Product Quantization и Importance Sampling: Полное руководство

1 Введение

Product Quantization (PQ) — это фундаментальная техника для эффективного сжатия и поиска в высокоразмерных векторных пространствах. Данный документ предоставляет полное математическое описание метода, его оптимизаций и расширений, включая комбинацию с Importance Sampling.

2 Математические основы Product Quantization

2.1 Формальная постановка задачи

Пусть $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ — набор векторов, где каждый $\mathbf{x} \in \mathbb{R}^D$. Цель PQ — найти отображение $q : \mathbb{R}^D \to \mathcal{C}$, где $\mathcal{C}$ — кодбук (codebook) размером K, минимизирующее ошибку квантования:

$$\min_q \sum_{i=1}^{N} \|\mathbf{x}_i - q(\mathbf{x}_i)\|^2$$

2.2 Разделение пространства

Основная идея PQ — разделение исходного пространства на M подпространств:

$$\mathbf{x} = [\underbrace{x_1, ..., x_{D/M}}_{\mathbf{x}^1} | \underbrace{x_{D/M+1}, ..., x_{2D/M}}_{\mathbf{x}^2} | ... | \underbrace{x_{(M-1)D/M+1}, ..., x_D}_{\mathbf{x}^M}]$$

Каждое подпространство $\mathbb{R}^{D/M}$ квантуется независимо.

2.3 Обучение кодбуков

Для каждого подпространства $m \in [1, M]$ обучается отдельный кодбук $\mathcal{C}^m = \{\mathbf{c}_1^m, \mathbf{c}_2^m, ..., \mathbf{c}_K^m\}$ с помощью k-means:

$$\min_{\mathcal{C}^m} \sum_{i=1}^{N} \min_{k=1..K} \|\mathbf{x}_i^m - \mathbf{c}_k^m\|^2$$

2.4 Кодирование и сжатие

Вектор $\mathbf{x}$ кодируется как последовательность индексов:

$$q(\mathbf{x}) = (j_1, j_2, ..., j_M), \quad j_m = \arg\min_k \|\mathbf{x}^m - \mathbf{c}_k^m\|^2$$

Размер сжатого представления: $M \cdot \log_2 K$ бит.

2.5 Асимметричное вычисление расстояний (ADC)

Для запроса $\mathbf{q}$ и закодированного вектора $\mathbf{y}$ с кодами $(j_1, ..., j_M)$:

$$d(\mathbf{q}, \mathbf{y})^2 \approx \sum_{m=1}^{M} \|\mathbf{q}^m - \mathbf{c}_{j_m}^m\|^2$$

Предвычисление таблицы расстояний для каждого подпространства ускоряет вычисления.

3 Методы оптимизации PQ

3.1 IVF-PQ (Inverted File System)

- **Грубая кластеризация**: Данные сначала кластеризуются на L кластеров
- **Поиск**: Для запроса ищутся только ближайшие *nprobe* кластеров
- **Эффективность**: Уменьшает количество вычисляемых расстояний

3.2 OPQ (Optimized Product Quantization)

Оптимизация через ортогональное преобразование:

$$\min_{R,\mathcal{C}} \sum_{i=1}^{N} \|R\mathbf{x}_i - q(R\mathbf{x}_i)\|^2$$

где R — ортогональная матрица ($R^T R = I$).

3.3 LOPQ (Locally Optimized PQ)

Комбинация IVF и OPQ: для каждого кластера IVF обучается отдельный OPQ.

4 Importance Sampling из PQ-кластеров

4.1 Постановка задачи Importance Sampling

Пусть требуется оценить математическое ожидание:

$$\mu = \mathbb{E}_{p(x)}[f(x)] = \int f(x)p(x)dx$$

Importance Sampling использует вспомогательное распределение $q(x)$:

$$\mu = \mathbb{E}_{q(x)}\left[f(x)\frac{p(x)}{q(x)}\right]$$

4.2 Интеграция с PQ-кластерами

4.2.1 Определение распределения $q(x)$

PQ-кластеризация естественным образом определяет распределение:

$$q(x) = \sum_{l=1}^{L} w_l \cdot q_l(x)$$

где:

- w_l — вес кластера (доля векторов в кластере)
- $q_l(x)$ — распределение внутри кластера l

4.2.2 Стратегии выборки

1. **Выборка кластеров**: Вероятность выбора кластера l:

$$P(l) = \frac{w_l \cdot I_l}{\sum_{j=1}^{L} w_j \cdot I_j}$$

   где I_l — важность кластера (на основе дисперсии $f(x)$ или других критериев)

2. **Внутрикластерная выборка**: Для выбранного кластера l:

$$x \sim q_l(x) = \mathcal{N}(\mu_l, \Sigma_l) \quad \text{(аппроксимация)}$$

4.2.3 Алгоритм реализации

Algorithm 1 Importance Sampling с PQ-кластерами

Вход: Функция $f(x)$, PQ-модель с L кластерами
Инициализация: Оценить μ_l, Σ_l для каждого кластера
Вычисление важности: $I_l = \mathbb{V}_{x \sim q_l}[f(x)]$ (дисперсия)
Нормировка весов: $P(l) = \frac{w_l I_l}{\sum_j w_j I_j}$
for $s = 1$ to S (число семплов) **do**
    Выбрать кластер $l \sim P(l)$
    Сгенерировать $x_s \sim q_l(x)$
    Вычислить вес: $w_s = \frac{p(x_s)}{q(x_s)}$
    Оценить: $\hat{f}_s = f(x_s) \cdot w_s$
end for
Оценка: $\hat{\mu} = \frac{1}{S} \sum_{s=1}^{S} \hat{f}_s$

4.3 Практические применения

- **Оценка суммы/среднего** по большой базе векторов
- **Ускорение обучения** моделей на больших наборах данных
- **Баейсовский вывод** с PQ-аппроксимацией апостериорного распределения

5 Расширения и вариации

5.1 Residual Quantization

Квантование остатков после первичного квантования:

$$\mathbf{r} = \mathbf{x} - q_{coarse}(\mathbf{x})$$

$$q_{PQ}(\mathbf{r}) = (j_1, ..., j_M)$$

5.2 Multi-Layer PQ

Иерархическая структура для большего сжатия.

5.3 PQ с адаптивными кластерами

Динамическое обновление кодбуков на потоковых данных.

6 Заключение

Product Quantization предоставляет мощный framework для работы с высокоразмерными данными. Комбинация с Importance Sampling открывает дополнительные возможности для эффективной статистической оценки и ускорения вычислений.