Efficient Private Embedding Lookup via Independent Vector Evaluation

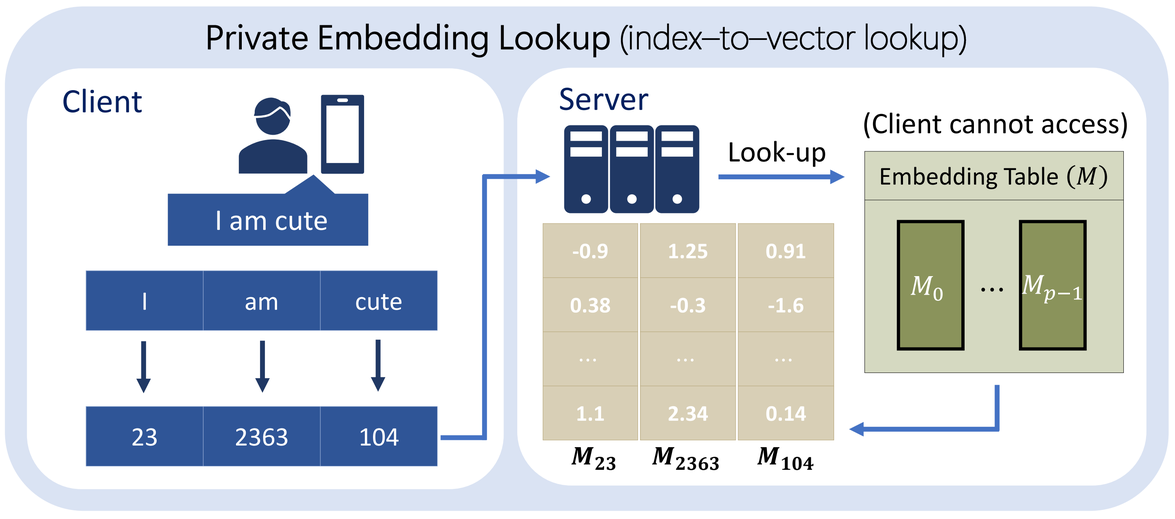
Daehyun Jang[1], Jaehee Kang[1], Hanee Rhee[1], Jung Hee Cheon[1,2]
[1]Seoul National University, Seoul [2]CryptoLab Inc., Seoul

Introduction

- FHE enables PPML for MLaaS with low client overhead.
- Private Embedding Lookup is a key challenge in NLP models.

Encrypted index $j \in [p]$ —Look-up→ Embedding vector $M_j \in \mathbb{R}^d$

- We target the single-index regime for low communication cost.
- Idea : Replace one-hot synthesis [1] with linearly independent vector evaluation, yielding efficient private embedding lookup.



Prior Approach: One-Hot Synthesis

Prior work [1] constructs a one-hot vector from the encrypted index j and multiplies it with the embedding matrix M:

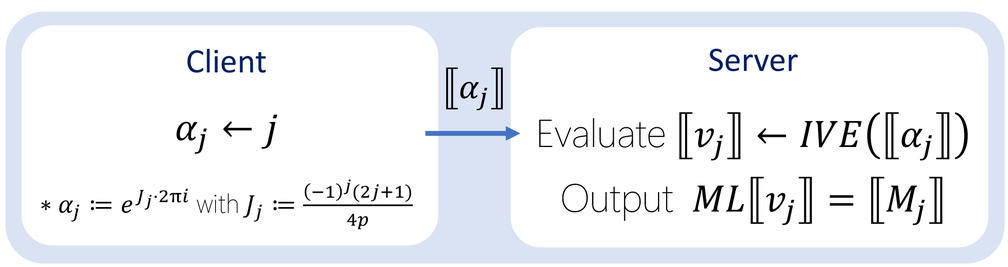
(Server)
Evaluate $[\![e_j]\!] \leftarrow OneHot([\![j]\!])$
Output $M[\![e_j]\!] = [\![M_j]\!]$

∗ Bottleneck: Making One-Hot uses many multiplications.
(making encrypted vector converge to one-hot vector)

→ High Depth, Poor Latency

Our Solution: Independent Vector Evaluation (IVE)

∗ Core Idea: Replace one-hot vectors with another basis v_j and a basis change matrix L.



We adopt a Discrete Cosine Transform (DCT) matrix $D \in \mathbb{R}^{p \times p}$ [2]:

- Orthogonal Matrix (Numerical Stability)
- Efficient Computability

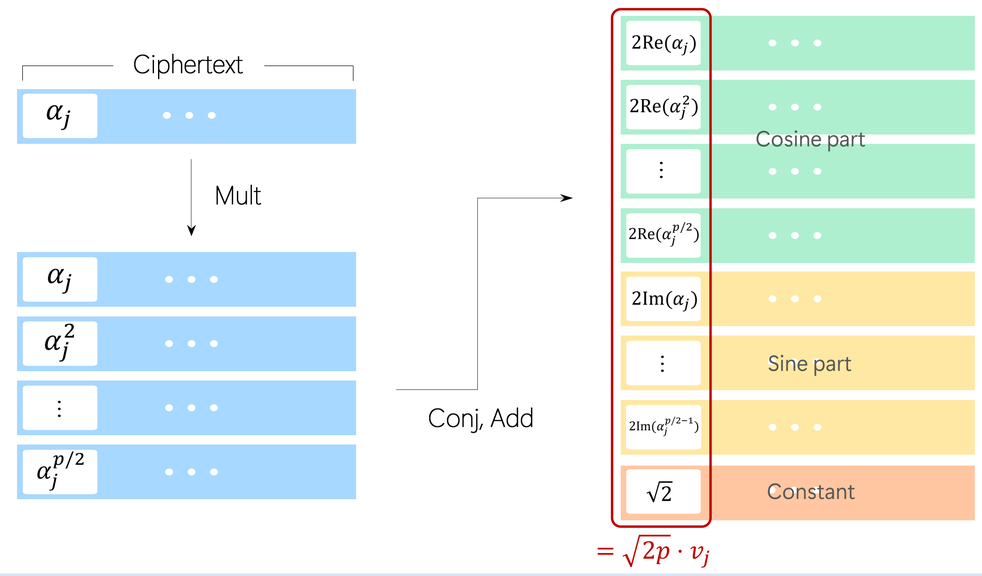
(row-permuted) DCT Matrix

$$D_{k,j} = \begin{cases} \sqrt{\frac{2}{p}} \cos\left(\frac{(-1)^j(2j+1)\pi}{2p}(k+1)\right) & 0 \le k < \frac{p}{2} \\ \sqrt{\frac{2}{p}} \sin\left(\frac{(-1)^j(2j+1)\pi}{2p}\left(k-\frac{p}{2}+1\right)\right) & \frac{p}{2} \le k < p-1 \\ \frac{1}{\sqrt{p}} & k = p-1 \end{cases}$$

$$D = \begin{bmatrix} | & | & & | \\ v_0 & v_1 & \cdots & v_{p-1} \\ | & | & & | \end{bmatrix}$$

- v_j: j-th column of D
- $L := D^{-1}$

Independent Vector Evaluation (IVE)



Comparison of IVE with [1]

∗ p: index size

- Complexity: $O(p \log p)$ [1] vs $O(p)$ (Ours)
- Depth Consumption:

$\log p$	2	4	6	8	10
Depth [1]	11	15	19	23	28
Depth (Ours)	1	3	5	7	9

Implementation Results

∗ Setup: GloVe42B.300d Dataset / C++ HEaaN Library / Intel Xeon Gold 6542 / 1-threaded CPU

- We implement both our method and [1] using PCMM [3].
- We compress the GloVe42B.300d table [4]; A single table lookup is replaced by four $\log p$-bit-to-300d lookup calls.

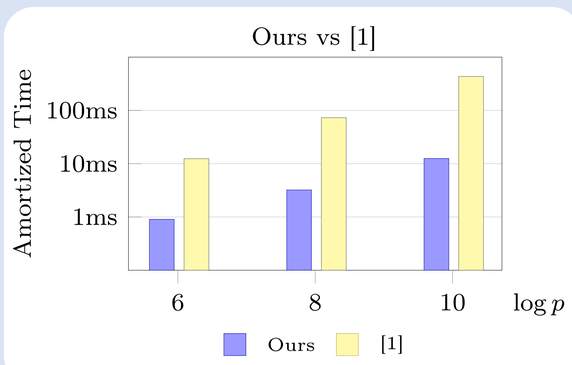
∗ Implementation 1: Private Embedding Lookup – Ours vs [1]

- Ours is up to 34.6x faster than [1].

$\log N = 17, \log \Delta = 51$ $\log PQ = 2070$

$\log p$	Method	VecGen	PCMM	Total
6	Ours	0.2299	0.6723	0.9022
	[1]	11.6995	0.6948	12.3943
8	Ours	0.9388	2.2757	3.2145
	[1]	70.6454	2.4227	73.0681
10	Ours	3.7677	8.7470	12.5147
	[1]	423.5058	9.5222	433.0280

* Amortized Time (ms)
* VecGen: Either IVE (ours) or One-Hot Synthesis ([1])



∗ Implementation 2: Ours + ExpBTS

$[\![\alpha_j]\!]$ can be generated from $[\![J_j]\!]$ via functional bootstrapping ExpBTS [5]. This reduces communication cost.

- Amortized time: 11.2ms for $\log p = 10$
- Ciphertext expansion ratio: up to 3.9x

$\log N = 16$, $\log \Delta = 49$ $\log PQ = 1424$, target precision: 16 bit

$\log p$	ExpBTS	IVE	PCMM	Total	CommOver
6	0.7795	0.1744	0.6947	1.6486	5.839
8	0.7437	0.5528	2.2413	3.5378	4.629
10	0.7670	1.7458	8.7133	11.2261	3.903

* Amortized Time (ms)
* CommOver: Ciphertext expansion ratio

References

[1] Kim et al., **Privacy-Preserving Embedding via Look-Up Table Evaluation with Fully Homomorphic Encryption**. (ICML, 2024)
[2] Ahmed et al., **Discrete Cosine Transform**. (IEEE Transactions on Computers, 1974)
[3] Bae et al., **Plaintext-Ciphertext Matrix Multiplication and FHE Bootstrapping: Fast and Fused**. (Crypto, 2024)
[4] Shu & Nakayama, **Compressing Word Embeddings via Deep Compositional Code Learning**. (ICLR, 2018)
[5] Bae et al., **Bootstrapping Bits with CKKS**. (Eurocrypt, 2024)