Maximum Inner-Product Search Using Cone Trees

Parikshit Ram    Alexander G. Gray

Georgia Institute of Technology
For a given query $q \in \mathbb{R}^d$ and a set $S \in \mathbb{R}^d$ of size $N$, find the point $p \in S$ such that

$$p = \arg \max_{r \in S} \langle q, r \rangle.$$
For a given query $q \in \mathbb{R}^D$ and a set $S \in \mathbb{R}^D$ of size $N$, find the point $p \in S$ such that

$$p = \arg \max_{r \in S} \langle q, r \rangle.$$

- Naive solution takes $N$ operations/query
- $O(N^2)$ operations for $O(N)$ queries
- Our goal is to solve this problem efficiently
is (implicitly) required for....

- Retrieval of Recommendation after Matrix Factorization
- Document Retrieval
- Max-kernel Operation
- Greedy Co-ordinate Descent Optimization
  ◆ SMO Algorithm for SVM training
We present ....

- a method to solve the problem efficiently* in the general setting using a common data structure (*ball-trees*)
- a method to deal with many queries (batch querying) using a new data structure (*cone-trees*)
Distinctions to usual searches
Applications
Branch-and-bound algorithms
Extensions
Existing Problems
For a given query $q \in \mathbb{R}^D$ and a set $S \in \mathbb{R}^D$ of size $N$, find the point $p \in S$ such that

$$p = \arg\min_{r \in S} \|q - r\|_2.$$
For a given query $q \in \mathbb{R}^D$ and a set $S \in \mathbb{R}^D$ of size $N$, find the point $p \in S$ such that

$$p = \arg\min_{r \in S} \| q - r \|_2 .$$

$$\arg\min_r \| q - r \|_2 = \arg\max_r \langle q, r \rangle \quad \text{IF} \quad \| r \| = k \forall r \in S$$

$\Rightarrow$ MIP can be reduced to NNS under some conditions.
Reduction to NNS in metric spaces allows use of existing methods

Without reduction, it is a harder unsolved problem:

- Lack of triangle-inequality
- Lack of co-incidence

So why do we care about the general problem?
Applications
The user $U$, items $I_j$ and the item biases $b_j$.

\[
\begin{bmatrix}
  u_1 \\
u_2 \\
  \vdots \\
  \vdots \\
u_d
\end{bmatrix}
\times
\begin{bmatrix}
  i_{11} & i_{12} & \cdots & \cdots & i_{1d} \\
i_{21} & i_{22} & \cdots & \cdots & i_{2d} \\
  \vdots & \vdots & \ddots & \vdots & \vdots \\
i_{N1} & i_{N2} & \cdots & \cdots & i_{Nd}
\end{bmatrix}
+ \begin{bmatrix}
  b_1 \\
b_2 \\
  \vdots \\
b_N
\end{bmatrix}
\]

The best recommendation for the user $U$ is

\[
\arg\max_{j \in 1, \ldots, n} U^\top I_j + b_j
\]
The user \( U \), items \( I_j \) and the item biases \( b_j \).

\[
\begin{bmatrix}
    u_1 \\
    u_2 \\
    \vdots \\
    \vdots \\
    u_d \\
    1
\end{bmatrix}
\times
\begin{bmatrix}
    i_{11} & i_{12} & \cdots & \cdots & i_{1d} & b_1 \\
    i_{21} & i_{22} & \cdots & \cdots & i_{2d} & b_2 \\
    \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\
    i_{N1} & i_{N2} & \cdots & \cdots & i_{Nd} & b_N
\end{bmatrix}
\]

The best recommendation for the user \( U \) is

\[
\arg \max_{j \in 1, \ldots, n} \tilde{U}^\top \tilde{I}_j
\]
### Table 1: With $l_2$ distance

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$k = 1$</th>
<th>$k = 5$</th>
<th>$k = 10$</th>
<th>$k = 50$</th>
<th>$k = 100$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MovieLens</td>
<td>0.4</td>
<td>0.54</td>
<td>0.59</td>
<td>0.72</td>
<td>0.77</td>
</tr>
<tr>
<td>Netflix</td>
<td>0.19</td>
<td>0.24</td>
<td>0.28</td>
<td>0.35</td>
<td>0.39</td>
</tr>
<tr>
<td>Y! Music</td>
<td>0.055</td>
<td>0.08</td>
<td>0.08</td>
<td>0.112</td>
<td>0.133</td>
</tr>
</tbody>
</table>

### Table 2: With cosine similarity

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$k = 1$</th>
<th>$k = 5$</th>
<th>$k = 10$</th>
<th>$k = 50$</th>
<th>$k = 100$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MovieLens</td>
<td>0.05</td>
<td>0.12</td>
<td>0.16</td>
<td>0.35</td>
<td>0.46</td>
</tr>
<tr>
<td>Netflix</td>
<td>0.14</td>
<td>0.24</td>
<td>0.31</td>
<td>0.48</td>
<td>0.56</td>
</tr>
<tr>
<td>Y! Music</td>
<td>0.004</td>
<td>0.01</td>
<td>0.014</td>
<td>0.033</td>
<td>0.047</td>
</tr>
</tbody>
</table>
- \(p\)-spectrum kernel for string matching
  - \(\|r\|\) corresponds to the length of the string.

- Maximum-matching kernel for string matching
  - \(\|r\|\) corresponds to presence of genetically valuable letters (like W, P, H)
Greedy Coordinate Gradient Descent

Initialize: Set the initial value of $w^0$.

for $t \leftarrow 1, \ldots$ do

\[ j \leftarrow \arg \max_l |\nabla_l \mathcal{L}(w^t)|. \]

\[ w^t \leftarrow w^{t-1} - c \cdot \nabla_l \mathcal{L}(w^t)e_j. \]

end for

Finding the best coordinate reduces to finding $\arg \max_l |\langle x_l, q \rangle|$.

\[
\max_l |\langle x_l, q \rangle| = \max \left( \max_l \langle x_l, q \rangle, -\min \langle x_l, q \rangle \right)
\]

$^1$Dhillon, et. al., 2009
**SMO for linear SVM**

Initialize ∀ k: \( y_k = +1 \), \([A_k, B_k] \leftarrow [0, C] \),
Initialize ∀ k: \( y_k = -1 \), \([A_i, B_i] \leftarrow [-C, 0] \),
∀k, \( \alpha_k \leftarrow 0 \), \( g_k \leftarrow 1 \).

**loop**

\[ i \leftarrow \arg \max_i y_i g_i \text{ subject to } y_i \alpha_i < B_i \]
\[ j \leftarrow \arg \min_j y_j g_j \text{ subject to } y_j \alpha_j > A_j \]

if \( y_i g_i \leq y_j g_j \) then
  stop.
end if

\( \lambda \leftarrow \min \left\{ B_i - y_i \alpha_i, y_j \alpha_j - A_j, \frac{y_i g_i - y_j g_j}{K_{ii} + K_{jj} - 2K_{ij}} \right\} \)
∀ k ∈ \{1, \ldots, n\} \quad g_k \leftarrow g_k - \lambda y_k K_{ik} + \lambda y_k K_{jk}
\( \alpha_i \leftarrow \alpha_i + y_i \lambda \)  \quad \( \alpha_j \leftarrow \alpha_j - y_j \lambda \)

**end loop**

\(^2\)Platt, 1999
Branch-and-bound Algorithms
The tree

Step 1: Index the set $S$ into a ball-tree:
Step 1: Index the set $S$ into a ball-tree:
Step 1: Index the set $S$ into a ball-tree:
Step 2: Upper bound the maximum inner-product for a query $q$ in the ball $B_p^R$:

$$\max_{r \in B_p^R} \langle q, r \rangle \leq \langle q, p \rangle + \|q\| \cdot R.$$
Step 3: Use the depth-first branch-and-bound algorithm:
Step 3: Use the depth-first branch-and-bound algorithm:
Step 3: Use the depth-first branch-and-bound algorithm:
Step 3: Use the depth-first branch-and-bound algorithm:
**Step 3:** Use the depth-first branch-and-bound algorithm:
Step 3: Use the depth-first branch-and-bound algorithm:
Batch-querying for many queries

- Index the queries as well.
- \( p = \arg \max_r \langle q, r \rangle = \arg \max_r \| r \| \cos \theta_{qr} \)
  - \( p \) is independent of \( \| q \| \)
  - \( p \) depends only on the direction of \( q \)
Batch-querying for many queries

Key Idea

Existing Problems

Applications

Branch-and-bound Algorithms

The tree

Query ball bound

Single-tree algorithm

Batch-querying

Cone tree

Cone-ball bound

Dual-tree algorithm

Performance

Artists

Genres
Step 1a: Index the queries into cones (index on the angles):

![Diagram showing cone tree indexing](image-url)
**Step 1a:** Index the queries into cones (index on the angles):
Step 1a: Index the queries into cones (index on the angles):
Step 2: Upper bound the maximum inner-product for any query \( q \in \mathcal{C}_{\omega q}^{w q} \) and the ball \( \mathcal{B}_{\mathcal{P}_{0}}^{R} \):

\[
\max_{q \in \mathcal{C}_{\omega q}^{w q}, p \in \mathcal{B}_{\mathcal{P}_{0}}^{R p}} \langle q, p \rangle \leq \|p_{0}\| \cos(\{|\phi| - \omega_q\}+) + R_p.
\]
Step 3: Use the dual-tree branch-and-bound algorithm:

Level 1 interaction
Step 3: Use the dual-tree branch-and-bound algorithm:

Level 2 interaction
Our methods

Key Idea
provide ...

- a simple solution to the general problem
- a solution for efficient batch-querying
Our methods

<table>
<thead>
<tr>
<th>Key Idea</th>
</tr>
</thead>
<tbody>
<tr>
<td>a simple solution to the general problem</td>
</tr>
<tr>
<td>a solution for efficient batch-querying</td>
</tr>
<tr>
<td>and can easily extend to ....</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>weighted maximum-inner product search</td>
</tr>
<tr>
<td>additive: ( \arg \max_{p \in S} \langle q, p \rangle + b(r) )</td>
</tr>
<tr>
<td>multiplicative: ( \arg \max_{p \in S} c(p) \cdot \langle q, p \rangle )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Branch-and-bound Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>a more general problem of max-kernel operation with any Mercer-kernel</td>
</tr>
<tr>
<td>more efficiency via controlled approximation</td>
</tr>
</tbody>
</table>
- Pseudo code (very simple) presented in the paper.
- Actual code to be available soon at http://mlpack.org