Handling Noise and Metric Issue in Few-Shot Learning Tasks with In-Memory Search

Speaker : B11901027 王仁軒
Mentor : Rick Huang
Advisor: Prof. An-Yeu (Andy) Wu
Date : 2025/06/17

Outline

- Method to tackle analog non-idealities in IMS architecture
  - Noise-aware training
  - Bayesian neural network

Part 1

- Comparison between metrics
  - Angular metric (Cosine)
  - Spatial metric (L_1, L_∞)

Part 2

- Locality sensitive hashing
  - Approximation of cosine similarity
  - 2-D projection of high-dimensional embedding with t-SNE

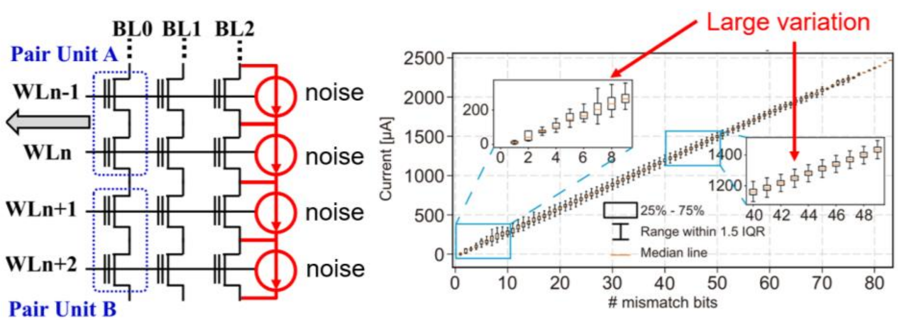
Part 3

- View metric difference as noise
  - Approximation of cosine similarity
  - 2-D projection of high-dimensional embedding with t-SNE

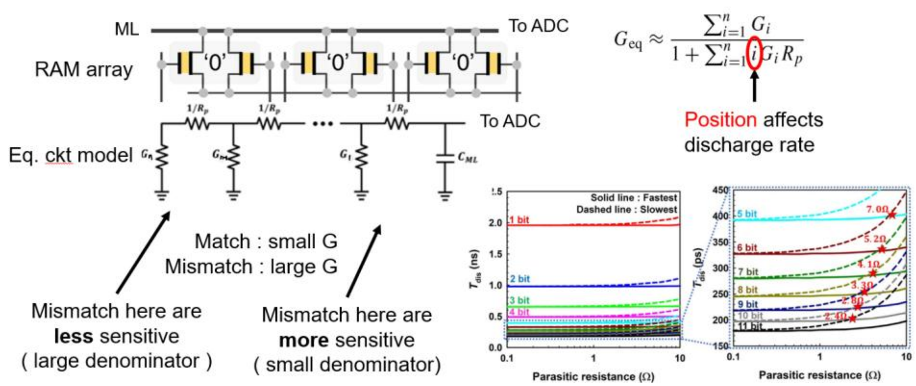
Part 4

Analog Non-Ideal Effects of TCAM

- TCAM : Ternary content addressable memory
- Analog non-ideal effects of in-memory-search



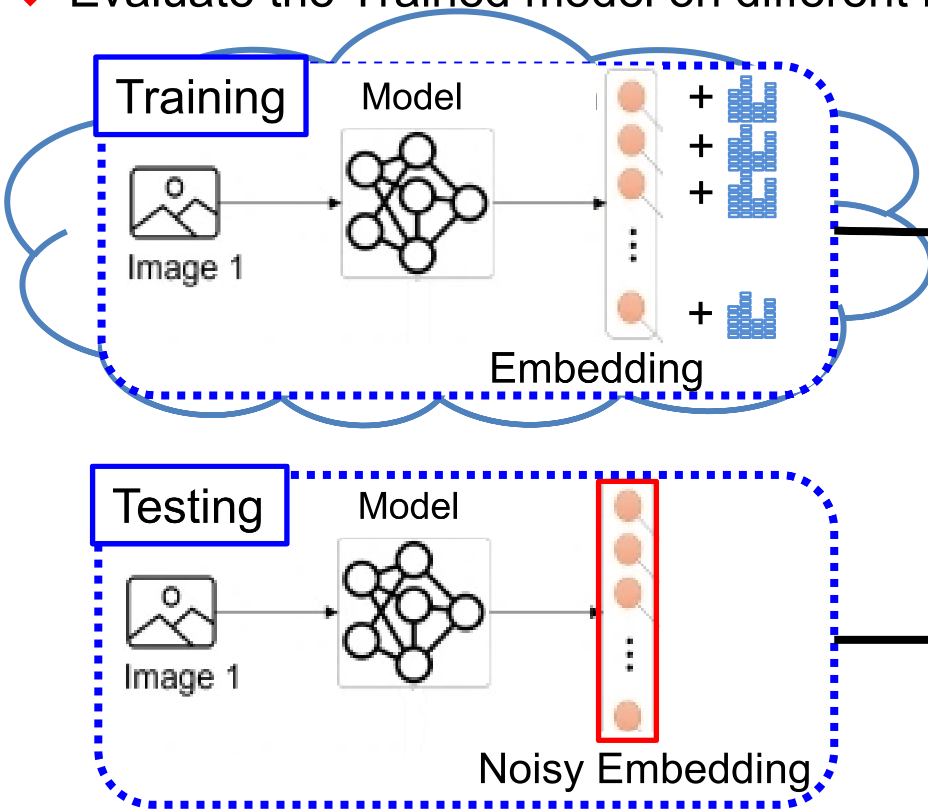
Noise from memory device
- Thermal noise
- Flicker noise
- Leakage current



Parasitic effects of lump elements
- Resistance
- Capacitance

Method 1 : Noise-Aware Training

- Noise-Aware Training
  - Inject simulated noise into embeddings while training the model
  - Evaluate the Trained model on different noise levels

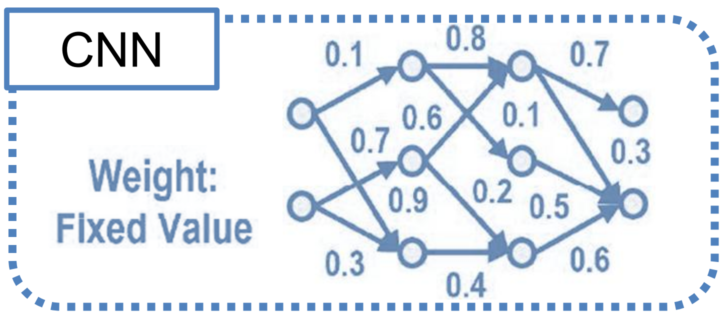


Train on cloud server
Simulate noisy device by injecting noise to embeddings

~~Deploy the model to device~~
Simulate the device on server

Method 2 : Bayesian Neural Network

- Bayesian Neural Network (BNN)
  - Train a robust model that **embraces noise**
  - BNN minimizes KL-divergence (maximize Evidence Lower Bound, ELBO)



Loss : Cross Entropy

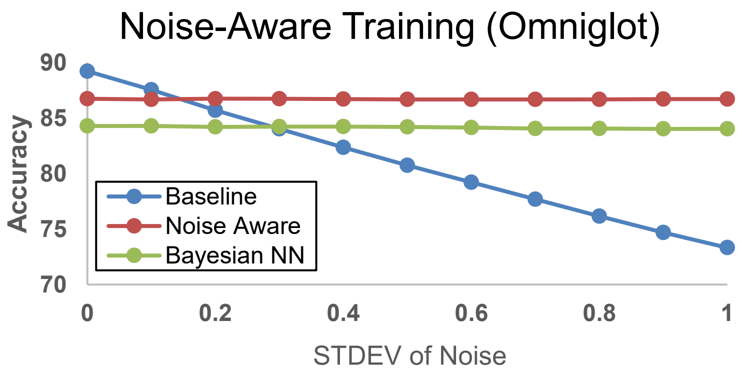
$$\sum -\mathrm{P(D)}\log \mathrm{P(W)}$$

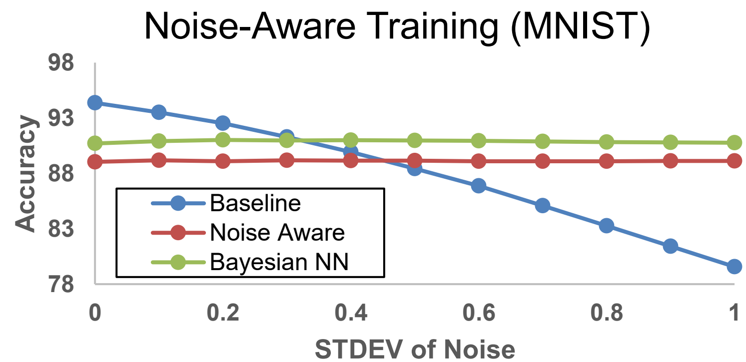
Loss : KL-divergence

$$\underbrace{\frac{1}{K}\sum_{k=1}^{K}\sum -\mathrm{f(D)}\log \mathrm{f(W)}}_{\text{Mean of cross entropy loss acorss samples}} + \underbrace{\beta \cdot KL(P(W)|\mathrm{Normal})}_{\text{Ensure robustness against noise}}$$

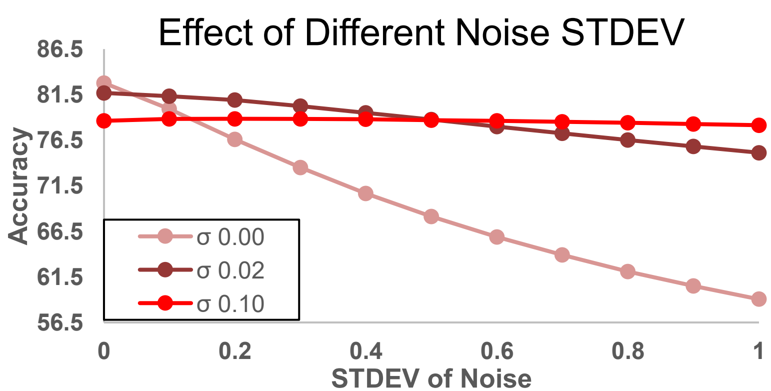
Robustness Against Noise

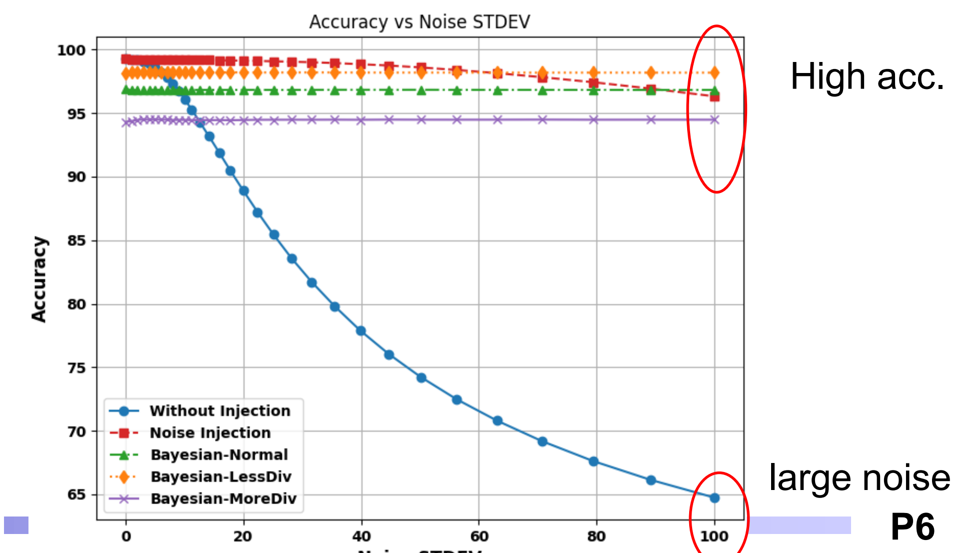
- Both method works well on different datasets
  - Trade-off between accuracy on clean data & noise tolerance





- Tolerance against large noise
  - Little noise has great effect

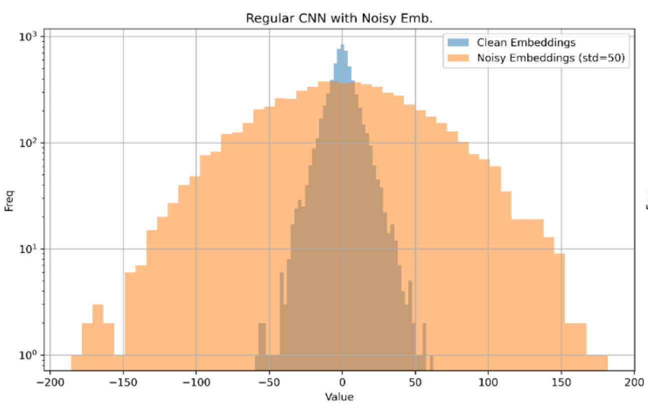




Origin of Noise Resilience in NN

- Collect the value of every embeddings
  - Blue : Original embedding value distribution
  - Orange : New distribution on simulated noisy device
  - Model learns to against noise by amplifying magnitude of embeddings

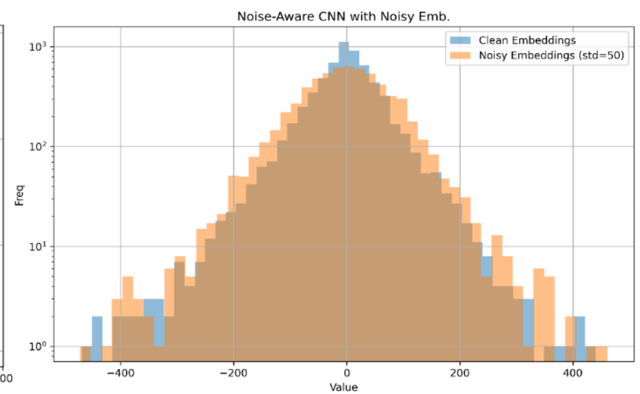
Baseline



27 % accuracy

Noise dominates the embedding

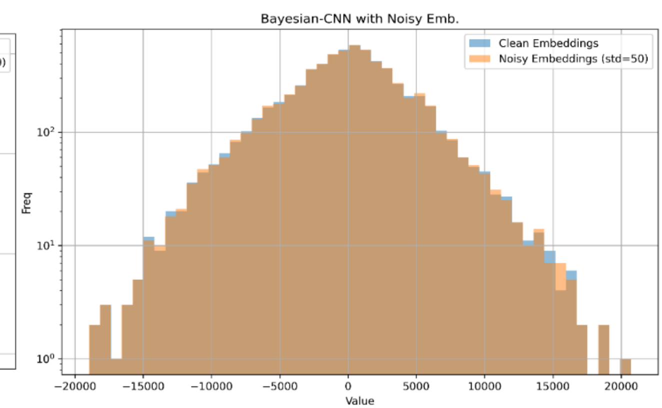
Noise-Aware Training



92 % accuracy

Noise has little impact on embedding

Bayesian NN

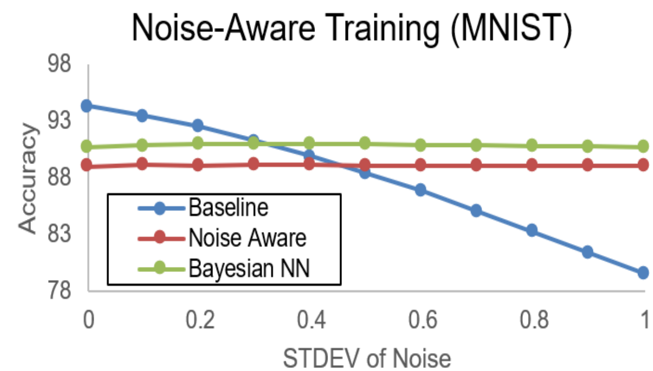


88 % accuracy

Noise has almost no impact on embedding

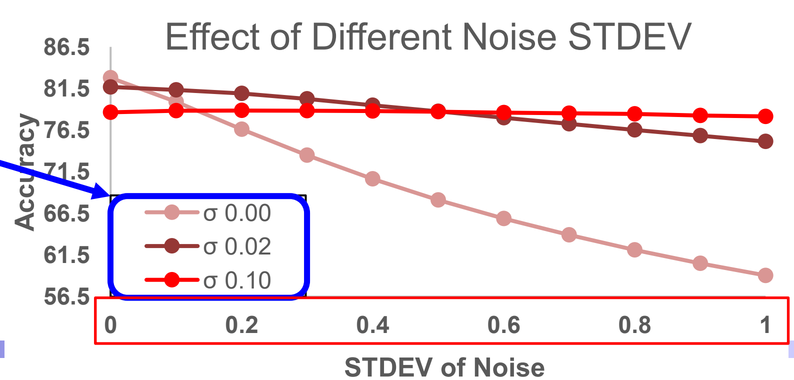
Conclusion 1

- Accuracy drop
  - Original CNN model has higher acc. on clean device, but accuracy drops on noisy device
  - Both noise-aware model and Bayesian NN resists noise by amplifying the mag. of embed.



  - Trade off between model-robustness and accuracy on ideal device
  - One small noise for training model, one giant leap for noise-tolerance

Little perturbation in training
Great effect in testing

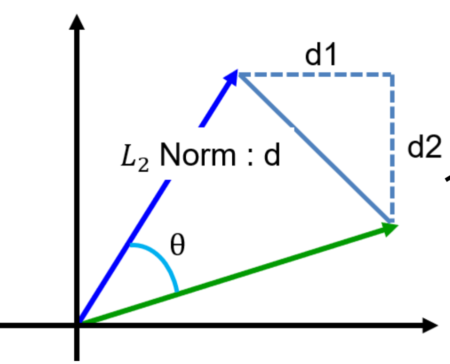


Impact of Metric Selection on Accuracy

- Cosine similarity is too complicated to implement in memory cell

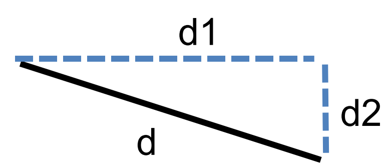
$$\mathrm{sim}(\boldsymbol{A},\boldsymbol{B}) = \frac{\boldsymbol{A} * \boldsymbol{B}}{||\boldsymbol{A}|| * ||\boldsymbol{B}||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt[2]{\sum_{i=1}^{n} A_i^2}\ \sqrt[2]{\sum_{i=1}^{n} B_i^2}}$$

- Spatial metric is used to calculate similarity in memory
  - Simple hardware, but at what cost?
  - The performance may vary slightly between different metrics.



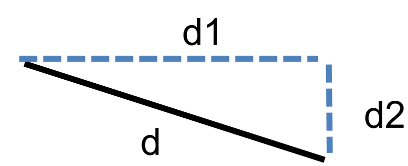
L_2 norm is non-normalized cos-sim

L_1 norm



$\|L_2\| < \|L_1\| < \sqrt{D}\|L_2\|$

L_∞ norm

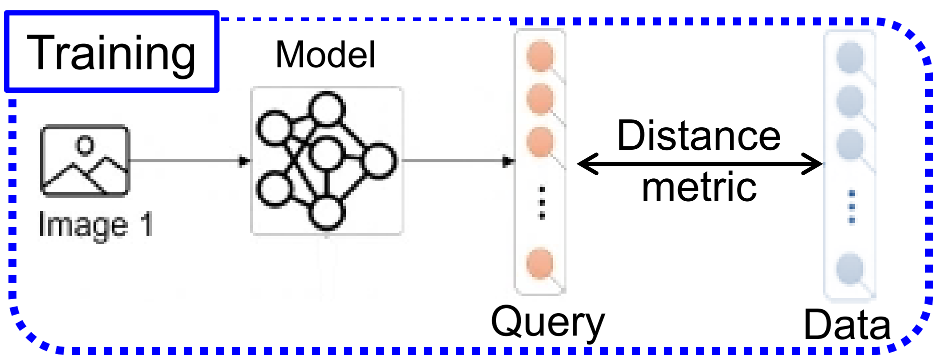


$\frac{1}{\sqrt{D}}\|L_2\| < \|L_\infty\| < \|L_2\|$

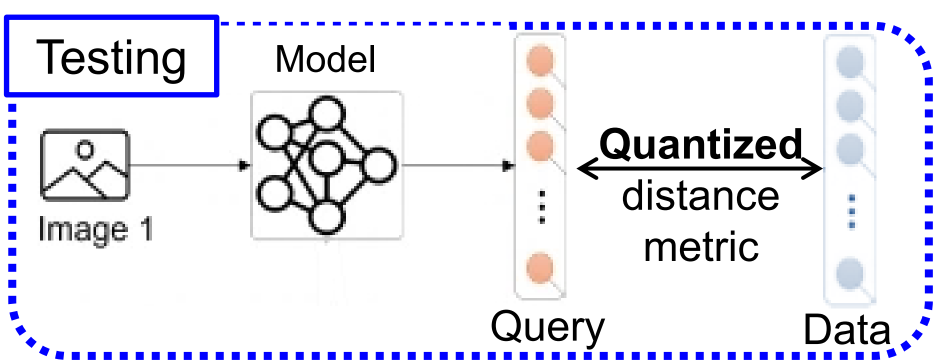
Approximation (Upper/Lower bound) of L_2 norm

Experiment Setup

- Evaluate accuracy of a model trained with different distance metric
  - Training : Approximated metric (L_∞ norm has no gradient for back prop.)
  - Testing : Regular metric on quantized embeddings



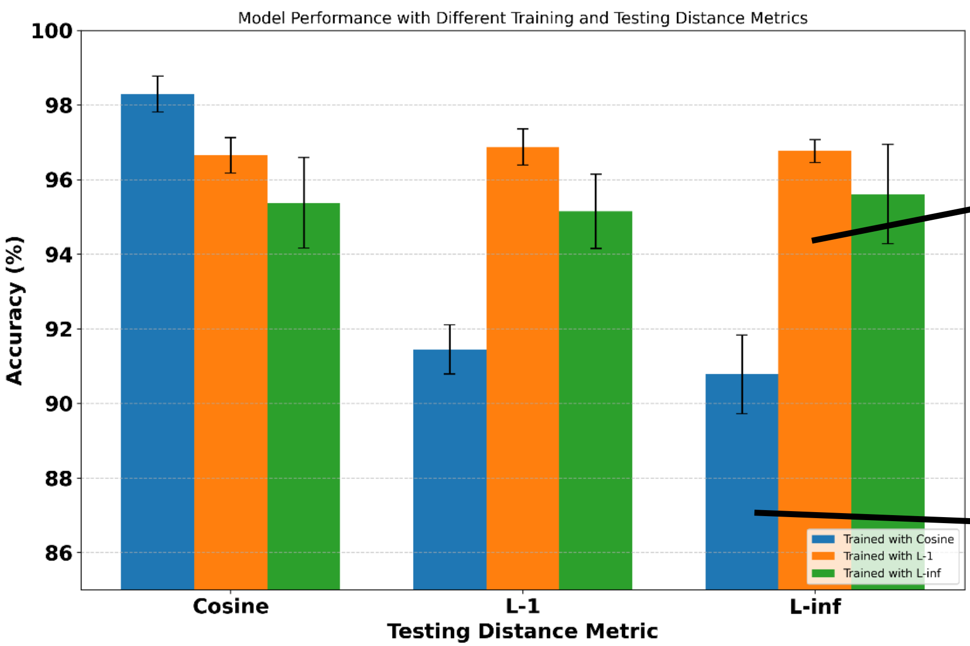
Metric (FP32) :
Cosine : regular cosine distance
L_1 : regular L_1 norm
L_∞ : $\sum(\text{emb} \odot \text{softmax}(\kappa\ \text{emb}))_i$



Metric (INT8) :
Cosine : cosine distance
L_1 : L_1 norm ($\sum \text{emb}_\text{i}$)
L_∞ : L_∞ norm ($\max(\text{emb}_\text{i})$)

Evaluate Accuracy with Different Metrics

- Compare accuracies across models trained with different metrics
  - Dataset Omniglot and MNIST is used in the experiment
    - Blue : Trained with cosine distance (angular metric)
    - Orange : Trained with L_1 norm (spatial metric)
    - Green : Trained with L_∞ norm (spatial metric)

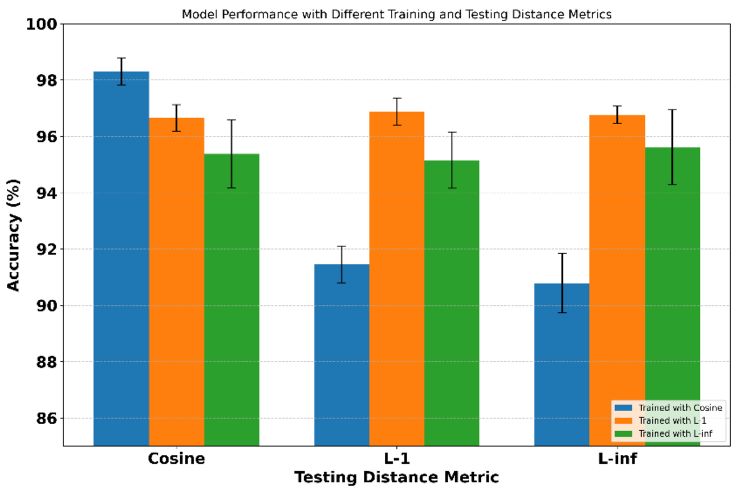


Approximate training metric of L_1 norm performs well on evaluation

Using angular metric for training results in lower accuracy under spatial metric evaluation

Observation Across Distance Metrics

- Model trained with L_1 norm outperforms model trained with L_∞ norm in evaluation under all metrics



L_2 norm : $\sqrt{\sum(\mathrm{emb_i} - q_i)^2}$

$\nabla L_2 = 2 \cdot (\mathrm{emb_i} - q_i)$

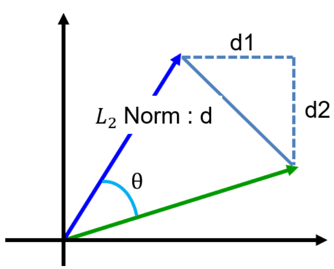
L_1 norm : $\sum|\mathrm{emb_i} - q_i|$

$\nabla L_1 = \underline{\mathrm{sign}(\mathrm{emb_i} - q_i)}$

Quantize to 1 bit

L_∞ norm : $\max(\mathrm{emb_i})$

$$\nabla L_\infty = \begin{cases} \underline{\mathrm{sign}(\mathrm{emb_i} - q_i)}, & \mathrm{emb_i} = \max(\mathrm{emb_i} - q_i) \\ \mathrm{None}, & \mathrm{else} \end{cases}$$

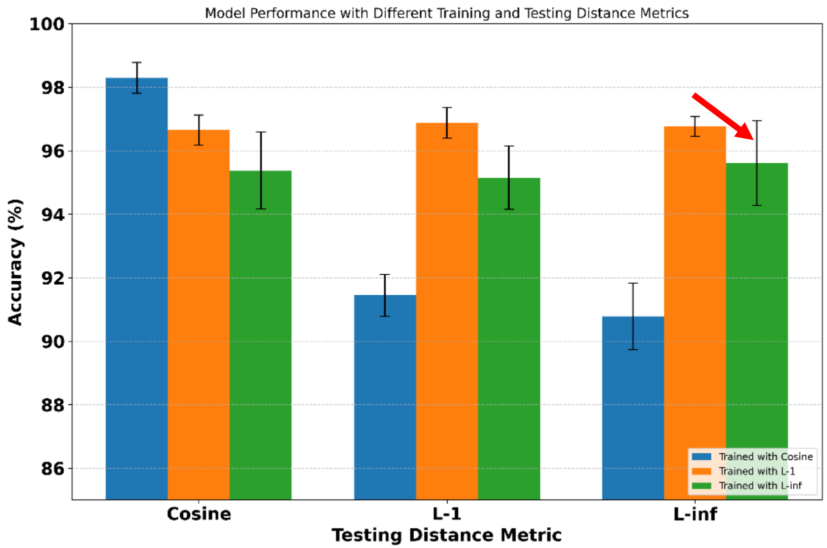


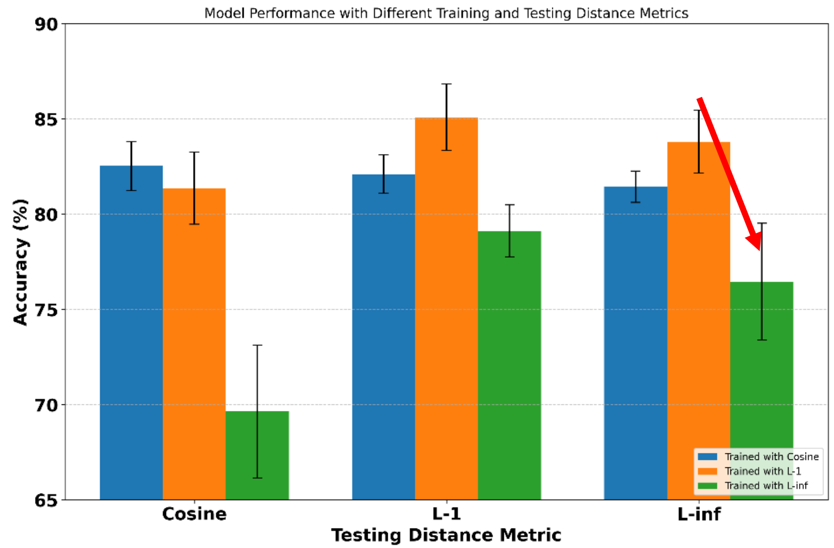
L_1 norm is a better metric?

L_1 norm can be view as a comprehensive version of L_∞ norm in training phase

Conclusion 2

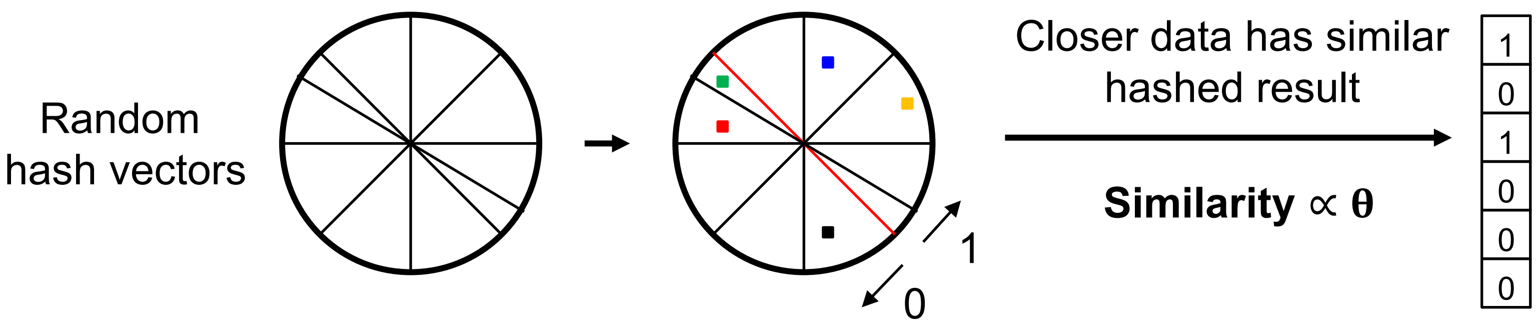
- Search function implemented in memory uses distance metric
  - Train the model with distance metric can achieve better performance
    - Use differentiable approximated distance function that allows backpropagation
  - Using L_1 norm for training may have higher accuracy than using L_∞ norm
    - Model Trained with L_1 norm may achieve higher accuracy than model trained with L_∞ norm when evaluating the performance with L_∞ norm

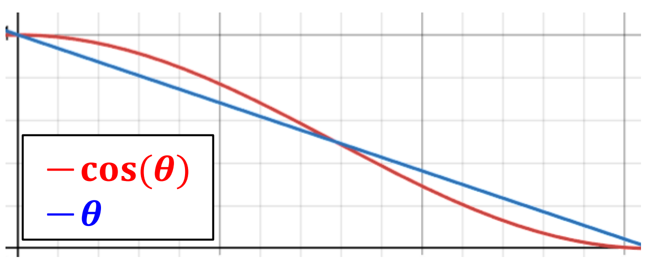




Approximation of Cosine Similarity

- Sometimes we can only use a pretrained model
  - Cannot customize distance metric used in training
- Locality-Sensitive Hashing
  - A stochastic technique for finding neighbor with highest cosine similarity
  - Similar items map to the same buckets with high probability.

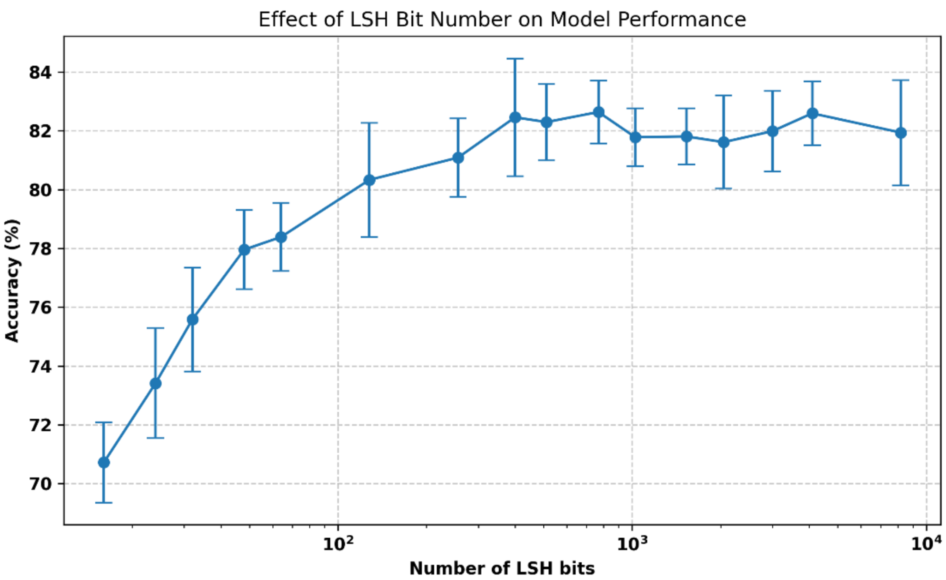


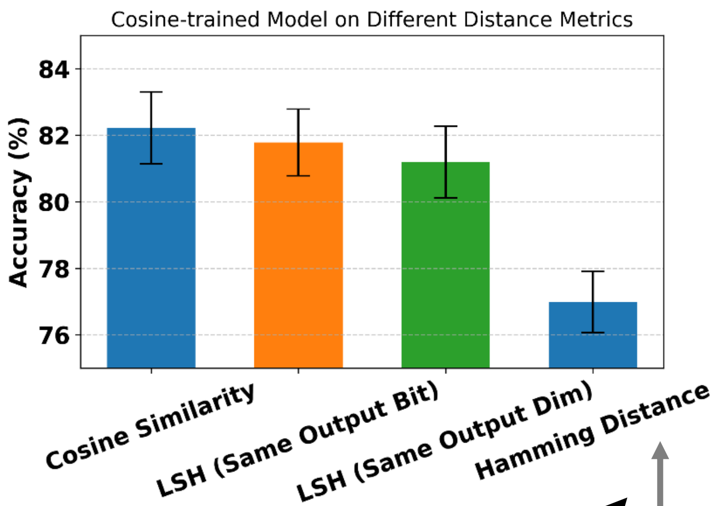


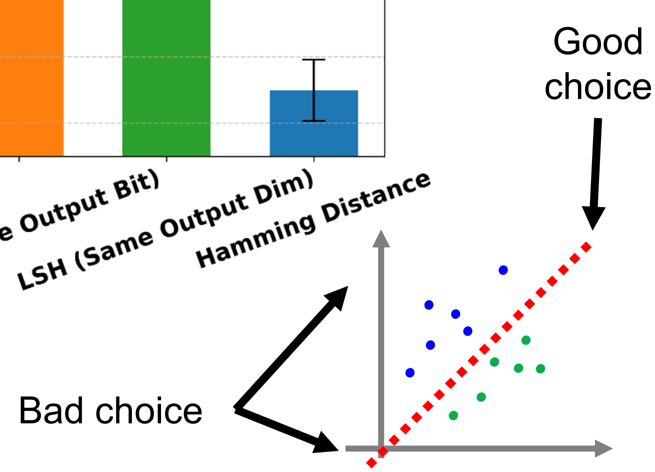
θ and $\cos(\theta)$ are monotonic functions
⇒ Should have same nearest neighbor structure

Effect of Locality-Sensitive Hashing

- Hashing vectors of LSH perform partitioning in Hilbert space.
  - Hamming distance is a special case of LSH
    (hashing vectors are normal vector of coordinate planes)
  - LSH can generally performs better than Hamming distance

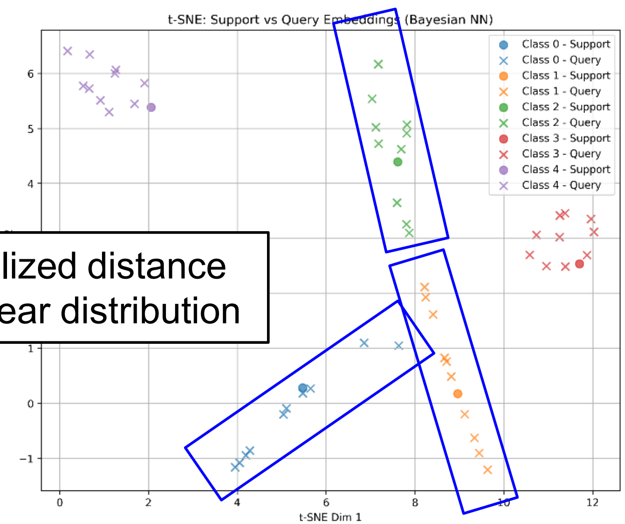




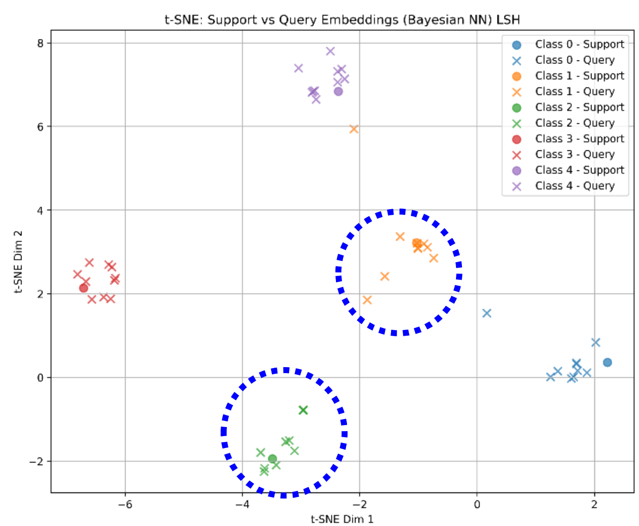


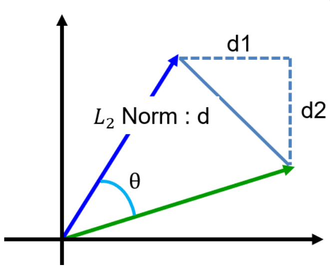
What Else Can LSH Do

- 2-D data visualization using t-SNE method
  - Visualization method that maintains distance in Hilbert space
  - Locality-Sensitive Hashing maps angular distance to spatial distance



Non-normalized distance results in linear distribution





Use embeddings generated by model trained with cosine distance directly

Do LSH on embeddings

2-D Data Visualization with LSH

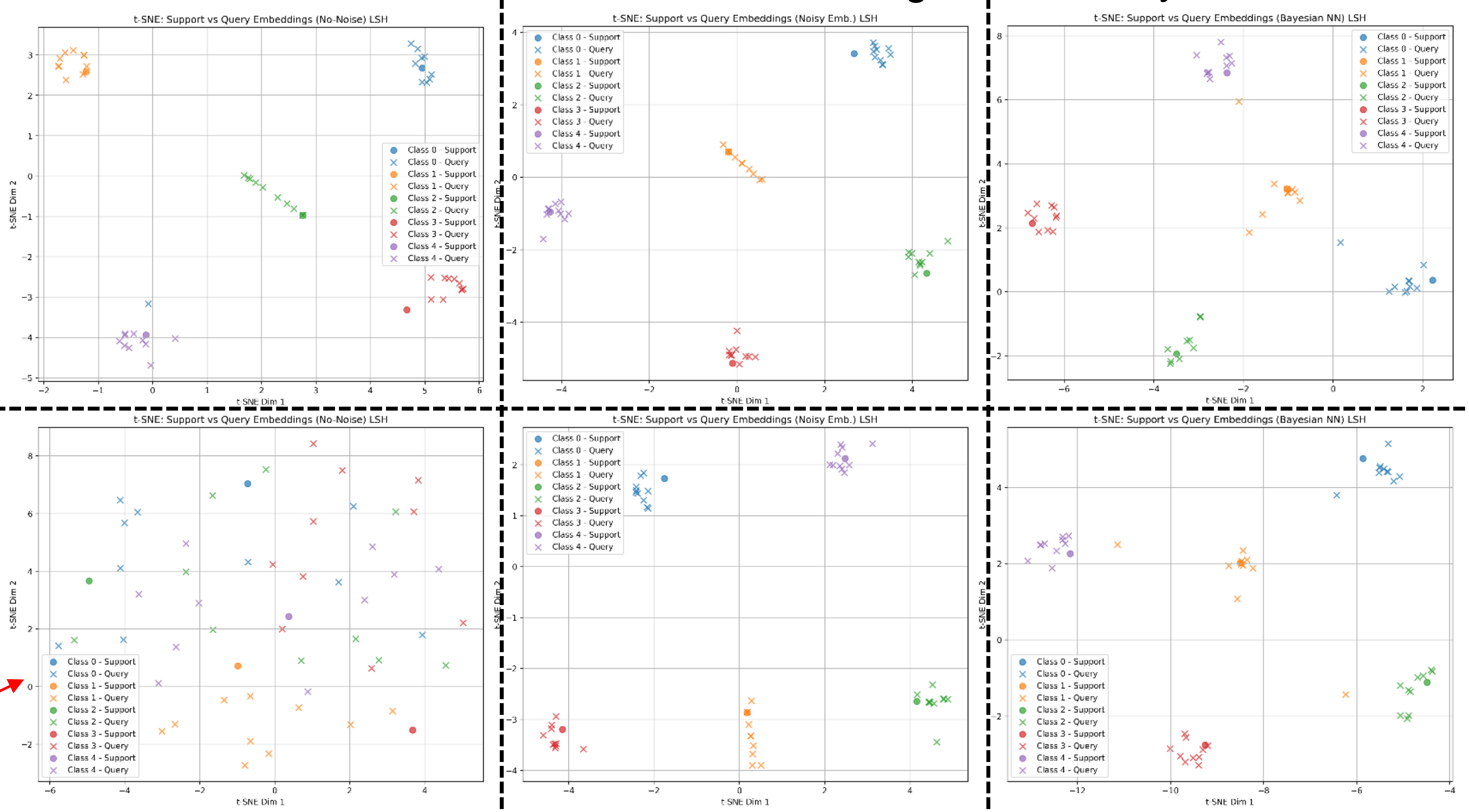
- Visualize clean and noisy data in experiment 1 using LSH & t-SNE

Baseline | Noise-Aware Training | Bayesian NN

Zero Noise

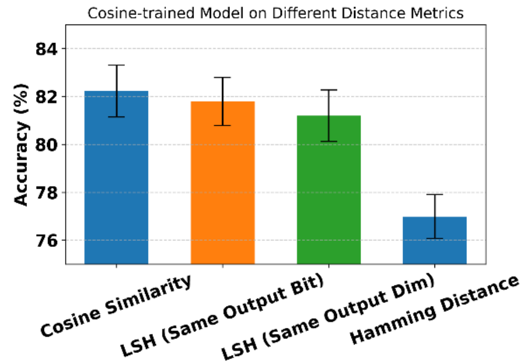
Large Noise

Low accuracy

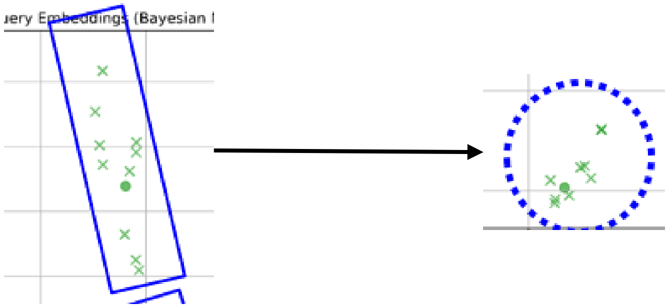


Conclusion 3

- Approximation of cosine similarity
  - Locality sensitive hashing is an alternative method if we can only get a model trained with cosine similarity which IMS does not support

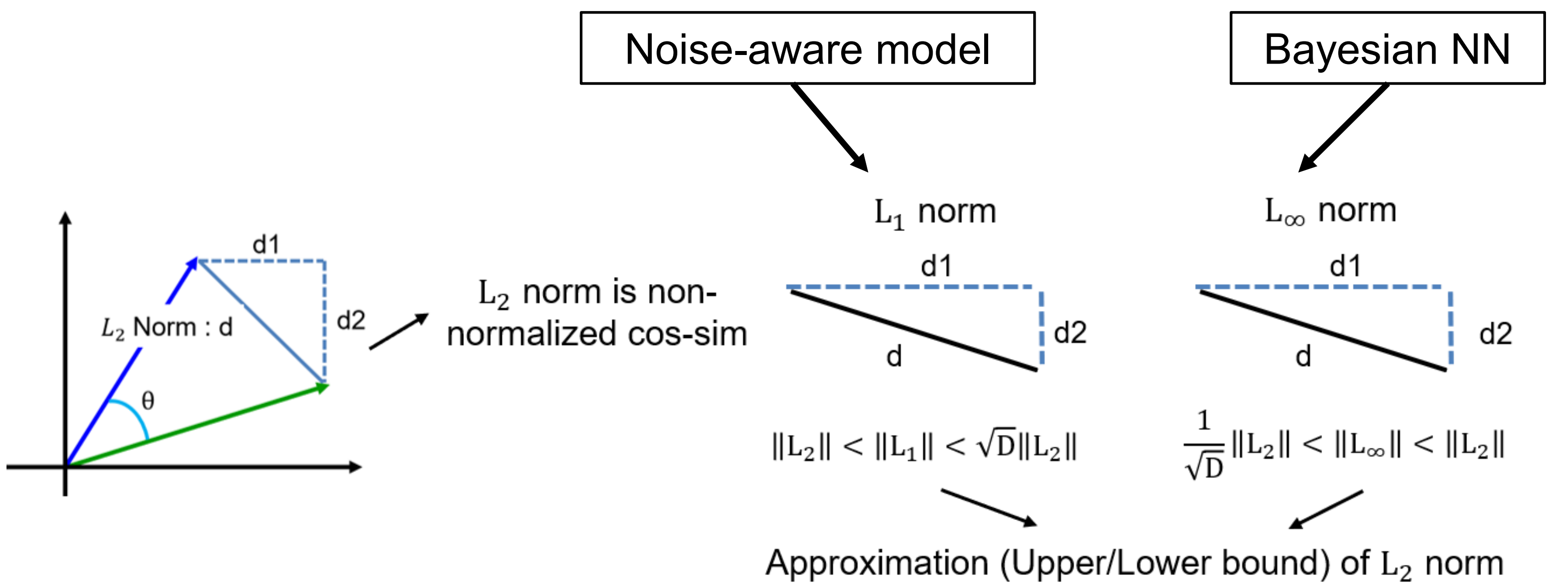


- 2-D data visualization
  - LSH converts angular metric to spatial metric which is better for t-SNE



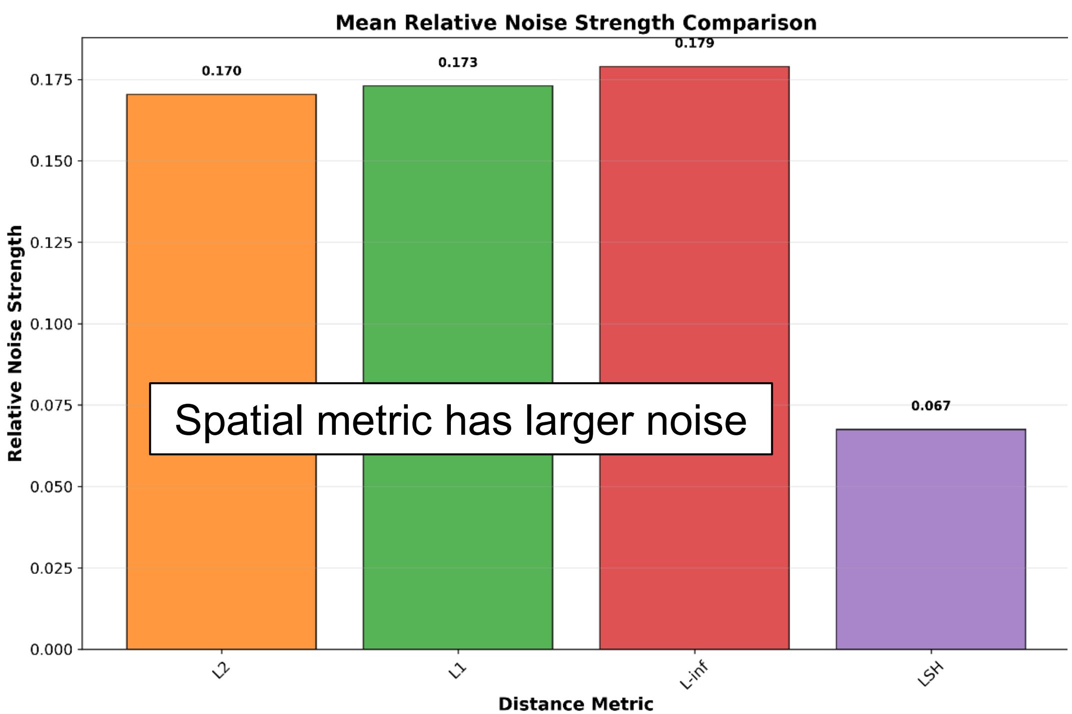
A General Way to Resolve Metric Issue

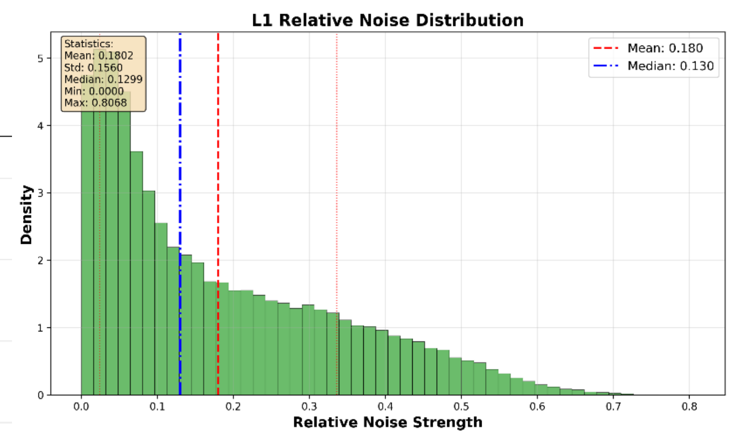
- Can we train a general model that has acceptable accuracy on each distance metric?
  - View different metric as a noisy version of cosine similarity
  - Train a noise-resilient model using cosine similarity

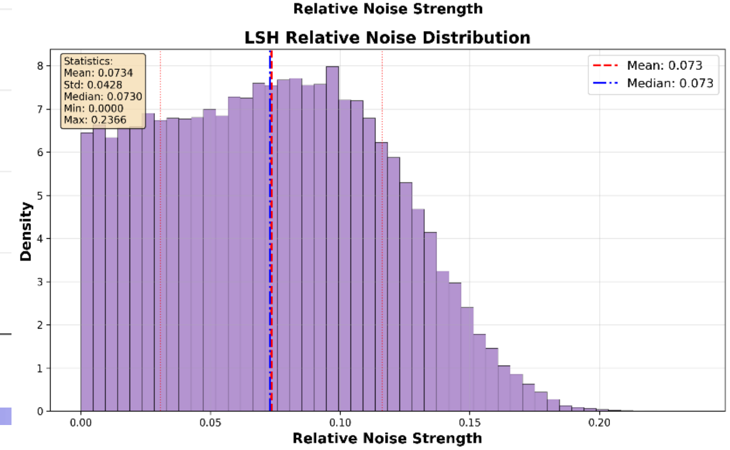


Noise Strength of Metrics

- Observation of the magnitude of each distance metric if we regard them as the source of noise
  - Clean signal : cosine similarity
  - Relative noise : LSH < L_2 < L_1 < L_∞

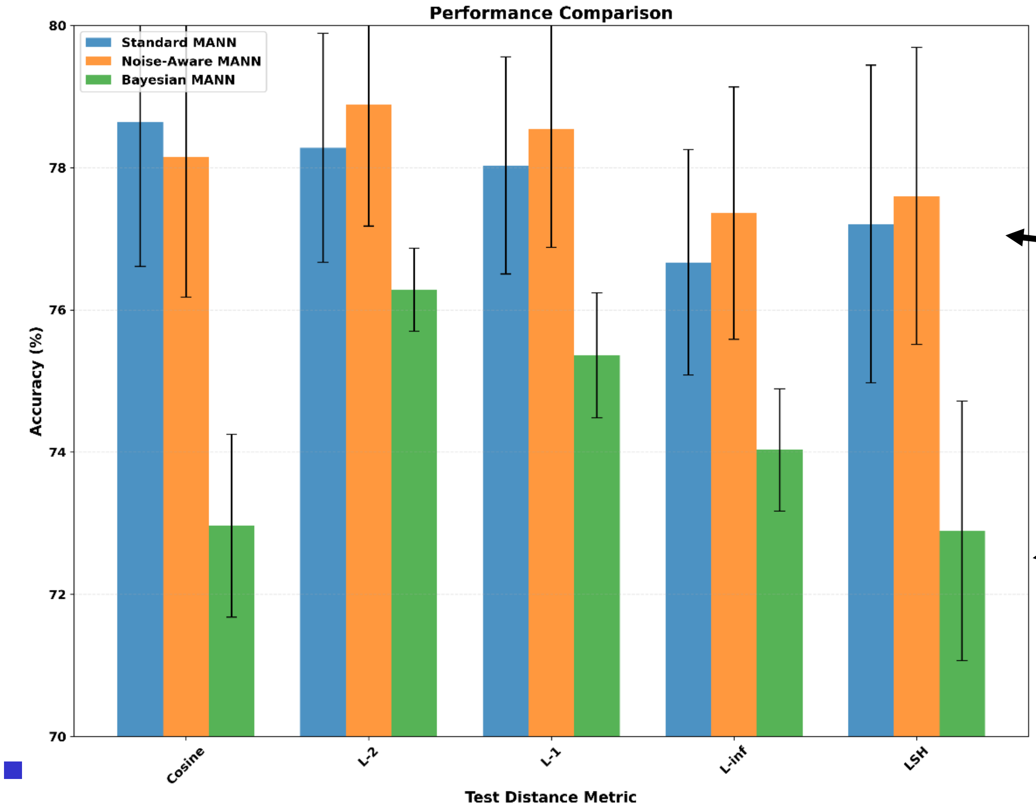






Noise-Resilient Model Resolves Metric Issue

- Noise-Resilient Model achieves higher accuracy when distance metric is not cosine distance
  - Original CNN model has highest accuracy on cosine similarity



noise-aware model has highest accuracy on other metric

Bayesian model has similar results, but has lower accuracy in general

Conclusion 4

- View different metrics as inaccurate versions of cosine similarity
  - L_2 norm : Non-normalized cosine distance
  - L_1 norm : Upper bound of L_2 norm
  - L_∞ norm : Lower bound of L_2 norm

$$\|L_\infty\| < \|L_2\| < \|L_1\|$$

- Train a model that performs well on general distance metrics
  - Noise-aware model can achieve better performance in general cases
  - Bayesian model has similar behavior

Conclusion of All Experiments