Locally Sensitive Hashing for the Content Based Image Retrieval

Masoud Badiei Khuzani, SCIT Mentor: Professor Lei Xing

January, 2021

Stanford University, Department of Radiation Oncology.

Content-based Image Retrieval

- Content-based image retrieval is the task of searching images by analyzing the contents of the image rather than the metadata such as keywords, tags, or descriptions associated with the image.

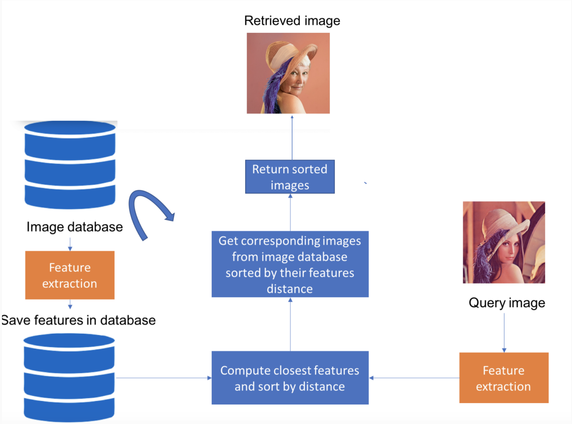


Figure 1: Content-Based Image Retrival

Content-based Image Retrieval for Medical Imaging

- Consider the image retrieval problem:

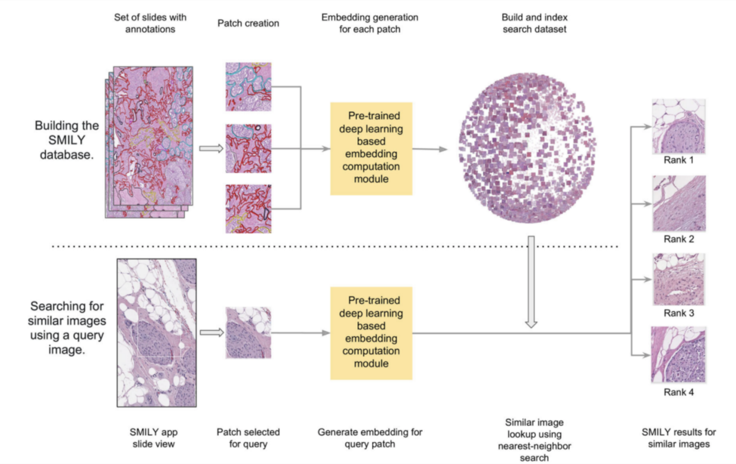


Figure 2: Image retrieval (N Hedge, et al., Nature 2019).

Other Variants of the Retrieval Systems.

- Such retrieval systems can be applied to histopathological images

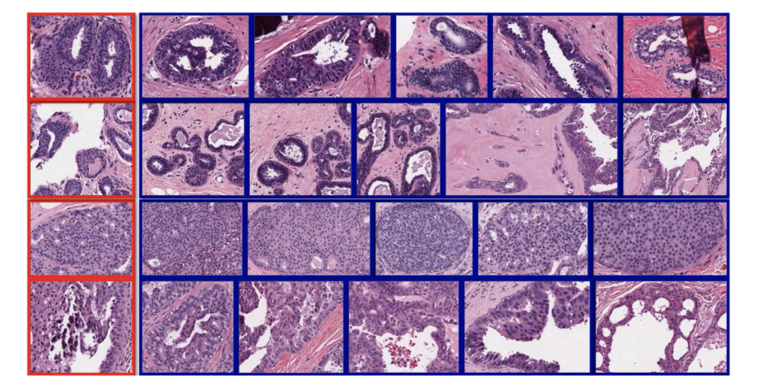


Figure 3: Image retrieval (Zhang, et al., IEEE Trans. Med 2015).

Other Variants of the Retrieval Systems.

- Such Retrieval system can also be applied to morphological neuron data-base

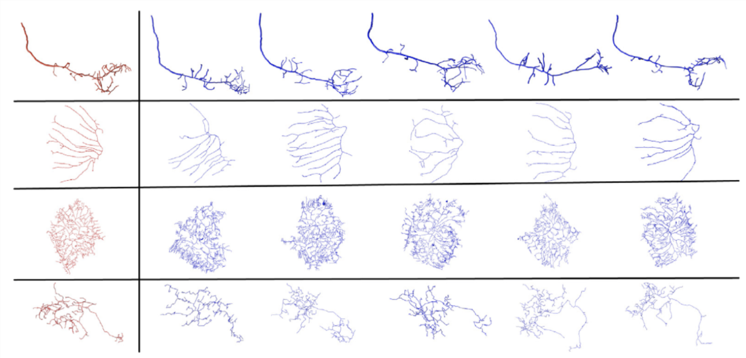


Figure 4: Image retrieval (Zhongyu Li, IEEE Trans. Pattern Recogn. 2017).

Other Variants of the Retrieval Systems.

- The crux of such retrieval system is a hash function that computes a binary code from the representations of the input

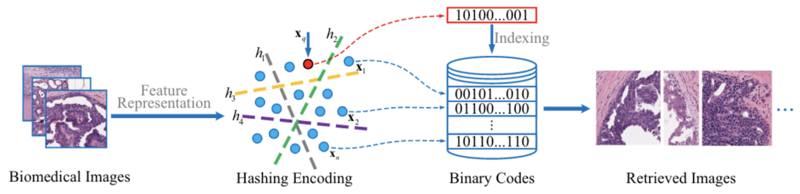


Figure 5: Image retrieval (Zhongyu Li, IEEE Trans. Pattern Recogn. 2017).

What is hashing?

- A hash function is a function takes a group of characters or numbers (called a key) and maps it into a value of a certain length.
- Hash functions and their associated hash tables are used in data storage and retrieval applications to access data in a small and nearly constant time per retrieval.
- A **collision** or **clash** occurs when two distinct pieces of data have the same hash value

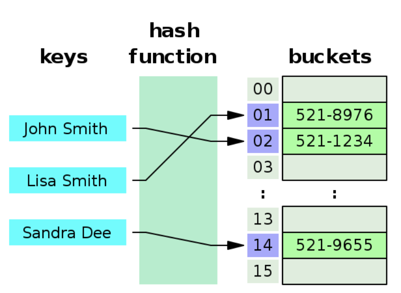


Figure 6: Hash function

What is the locally sensitive hashing (LSH)?

- A locally sensitive hash (LSH) function is an algorithmic technique that hashes similar input items into the same "buckets" with high probability.
- In a sense, we want a *controlled* collision in LSH.
- Mathematically,

$$\mathbb{P}_{h \in \mathcal{H}}[h(\boldsymbol{x}) = h(\tilde{\boldsymbol{x}})] = \mathrm{sim}(\boldsymbol{x}, \tilde{\boldsymbol{x}}). \tag{1}$$

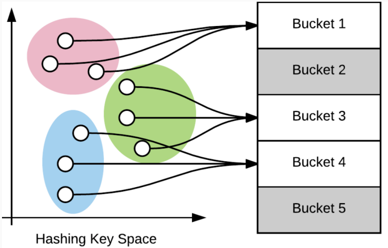


Figure 7: Locally Sensitive Hash function.

What is the use case of LSH?

- Given a query, we need to do a look up via nearest neighbor search.
- The complexity of k-NN using Euclidean distance is $\mathcal{O}(mdk)$, where m is the size of data-base and d is the dimension of data-points.

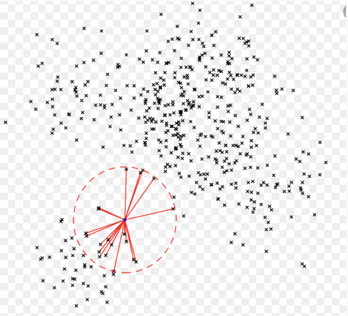


Figure 8: Image Lookup via the Nearest Neighborhood.

Application of LSH to the treatment planning

- There are objectives of treatment planning:
  1. Develop a plan that treats the tumor volume. This plan should give as homogeneous a dose distribution as possible throughout the clinical target volume.
  2. Minimize radiation dose to healthy organs. Areas outside the target volume should receive as little radiation as possible.

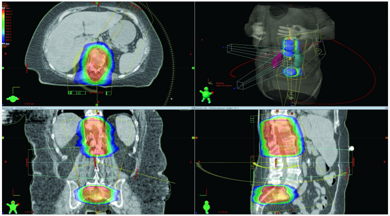
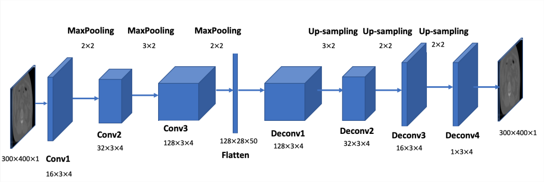


Figure 9: Volumetric Isodose distribution

Feature extraction (shallow architecture)

- We trained two convolutional auto-encoders to extract the features from CT and Isodose contours since we don't have labeled data:



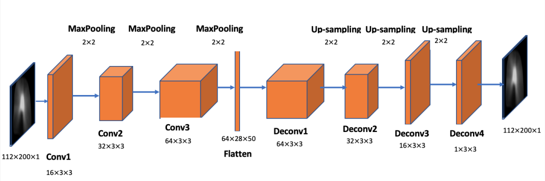


Figure 10: The structure of convolutional autoencoder for CT and Isodose contour.

Feature extraction (deep architecture)

- We recieved new data which enabled us to design a deeper convolutional autoencoder:

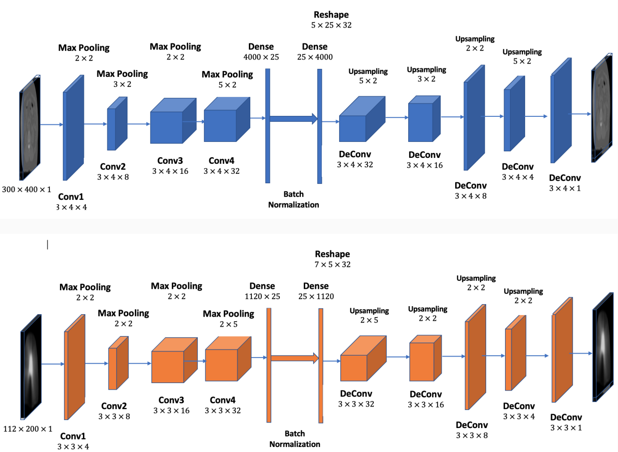


Figure 11: The structure of convolutional autoencoder for CT and Isodose contour.

An example of LSH

- We the use a hash map to map the extracted features $h : \mathbb{R}^d \mapsto \{0, 1, 2, \cdots, q-1\}$.
- A possible hash map for $q = 2$ (Hamming code) is proposed by Raginsky, et al. [NIPS 2009], where

$$h_{t,b,\boldsymbol{\omega}}(\boldsymbol{x}) \stackrel{\text{def}}{=} \frac{1}{2}\Big[1 + \mathrm{sign}(\cos(\langle \boldsymbol{\omega}, \boldsymbol{x} \rangle + b) + t)\Big], \tag{2}$$

  where

  1. $t \sim \mathrm{Uniform}[-1, 1]$.
  2. $b \sim \mathrm{Uniofmr}[-\pi, \pi]$.
  3. $\boldsymbol{\omega} \sim \mathrm{N}(0, \boldsymbol{I}_{d\times d})$.

- A hash code of length n can be constructed by sampling these random variables i.i.d., i.e., $h^n(\boldsymbol{x}) = (h_{t_1,b_1,\boldsymbol{\omega}_1}(\boldsymbol{x}), \cdots, h_{t_n,b_n,\boldsymbol{\omega}_n}(\boldsymbol{x}))$, where $t_1, \cdots, t_n \sim \mathrm{Uniform}[-1, 1]$, $b_1, \cdots, b_n \sim \mathrm{Uniofmr}[-\pi, \pi]$, and $\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_n \sim \mathrm{N}(0, \boldsymbol{I}_{d\times d})$.

Relationship between Hamming distance and Euclidean distance in LSH

- The hamming distance remains flat as the Euclidean distance change.

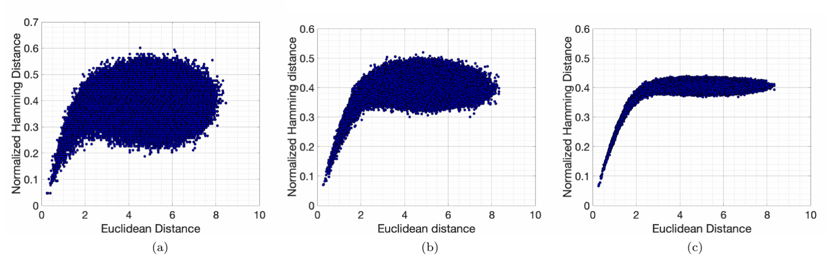


Figure 12: The scatter plots of normalized Hamming distance versus Euclidean distance for the hash function, where the hash functions. Panel (b): 512 bits, Panel (c): 4096 bits

The input/output from a shallow convolutional auto-encoders

- Reconstruction of CT slices with the shallow convolutional auto-encoders:

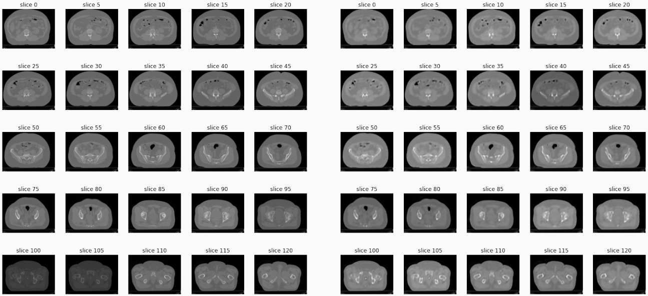


Figure 13: Left:input, Right:output

The input and reconstructions with the shallow convolutional auto-encoders

- Reconstruction of Isodose curves with the shallow convolutional auto-encoders architecture:

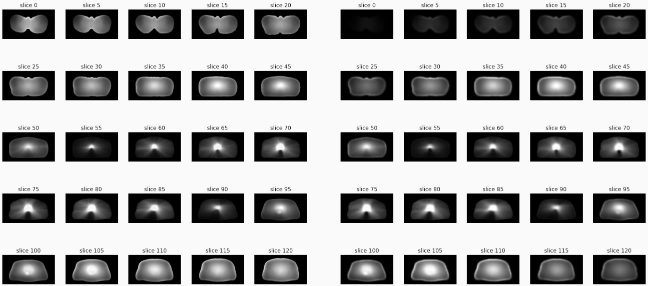


Figure 14: Left:input, Right:output

3D view of our reconstruction with the shallow convolutional auto-encoders

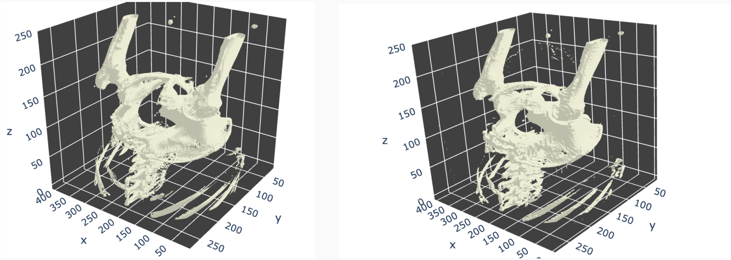


Figure 15: Left:input, Right:output

The input/output from a deep convolutional auto-encoders

- Reconstruction of CT slices with the new architecture:

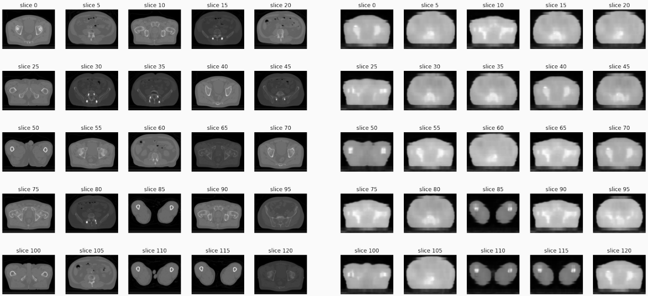


Figure 16: Left:input, Right:output

The input/output from the deep convolutional auto-encoders

- Reconstruction of RT dose slices with the deep convolutional auto-encoders:

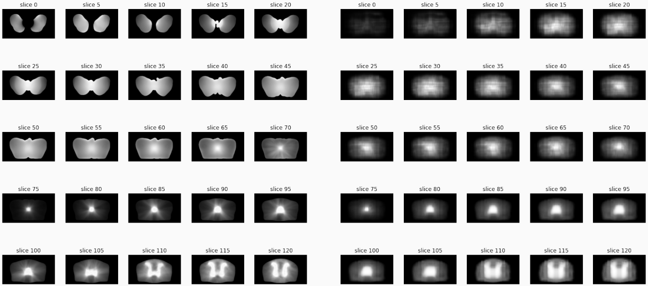


Figure 17: Left:input, Right:output

Retrieval Results for CT slices only-Shallow network

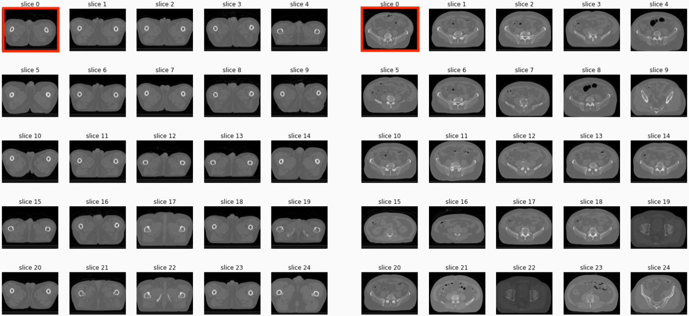


Figure 18: Examples of Retrieval for CT slices without including their isodose contours.

Joint Retrieval Results for the CT and Isodose Contours-Shallow network

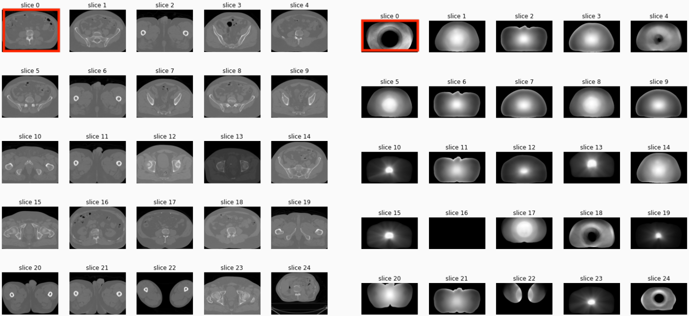


Figure 19: Examples of Retrieval for CT slices without including their isodose contours.

Retrieval Results for CT slices only-Deep network

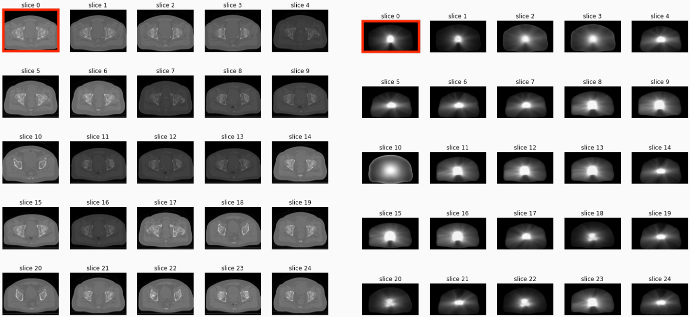


Figure 20: Examples of Retrieval for CT slices without including their isodose contours.

Retrieval Results for CT slices only-Deep network

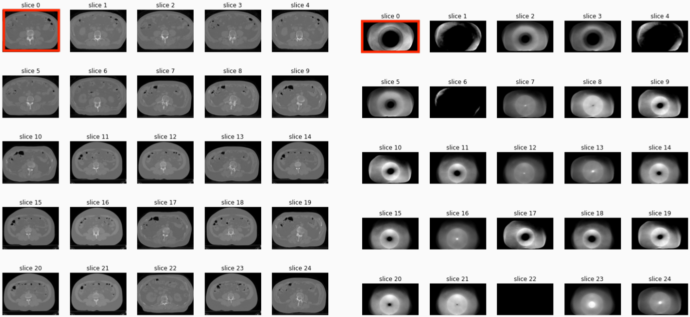


Figure 21: Examples of Retrieval for CT slices without including their isodose contours.

Retrieval Results for CT slices only-Deep network

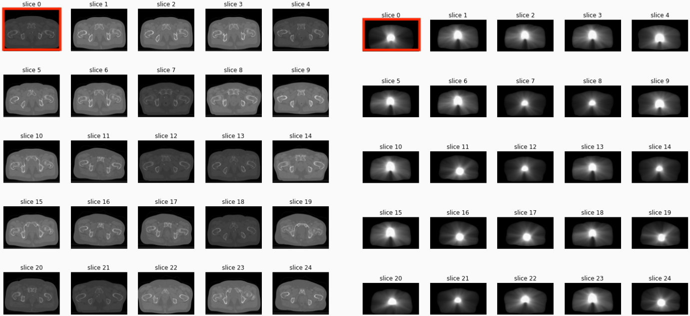


Figure 22: Examples of Retrieval for CT slices without including their isodose contours.

Retrieval Results for CT slices only-Deep network

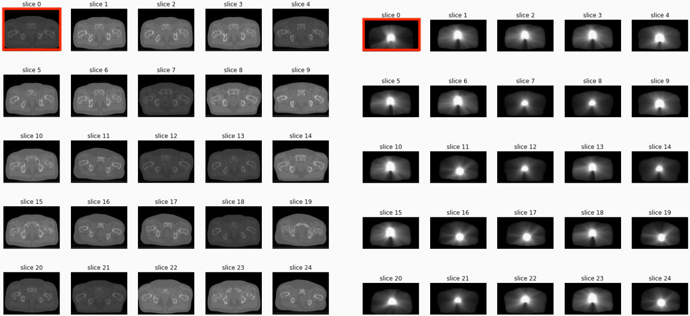


Figure 23: Examples of Retrieval for CT slices without including their isodose contours.

Performance metrics

- The Precision-Recall and ROC curves cannot be measured for unsupervised application.
- However, we observe that a hash function is the Hamming embedding of the Euclidean vectors.
- An ideal hash function should compute a Hamming distance comparable to the Euclidean distance between two input vectors.

Performance metrics

- We define the Sørensen–Dice score as follows

$$\mathcal{D}(\mathcal{I}_{\text{Euclidean}}, \mathcal{I}_{\text{Hamming}}) = \frac{2|\mathcal{I}_{\text{Euclidean}} \cap \mathcal{I}_{\text{Hamming}}|}{|\mathcal{I}_{\text{Euclidean}}| + |\mathcal{I}_{\text{Hamming}}|},$$

  where $\mathcal{I}_{\text{Euclidean}}$ is the set of retrieved images based on the Euclidean distance, and $\mathcal{I}_{\text{Hash}}$ is the set of images based on the Hamming distance.
- The Sørensen–Dice score is closely related to the Jaccard index

$$\mathcal{J}(\mathcal{I}_{\text{Euclidean}}, \mathcal{I}_{\text{Hamming}}) = \frac{|\mathcal{I}_{\text{Euclidean}} \cap \mathcal{I}_{\text{Hamming}}|}{|\mathcal{I}_{\text{Euclidean}} \cup \mathcal{I}_{\text{Hamming}}|},$$

  which is a statistic used in understanding the similarities between sample sets.

Violin plots for the number of bits

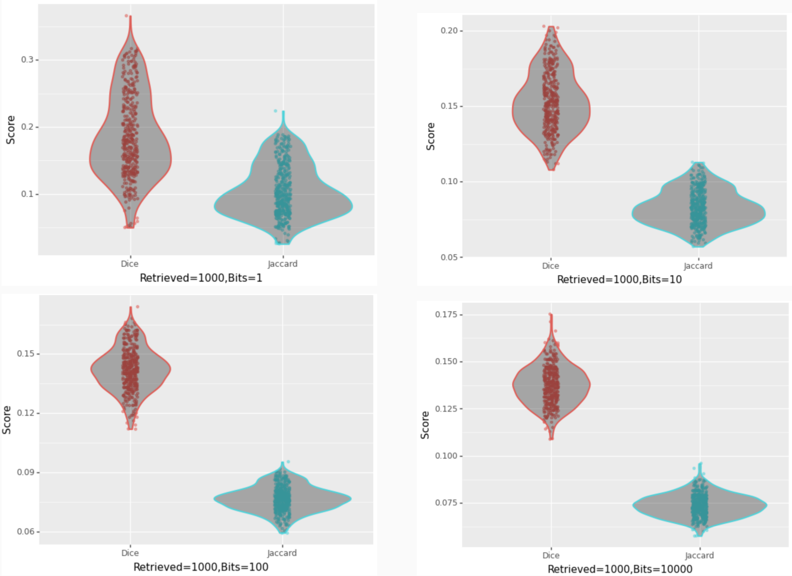
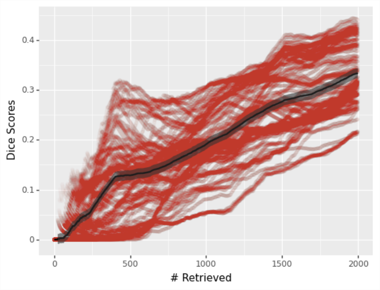


Figure 24: The violin diagram of the Jaccard and Dice scores for 400 query images

Scores for different retrieved images



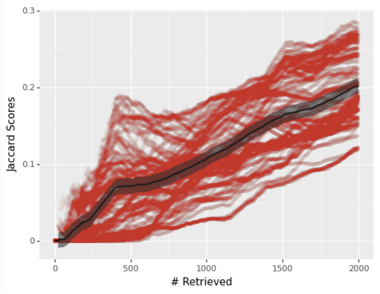


Figure 25: The Jaccard and Dice scores of 100 query slices for different number of retrieved images.

Scores for different retrieved images

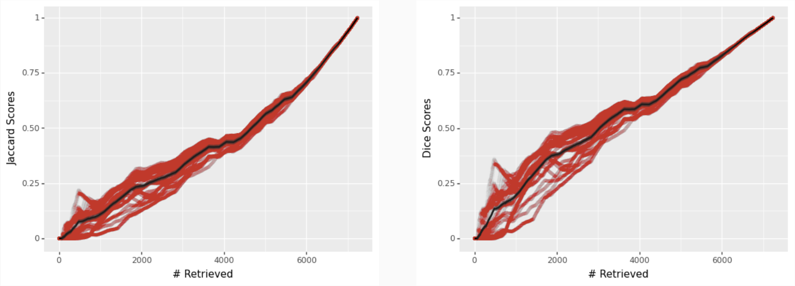


Figure 26: The Jaccard and Dice scores of 100 query slices for different number of retrieved images.

Conclusion

- We developed a locally sensitive hashing for the CT image retrieval with the application to the treatment planning.
- Our LSH is based on a kernel function that maps the extracted feature vectors into a binary (Hamming) code.
- Currently, we intend to improve our hashing method by employing recent kernel training techniques that we have developed in our previous papers.