順)	1.04	1.31	2.00	8.00	10.3
Naru (MADE・逆順)	60.8	$7.5 \cdot 10^2$	$2.0 \cdot 10^3$	$4.2 \cdot 10^5$	10.4
Naru (TF・正順)	1.09	5.34	$9.7 \cdot 10^5$	$9.9 \cdot 10^5$	99.5
Naru (TF・逆順)	1.18	$2.0 \cdot 10^2$	$1.0 \cdot 10^3$	$2.1 \cdot 10^4$	$1.0 \cdot 10^3$
提案手法 (MLP)	1.05	1.56	2.60	49.0	5.62
提案手法 (TF)	1.04	1.41	2.33	49.0	54.0

表 2　JOB-light での Q-error と平均応答時間（ms）

手法（実装・属性順）	50th	95th	99th	Max	応答時間
PostgreSQL	7.44	$8.4 \cdot 10^2$	$3.0 \cdot 10^3$	$3.5 \cdot 10^3$	3.88
NeuroCard (正順)	1.79	19.5	36.5	43.2	$1.6 \cdot 10^2$
NeuroCard (逆順)	6.04	$1.2 \cdot 10^2$	$3.0 \cdot 10^2$	$4.0 \cdot 10^2$	$1.6 \cdot 10^2$
提案手法 (MLP)	3.14	60.5	$7.7 \cdot 10^3$	$2.2 \cdot 10^3$	46.5
提案手法 (TF)	2.41	16.8	$4.1 \cdot 10^3$	$1.3 \cdot 10^4$	$1.9 \cdot 10^3$

表 1 から，Naru は属性順によって数倍から数十倍以上性能が悪化してしまうことが分かる．一方提案手法では，そのヒューリスティックによる分散がなく，性能は Naru のピークと同等程度を達成した．より複雑なベンチマークの結果である表 2 も，提案手法が多くのクエリに対して性能が高いことを示している．しかしながら，一部のクエリに含まれる頻度が非常に低い要素の影響で性能が悪化してしまうことも確認された．全体として，推論や埋め込みの回数が削減できたことにより，同規模の Naru のモデルと比較して 2 倍程度の高速化に成功した．

5 結論

本稿では非自己回帰モデルによる安定したカーディナリティ推定手法を提案した．非自己回帰モデルの密度推定としての性質でデータの分布を学習しクエリに基づく動的な推論を行うことで，安定したカーディナリティ推定を実現した．評価実験により，実データで適切に性能を発揮することが確認された．

謝辞

この成果は，国立研究開発法人新エネルギー・産業技術総合開発機構（ＮＥＤＯ）の委託業務の結果得られたものです．

参考文献

[1] V. Leis et al. How good are query optimizers, really? *Proceedings of the VLDB Endowment*, 2015.
[2] Z. Yang et al. Deep Unsupervised Cardinality Estimation. *Proceedings of the VLDB Endowment*, 2019.
[3] Z. Yang et al. NeuroCard: One Cardinality Estimator for All Tables. 2020.
[4] State of New York. Vehicle, Snowmobile, and Boat Registrations, 2019.
[5] A. Kipf et al. Learned Cardinalities: Estimating Correlated Joins with Deep Learning. *Proceedings of the CIDR*, 2019.

[*1] NeuroCard は Naru を複数リレーションに拡張した手法である．いずれも基本的に MADE 実装だが TF 実装も提案されている．