Privacy-aware Document Ranking with Neural Signals

Jinjin Shao, Shiyu Ji, Tao Yang
Department of Computer Science
University of California, Santa Barbara
United States
Challenge for Private Ranking

Client uploads encrypted documents and index, utilizing its massive storage and computing power.

Server is honest-but-curious, i.e., correctly executes protocols but observes/infers private information.

Challenges for Private Search:

• Feature leakage (e.g., term frequency) can lead to plaintext leakage.
• Crypto-heavy techniques are too expensive.
Related Work for Private Search

- **Searchable Encryption** [Curtmola et al. Crypto06, Cash et al. Crypto13] does not support ranking.
- **Private Additive Ranking** [Xia et al. TPDS16] works for small datasets only [Agun et al. WWW18] only supports partial cloud ranking.
- **Private Tree-based Ranking** [Bost et al. NDSS15] uses computational-heavy techniques such as Homomorphic Encryption, [Ji et al. SIGIR18] does not support neural signals.
Neural Ranking Models for Ad-hoc Search

Two categories of neural ranking models:
• Representation-based
• Interaction-based

Interaction-based models outperform in TREC relevance benchmarks:
• Guo et al. CIKM16, Xiong et al. SIGIR17, Dai et al., WSDM18

Steps of interaction-based neural ranking:
• Pairwise interaction of query and document terms
• Kernel vector derivation from interaction matrices
• Forward neural network calculation
Leakage in Interaction-based Neural Ranking

Document $m$ terms

Interact

Query $n$ terms

Similarity Matrix $m \times n$ real values

Kernel Comp.

Kernel Vector $n \times R$ real values

Term Frequency / Term Co-occurrence

Plaintext attack

[Islam et al. NDSS12, Cash et al. CCS15]
Leakage in Interaction-based Neural Ranking

Document $m$ terms

Query $n$ terms

*Interact*

Similarity Matrix $m \times n$ real values

*Kernel Comp.*

Kernel Vector $n \times R$ real values

*Forward Network Calculation*

1. Pre-compute kernel vectors with *closed soft match map*.
2. Hide exact match signal and obfuscate kernel values.

Term Frequency / Term Co-occurrence
How Kernel Values Leak Term Frequency

\[ \left\{ \sum_{t \in q} \log K_1(t, d), \sum_{t \in q} \log K_2(t, d), \ldots, \sum_{t \in q} \log K_R(t, d) \right\} \]

\( K_i(t, d) \) is the \( i \)-th kernel value on the interaction of a possible query term \( t \) and document \( d \), representing semantic similarity. [Xiong et al. SIGIR17]

**Decompose kernel values** into two parts:
- \( K_1(t, d), \ldots, K_{R-1}(t, d) \) *Soft Match Signals*
- \( K_R(t, d) \) *Exact Match Signal*

**Our analysis:** Term frequency of \( t \) in \( d \) can be well approximated by \( K_R(t, d) \).

**Solution for privacy-preserving:** Replace \( K_R(t, d) \) with relevance scores from private tree ensemble.
How to Hide/Approximate Exact Match Signal

Propose privacy-preserving approach:
Use private tree ensemble, with encrypted features, and compute a relevance score. [Ji et al., SIGIR18]

$$\log K_R(t, d), t \in q$$

Kernel Vector

Encrypted features, e.g., Term frequency, proximity, and page quality score.

Approximated Kernel Vector
Closed Soft Match Map in Detail

Motivation for Soft Match

- Limit precomputing. Avoid to compute all possible pairs of terms and documents.
- Otherwise, 1 million docs cost ~10TB storage.
- **Basic idea**: Precompute kernel values only for term $t$ and document $d$, if $t$ appears in $d$ $t$ is soft-relevant to $d$.

Closed Soft Match:

- For two terms $t_0$ and $t_1$ if
  1) $(t_0, d)$ is in a closed soft match map and
  2) $t_0$ and $t_1$ are similar, then $(t_1, d)$ is in that map. **Build closed soft match map with clustering**
- **Privacy advantage**: Prevent leaking term occurrence to the server (shown later).
Build Closed Soft Match Map with Clustering

If a term \( t_0 \) is in a \( \tau \)-similar term closure, there exists a term \( t_1 \), \( \text{sim}(t_0, t_1) \geq \tau \).

**Fixed-threshold Clustering:**
Apply a uniform \( \tau \) for all closures.

**Weakness:** Closures can include
1) too many terms, which incurs huge storage cost;
2) too few terms, which leads to high privacy leakage.

\[
\begin{align*}
\text{Sim}(A, B) &= 0.763 \\
\text{Sim}(B, C) &= 0.722 \\
\text{Sim}(D, E) &= 0.601 \\
\text{Sim}(B, D) &= 0.531 \\
\text{Sim}(E, F) &= 0.513 \\
\text{Sim}(F, G) &= 0.481 \\
\text{Sim}(C, F) &= 0.467
\end{align*}
\]
Build Closed Soft Match Map with Clustering

If a term $t_0$ is in a $\tau$-similar term closure, there exists a term $t_1$, $sim(t_0, t_1) \geq \tau$.

Adaptive Clustering:
Given a closure minimum size $p$, and maximum size $x$, apply a series of decreasing thresholds: $\tau_1 > \tau_2 \ldots > \tau_m$, to gradually expand all term closures, such that in the end, all closures are of size between $p$ and $x$.

```
Threshold 1: 0.7
Threshold 2: 0.4
Size target: [3, 4]
```
Privacy Property of Closed Soft Match Map

**Objective:** Given a closed soft match map, show that a server adversary is unlikely to learn term frequency/occurrence of dataset $D$.

**How to prove:** There are too many different datasets $D'$ whose soft match maps, compared to $D$,  
• have the same set of keys (guaranteed by Closed Soft Match Map);  
• have indistinguishable kernel values.  
The cloud server is unlikely to differentiate them.

**How to produce those many datasets:**  
• Use *closure-based transformation*. 
Closure-based Transformation: Produce Indistinguishable Datasets

Step 1: For each document $d$, partition all terms in $d$ into different groups such that terms in each group belong to the same term closure.

Step 2: For each term group in $d$, replace that group with any nonempty subset of the term closure associated with that group.

Document $d = \{t_1, t_2, t_3, t_4, t_5, t_6\}$

Document $d' = \{t_1, t_2, t_7, t_4, t_5, t_8, t_9\}$

Term closure $\{t_1, t_3, t_6, t_7, t_8, t_9\}$

Note: Server only knows hashed term ids in each term closure, but not their meanings and their individual statistical info.

Statistical distance between kernel values of $d$ and $d'$ with respect to a term can be very small.
**Definition: ε-statistically indistinguishable**

Kernel values of a term $t$ in document $d$ and its transformation $d'$:

$$\hat{f}_{t,d} = (a_1, a_2, a_3, ..., a_{R-1}), \quad \hat{f}_{t,d'} = (a'_1, a'_2, a'_3, ..., a'_{R-1}).$$

$$\varepsilon \geq \text{Statistical Distance}(\hat{f}_{t,d}, \hat{f}_{t,d'}) = \frac{1}{2} \sum_{i=1}^{R-1} |a_i - a'_i|,$$

True for all corresponding document $d$ and its transformation $d'$ with all terms.

**Takeaway:**

\[ \downarrow \varepsilon \longrightarrow \downarrow \text{Prob}\text{(successfully differentiate d from d')} \]
How to Minimize Statistical Dist. \((\vec{f}_{t,d}, \vec{f}_{t,d'})\)

**Kernel Value Obfuscation**

For the \(j\)-th soft kernel value in the kernel value vector:

\[
a_j = \begin{cases} 
\lfloor \log_r K_j(t, d) \rfloor, & \text{if } K_j(t, d) > 1, \\
1, & \text{otherwise},
\end{cases}
\]

where \(r\) is a privacy parameter, \(t\) is a term, \(d\) is a document.

**Trade-off between Privacy and Ranking Accuracy:**

\[\uparrow r \quad \Downarrow \text{Statistical Dist.} \quad \Downarrow \text{Effectiveness of Soft Match Signals} \quad \uparrow \text{Privacy Guarantee}\]
Datasets and Evaluation Objectives

✓ Robust04: ~0.5 million docs with 250 queries.
✓ ClueWeb09-Cat-B: ~50 million docs with 150 queries from Web 09-11.

• Evaluation Objectives:
  1. Can kernel vectors approximated with private tree ensemble rank well?
  2. Can kernel value obfuscation, preserve the ranking accuracy?
  3. How effective are two different methods of clustering term closures for closed soft match maps?
## Evaluation on Approx. Exact Match Signal

<table>
<thead>
<tr>
<th>Model</th>
<th>ClueWeb09-Cat-B</th>
<th>Robuts04</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NDCG@1</td>
<td>NDCG@3</td>
</tr>
<tr>
<td>LambdaMART</td>
<td>0.2893</td>
<td>0.2828</td>
</tr>
<tr>
<td>DRMM</td>
<td>0.2586</td>
<td>0.2659</td>
</tr>
<tr>
<td>KNRM</td>
<td>0.2663</td>
<td>0.2739</td>
</tr>
<tr>
<td>C-KNRM</td>
<td>0.3155</td>
<td>0.3124</td>
</tr>
<tr>
<td>C-KNRM*</td>
<td>0.2884</td>
<td>0.2927</td>
</tr>
<tr>
<td>C-KNRM*/T</td>
<td><strong>0.3175</strong></td>
<td><strong>0.3122</strong></td>
</tr>
</tbody>
</table>

C-KNRM is CONV-KNRM [Dai et al. WSDM18]  
C-KNRM* is C-KNRM without bigram-bigram interaction.  
C-KNRM*/T is C-KNRM* with private tree ensemble.  
**Takeaway:** Tree signal integration for neural kernel vectors can rank well, and even boost ranking performance.
## Evaluation on Kernel Value Obfuscation

<table>
<thead>
<tr>
<th>Model</th>
<th>ClueWeb09-Cat-B</th>
<th>Robuts04</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NDCG@1</td>
<td>NDCG@3</td>
</tr>
<tr>
<td>C-KNRM</td>
<td>0.3155</td>
<td>0.3124</td>
</tr>
<tr>
<td>C-KNRM*</td>
<td>0.2884</td>
<td>0.2927</td>
</tr>
<tr>
<td>C-KNRM*/TO No Obfuscation</td>
<td>0.3175</td>
<td>0.3122</td>
</tr>
<tr>
<td>C-KNRM*/TO r = 5</td>
<td>0.3178</td>
<td>0.3067</td>
</tr>
<tr>
<td>C-KNRM*/TO r = 10</td>
<td>0.3121</td>
<td>0.3097</td>
</tr>
</tbody>
</table>

C-KNRM*/TO is C-KNRM* with private tree ensemble and kernel value obfuscation.

**Takeaway:** Kernel value obfuscation can result in small degradation (~1.6% for NDCG@1 in ClueWeb) on ranking performance, when $r = 10$. 
## Evaluation on Term Clustering Methods

<table>
<thead>
<tr>
<th>Clustering Method</th>
<th>ClueWeb09-Cat-B NDCG@1</th>
<th>ClueWeb09-Cat-B NDCG@3</th>
<th>ClueWeb09-Cat-B NDCG@10 (Storage)</th>
<th>Robuts04 NDCG@1</th>
<th>Robuts04 NDCG@3</th>
<th>Robuts04 NDCG@10 (Storage)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-KNRM</td>
<td>0.3155</td>
<td>0.3124</td>
<td>0.3085</td>
<td>0.5373</td>
<td>0.4875</td>
<td>0.4586</td>
</tr>
<tr>
<td>Fixed $\tau = 0.3$</td>
<td>0.3136</td>
<td>0.3078</td>
<td>0.3091 (1700 TB)</td>
<td>0.5225</td>
<td>0.4974</td>
<td>0.4621 (45 TB)</td>
</tr>
<tr>
<td>Fixed $\tau = 0.7$</td>
<td>0.3064</td>
<td>0.3048</td>
<td>0.3104 (16 TB)</td>
<td>0.4886</td>
<td>0.4644</td>
<td>0.4169 (0.3 TB)</td>
</tr>
<tr>
<td>Adaptive $\tau = 0.3$</td>
<td>0.3052</td>
<td>0.3069</td>
<td>0.3120 (46 TB)</td>
<td>0.5127</td>
<td>0.4892</td>
<td>0.4582 (1 TB)</td>
</tr>
<tr>
<td>Adaptive $\tau = 0.7$</td>
<td>0.3067</td>
<td>0.3012</td>
<td>0.3060 (7.6 TB)</td>
<td>0.4899</td>
<td>0.4608</td>
<td>0.4090 (0.3 TB)</td>
</tr>
</tbody>
</table>


Use C-KNRM*/TOC here: private tree ensemble, kernel value obfuscation, and closed soft map.

**Takeaway:** 1) Clustering threshold choices has impact on relevance. 2) Adaptive clustering is competitive with up to $\sim$40x storage cost saving
Concluding Remarks

- **Contribution**: A privacy-aware neural ranking for this open problem.
- **Evaluation results** with two datasets
  - NDCG can be improved by approximating the exact match kernel of neural ranking with a tree ensemble.
  - Kernel value obfuscation on soft match signals does carry a modest relevancy trade-off for privacy.
  - Adaptive clustering for term closures significantly reduce storage demand.