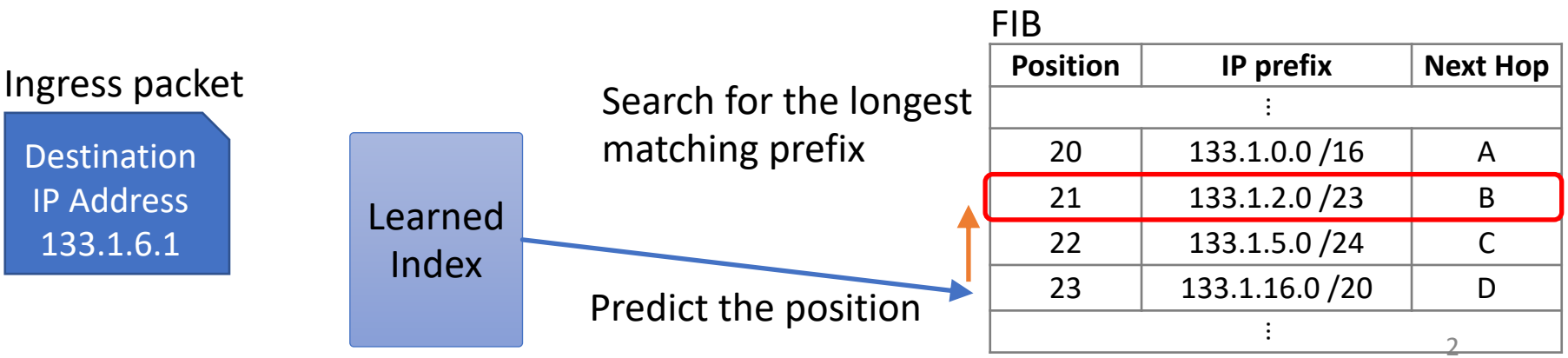
Feasibility of Longest Prefix Matching using Learned Index Structures

◎Shunsuke Higuchi(Osaka univ)
Junji Takemasa (KDDI Research)
Yuki Koizumi (Osaka univ)
Atsushi Tagami(KDDI Research)
Toru Hasegawa (Osaka univ)

Longest Prefix Matching Using Learned Index

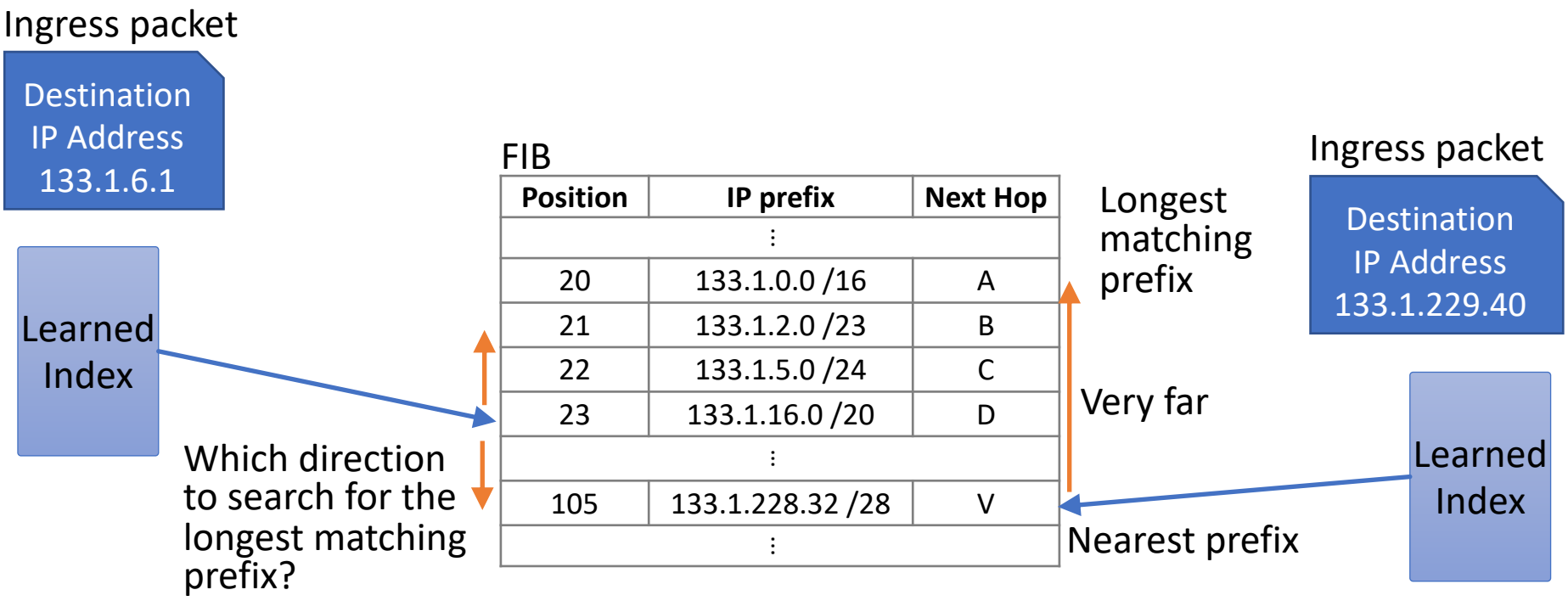
- IP forwarding is longest prefix matching (LPM)
  - Searching a forwarding information base (FIB) for the longest prefixes that match the destination IP addresses
- LPM using learned index
  - Learned index [1]
    - Learning key-position relationships using machine learning (e.g., neural network)
  - Design
    - Sort IP prefixes in ascending order and store them in an array
    - Learn the relation between each IP prefix and its position with a neural network
    - Predict the position of the destination IP address of an ingress packet with the neural network
    - Search for the longest matching prefix around the predicted position



[1] T. Kraska, et al., "The case for learned index structures," in Proceedings of International Conference on Management of Data, June 2018.

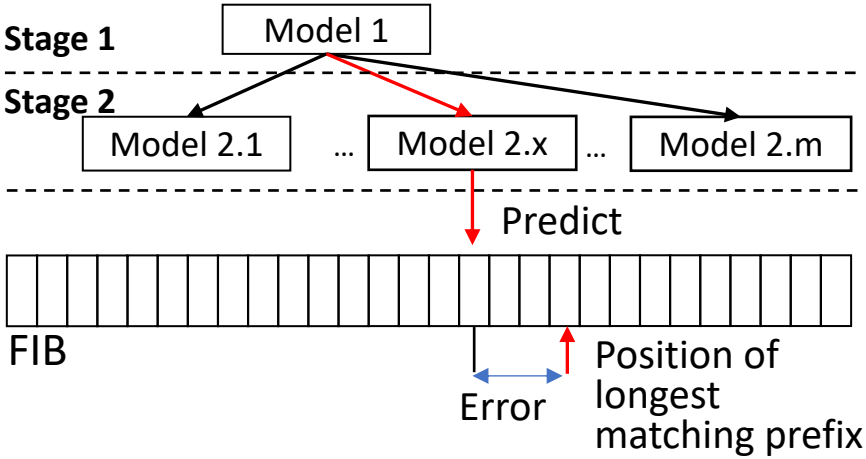
Challenge 1: How to Realize LPM

1. It is difficult to determine which direction the longest matching prefix exists
2. The longest matching prefix sometimes exists very far from the IP prefix that is nearest to the ingress IP address



Challenge 2: Time and Space Complexity

- Hierarchical learned index
  - A single neural network alone cannot predict positions of entries accurately [1].
    1. The model in the first stage roughly predicts positions
    2. Models in the second stage precisely predicts positions



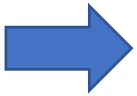
- Challenge
  - How to reduce the computation time of the hierarchical learned index
  - How to reduce the memory size of the hierarchical learned index

Solution 1: LPM to Exact Matching

- Core Idea and approach
  - LPM is reduced to exact matching if an FIB maintains next-hop information for every IP address
  - Aggregate FIB entries according to the next hop information of consecutive entries
- Construction
  - Create an FIB that holds the next hop for all IP addresses
  - Aggregate consecutive entries whose next hops are the same
- Searching
  - Predict the position of the destination IP address of an ingress packet with the neural network
  - Search for the nearest IP addresses in the FIB

Before

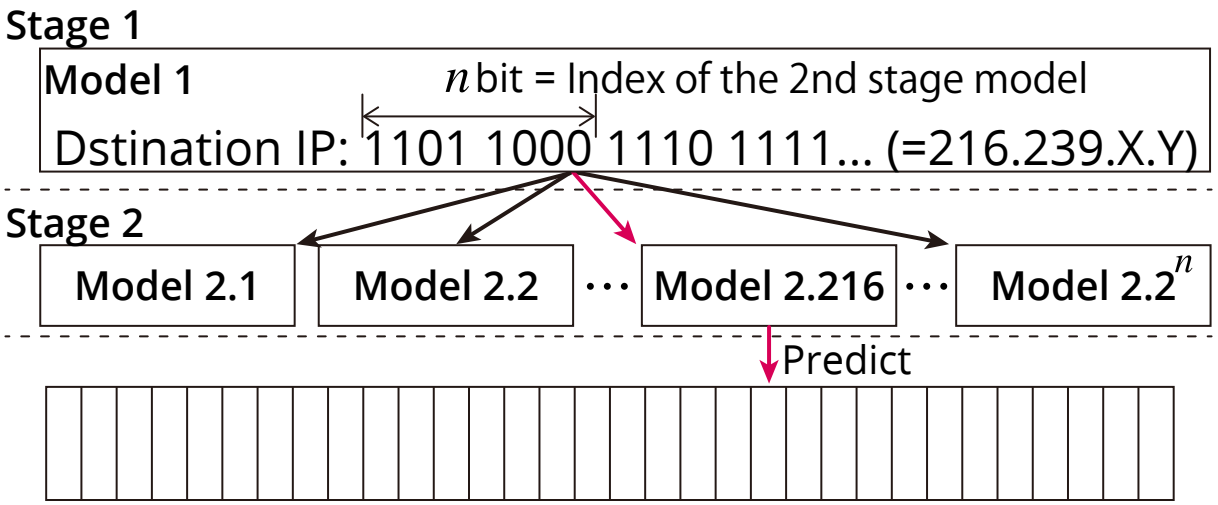
Position	IP prefix	Next Hop
⋮		
20	133.1.0.0 /16	A
21	133.1.2.0 /23	B
22	133.1.5.0 /24	C
23	133.1.16.0 /20	D
⋮		



Position	IP address	Next Hop
⋮		
x	133.1.0.0	A
x+1	133.1.2.0	B
x+2	133.1.4.0	A
x+3	133.1.5.0	C
⋮		

Solution 2: Fast Computation and Small Table

- Accelerate the computing speed of the 1st stage model
  - Use the significant n bits to determine the 2nd stage model, instead of using a neural network
- Accelerate the computing speed and reduce the size of the 2nd stage model
  - Use as few neurons in a layer and hidden layers as possible



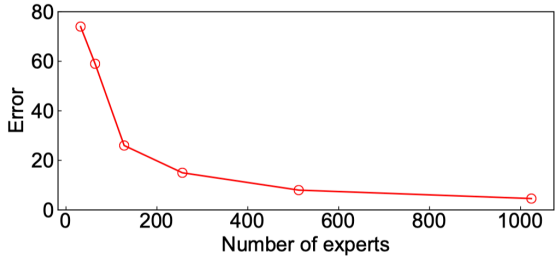
Evaluation Purpose and Condition

- Evaluation Purpose
  - Investigate a sub-optimal parameters of the learned index for FIBs
  - Compare the FIB based on the learned index with the FIB based on level compression trie (LC-trie) [2]
- Evaluation Conditions
  - Data
    - BGP routing information base (RIB) snapshot on Nov. 20th, 2019 [3]
    - The next-hop information is computed according to the AS relationship [4]
  - Measurement platform
    - Xeon Gold 6126 CPU (2.4GHz x 12 CPU cores, 32KB L1D, 1MB L2 cache)

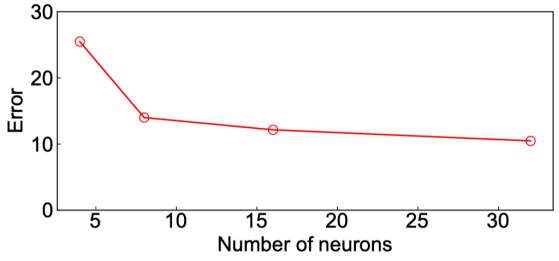
[2] S. Nilsson and G. Karlsson. IP-address lookup using LC-tries. IEEE Journal on Selected Areas in Communications, 17(6):1083–1092, June 1999.
[3] University of Oregon Route Views Project. http://www.routeviews.org/routeviews/.
[4] CAIDA. https://www.caida.org/home/.

Accuracy of the Learned Index

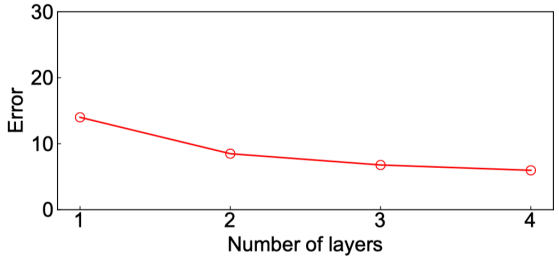
- Varying parameters and their default values
  - The number of models in the 2nd stage: 256
  - The number of hidden layers: 3
  - The number of neurons 8
- The number of expert models
  - The improvement in the error is saturated for more than 256 models
- The number of neurons
  - The improvement in the error is saturated around 8 neurons
- The number of hidden layers
  - The improvement in the error is saturated around 2 layers
  - The difference of the error between the 1 layer case and the 2 layers case is small



(a) Error vs. expert models



(b) Error vs. neurons



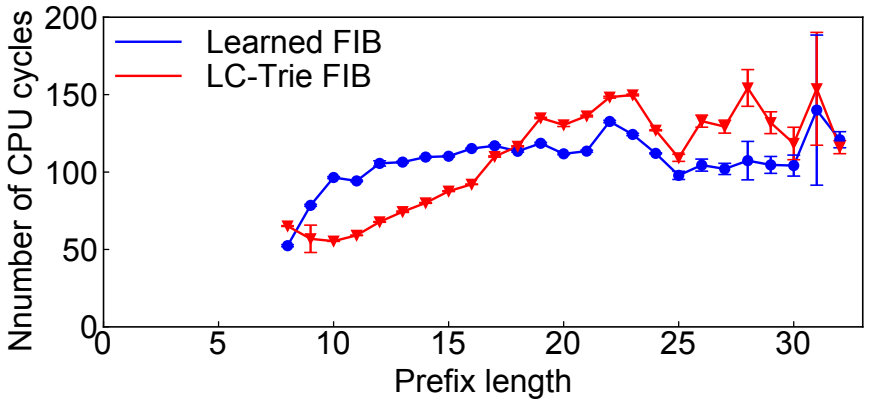
(c) Error vs. hidden layers

Computation Time and Memory Size

- Computation time: The number of CPU cycles required to find the next hop for one IP address
  - LC-Trie FIB: 112 CPU cycles
  - Learned FIB: 115 CPU cycles
- Memory size
  - LC-Trie FIB: 2.504 Mbytes
    - The size of each vertex is 32 bits and the number of vertices is 534,093
    - LC-Trie FIB must maintain a bit sequence for validating skipped bits, and the number of bits for the skip bit validation is 6,772 bytes
    - The total size of fields for next-hop information is 360,879 bytes
  - Learned FIB: 1.173 Mbytes
    - The size of each key is 4 bytes, and FIB has 229,454 entries
    - The size of each weight in the neural network is 4 bytes, and the number of weights is 6,400
    - The total size of fields for next-hop information is 360,879 bytes

Prefix Length vs. Computation Time

- Analysis
  - We investigate dependency between computation time and matched prefix length
  - Figure presents the average number of CPU cycles for each length of matched prefixes with 95% confidence intervals
- Result
  - Learned FIB: The computation time does not depends on the length of prefixes
  - LC-trie FIB: The computation time increases as the length of prefixes increases



Conclusion

- Challenges for realizing an IP FIB using learned index
  1. Learned index supports exact matching while IP forwarding requires longest prefix matching
  2. How to reduce the computation time and the memory size
- Approach
  1. Reducing longest prefix matching to exact matching by aggregating FIB entries according to the next hop information of consecutive entries
  2. Using the first n bits (5 bits) to determine the 2nd stage models in the hierarchical learned index
- Performance
  - Learned FIB is half as small as LC-trie FIB while it achieves the same computation time
  - Computation time of the Learned FIB is independent of the prefix length unlike existing FIBs

Appendix

WHAT IS THE LEARNED INDEX ?

- KVS(Key-Value Store)
  - A datastore where the value associated with the key is called by specifying the key
- Applying the Learned Index to KVS
  - Table are sorted in ascending order of keys
  - Learning key-position relationships using neural network
  - Value is gotten from the predicted Position

Position	Key	Value
0	6	A
1	14	B
2	32	a
3	59	D
4	83	f
⋮	⋮	⋮

Sorted in ascending order of keys

Learnning
key-position relationship

Learned model

WHAT IS THE LEARNED INDEX ?

- Merit and Demerit
  - Merit: Position can be predicted by sum-of-products operation on Key
  - Demerit: There is an error in the predicted position
- Error
  - $|predicted\ position - actual\ position|$