i-Vector Transformation using k-Nearest Neighbors for Speaker Verification

Umair Khan, Miquel India and Javier Hernando

TALP Research Center,
Department of Signal Theory and Communications,
Universitat Politècnica de Catalunya, Barcelona, Spain

ICASSP-2020

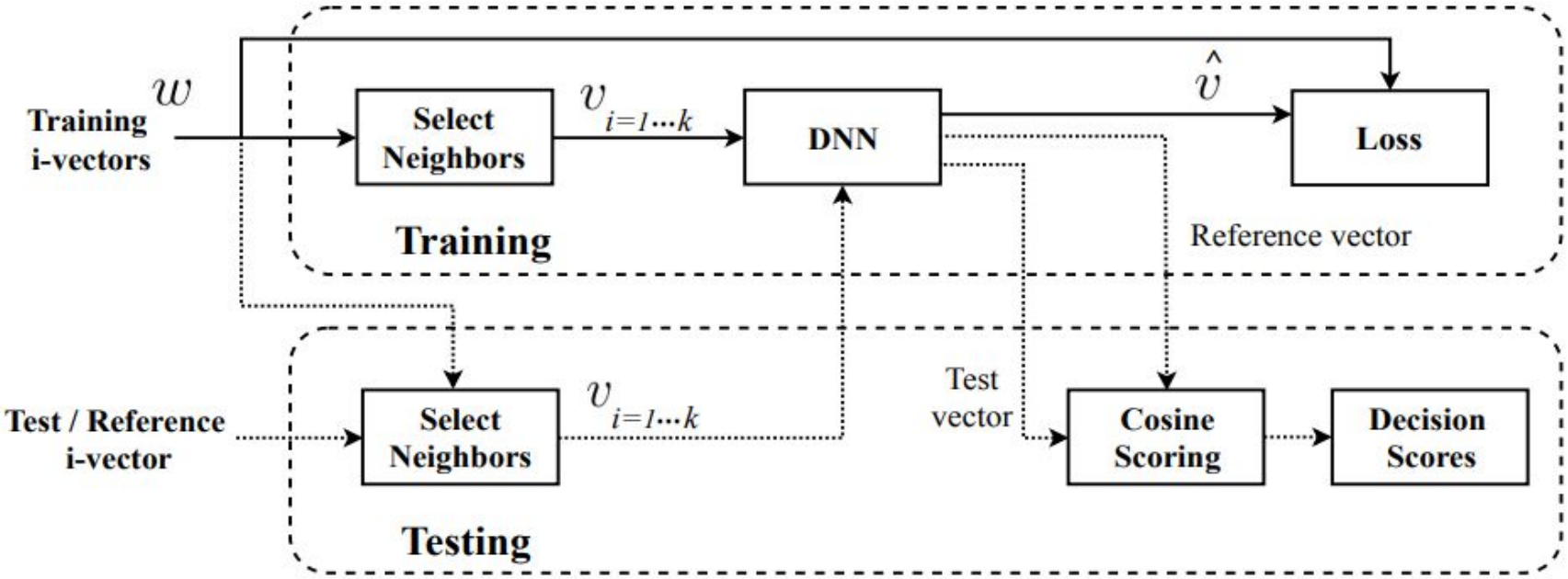
Outline

- Introduction and Motivation
- Proposed System
  - Selection of Nearest Neighbor i-vectors
  - DNN Training
  - Speaker Vector Extraction
- Experimental Setup and Database
- Results
- Conclusion

Motivation

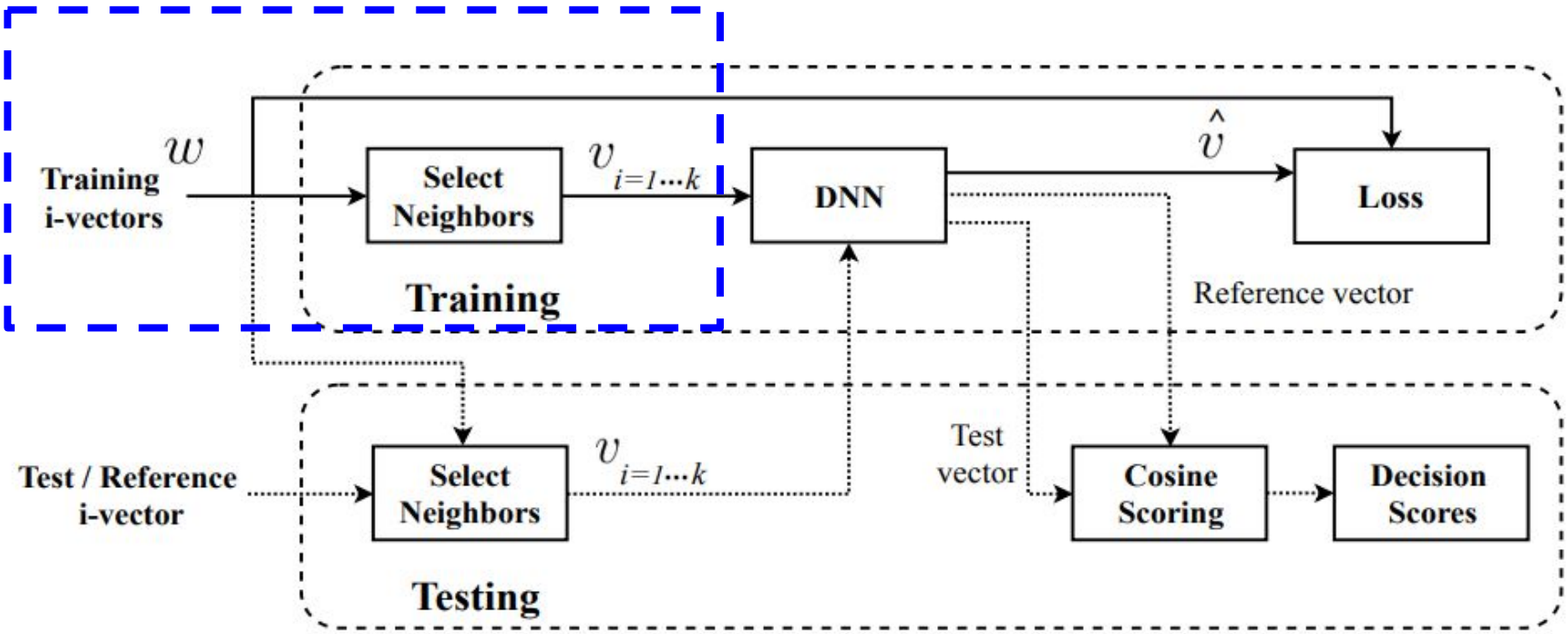
- i-vector is the unsupervised state of the art in speaker recognition.
- PLDA is the most efficient backend but requires speaker labels.
- Cosine avoids speaker labels but degrades performance.
- Propose unsupervised backend of i-vectors to avoid speaker labels and increase their discriminative power.
- We transform i-vectors into a new speaker vectors, using a DNN trained with Nearest Neighbor i-vectors.

Proposed System



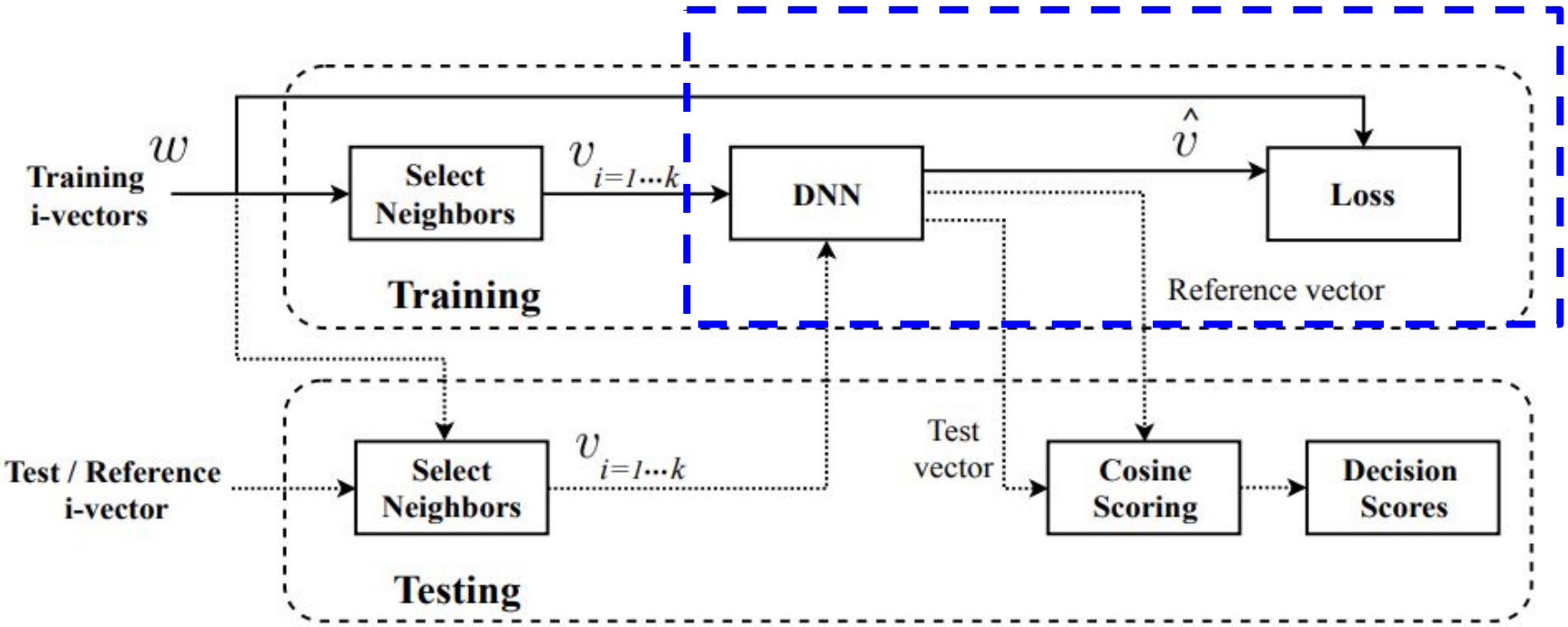
- There are three main Stages

Proposed System



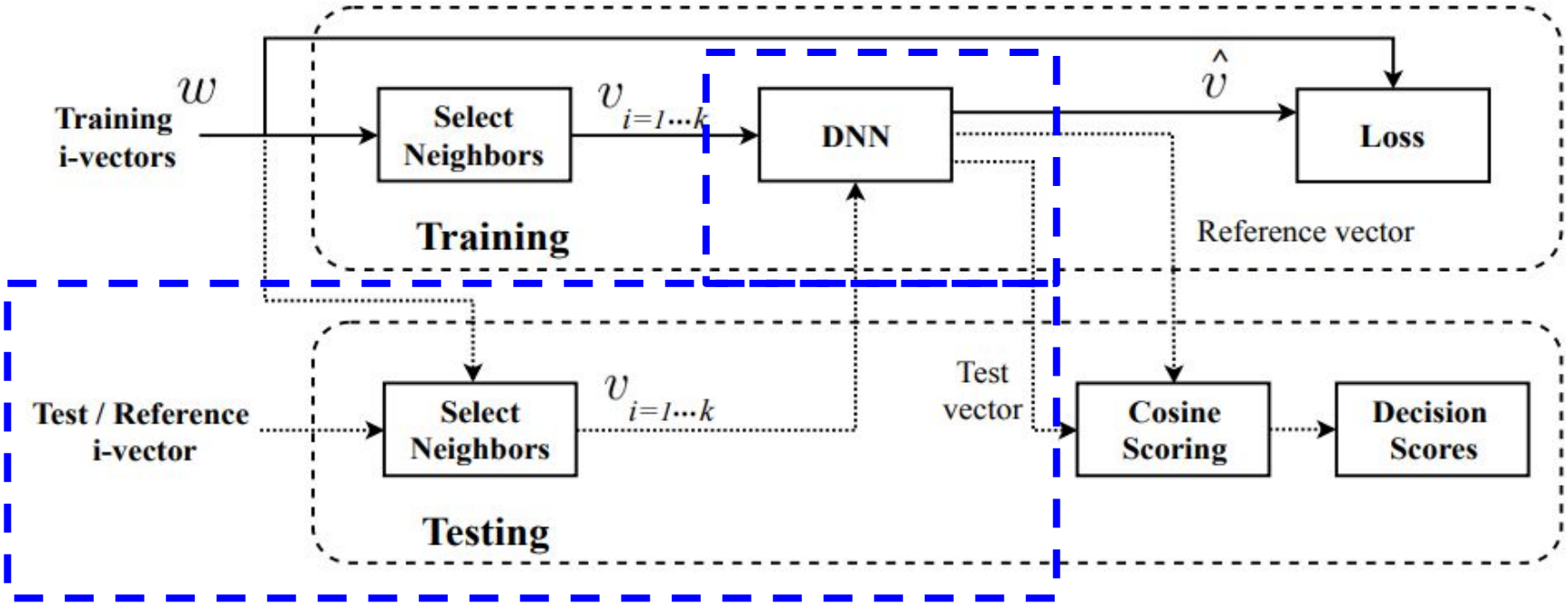
- Stage 1
  - Selection of Nearest Neighbor i-vectors

Proposed System



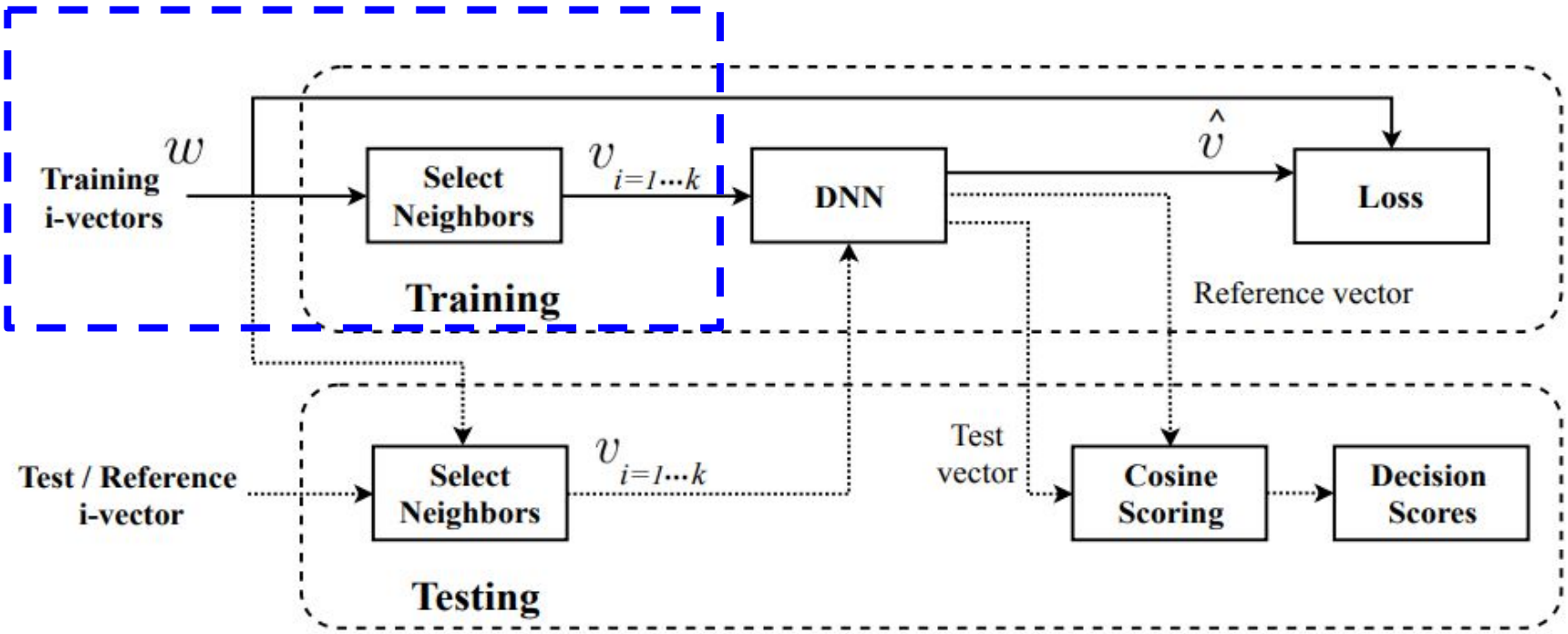
- Stage 2
  - DNN Training

Proposed System



- Stage 3
  - Speaker Vector Extraction

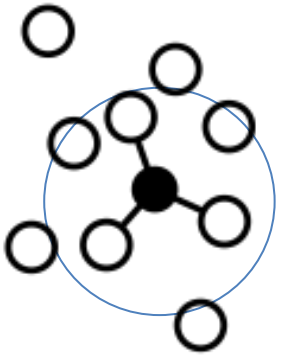
Selection of Nearest Neighbor i-vectors



- Stage 1
  - Selection of Nearest Neighbor i-vectors

Selection of Nearest Neighbor i-vectors

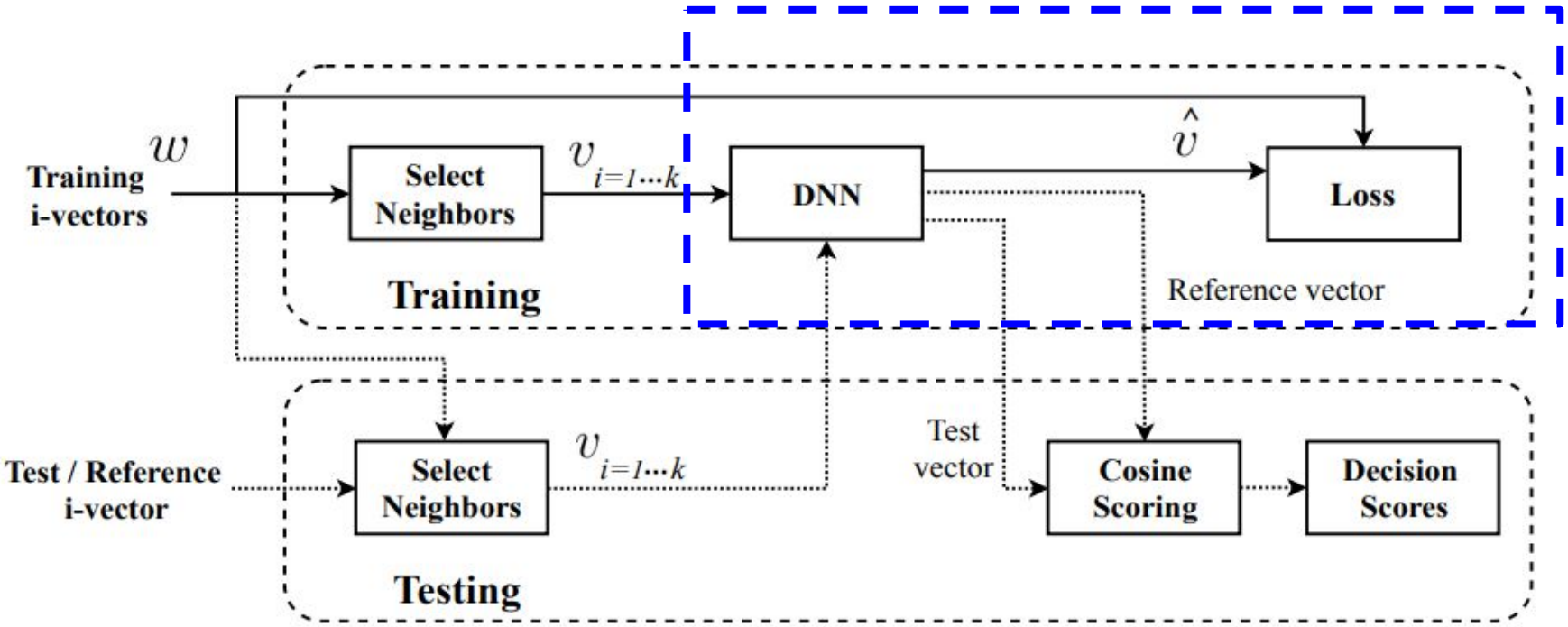
- Unsupervised manner
- For every i-vector in background data, select k-nearest neighbor i-vectors, that are:
  - Closest according to cosine score
  - Pass a certain threshold



● Training i-vector

○ Potential Neighbor i-vectors

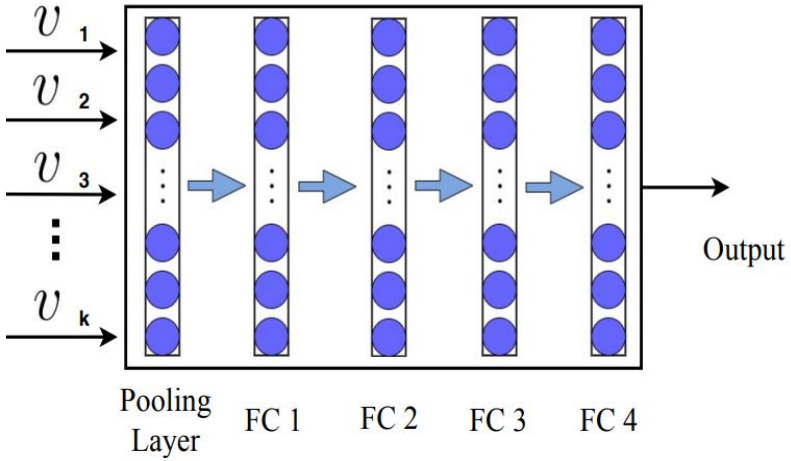
DNN Training



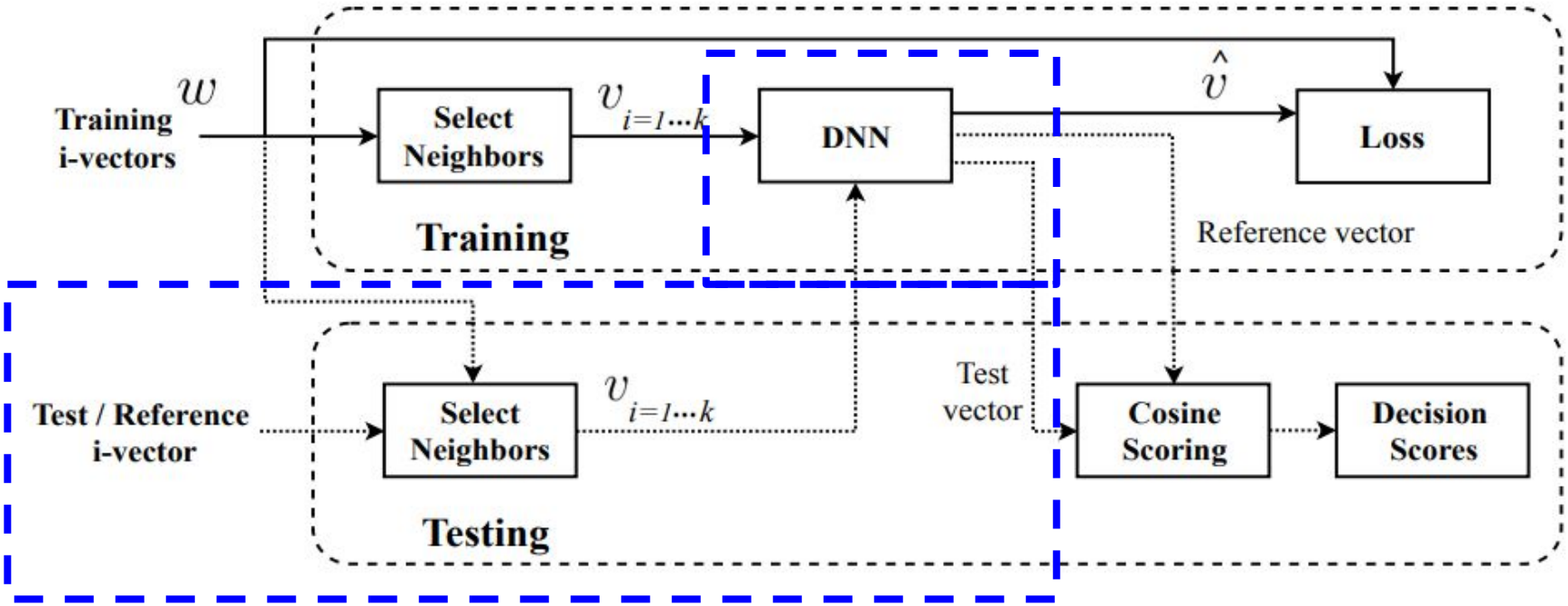
- Stage 2
  - DNN Training

DNN Training

- Inputs are the k-Nearest Neighbors
- First layer performs Average Pooling operation
- FC1-FC3 perform ReLU operation, while FC4 performs linear function.
- Minimize the loss function $L(\hat{v}, w)$
- $L(\cdot)$ can be Cosine Distance (CD) or Mean Squared Error (MSE)



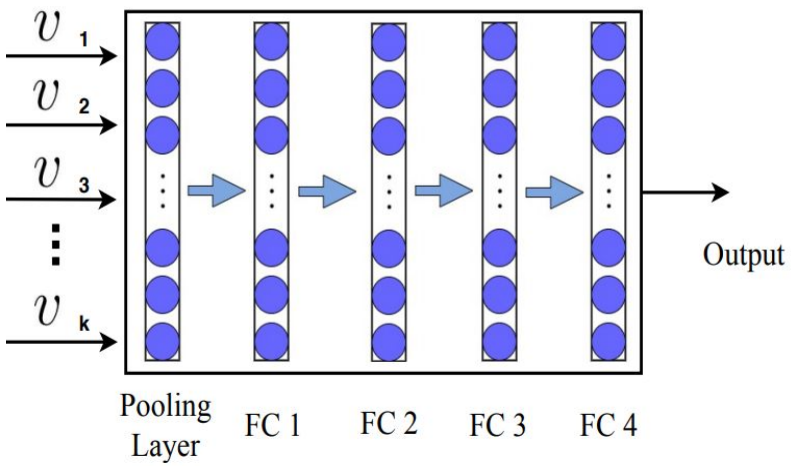
Speaker Vector Extraction



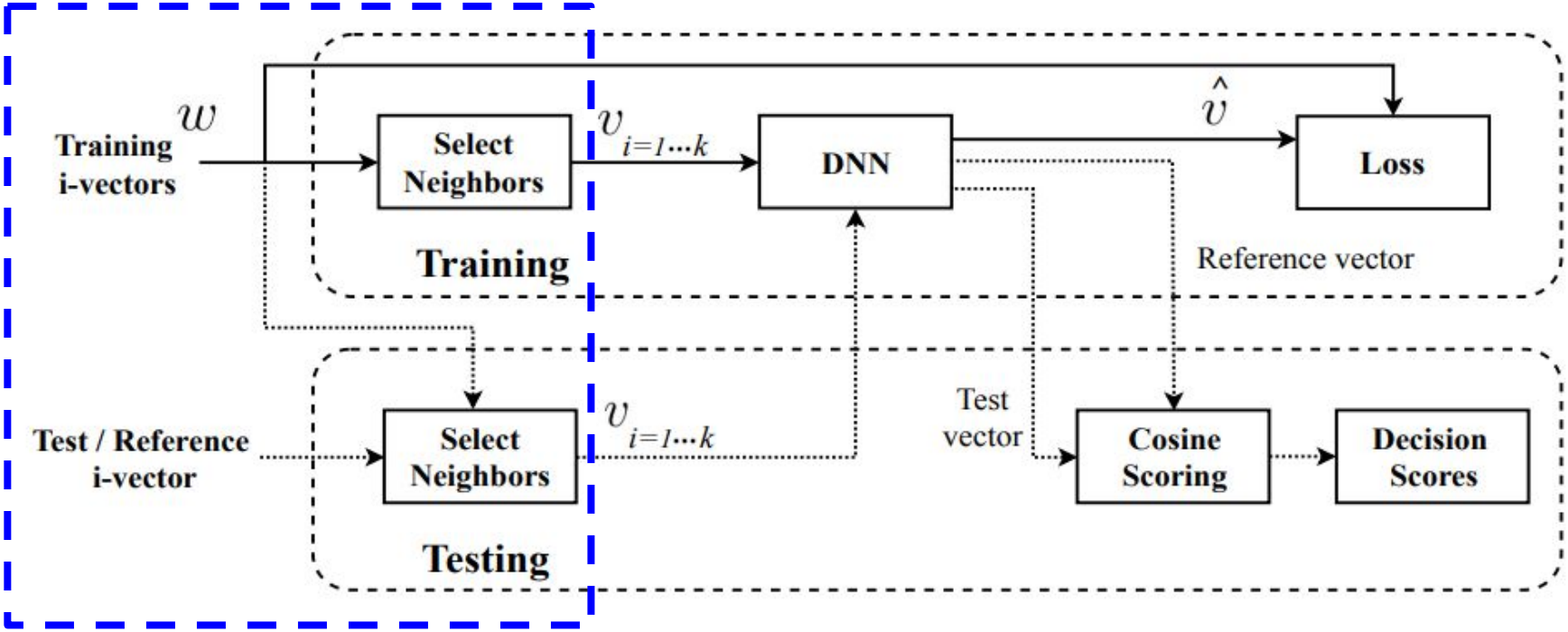
- Stage 3
  - Speaker Vector Extraction

Speaker Vector Extraction

- Select k-Nearest Neighbors for every test i-vector
- Input to the already trained DNN
- Extract Speaker vector at the output of the DNN
- Score experimental trials using cosine scoring technique



Proposed System



➢ **Background data is required in the testing phase**

Experimental Setup and Database

- Experiments on VoxCeleb-1 database.
- Train partition: 148642 utterances in total
- Test partition: 4874 utterances in total
- Nearest Neighbor and DNN training on Train partition (1211 speakers).
- Evaluation on Test partition (40 speakers, 37720 trials).
- UBM, T-matrix, and PLDA trained on VoxCeleb-1, Train partition.
- 20 MFCC + Deltas and 1024 components UBM were used to extract i-vectors.

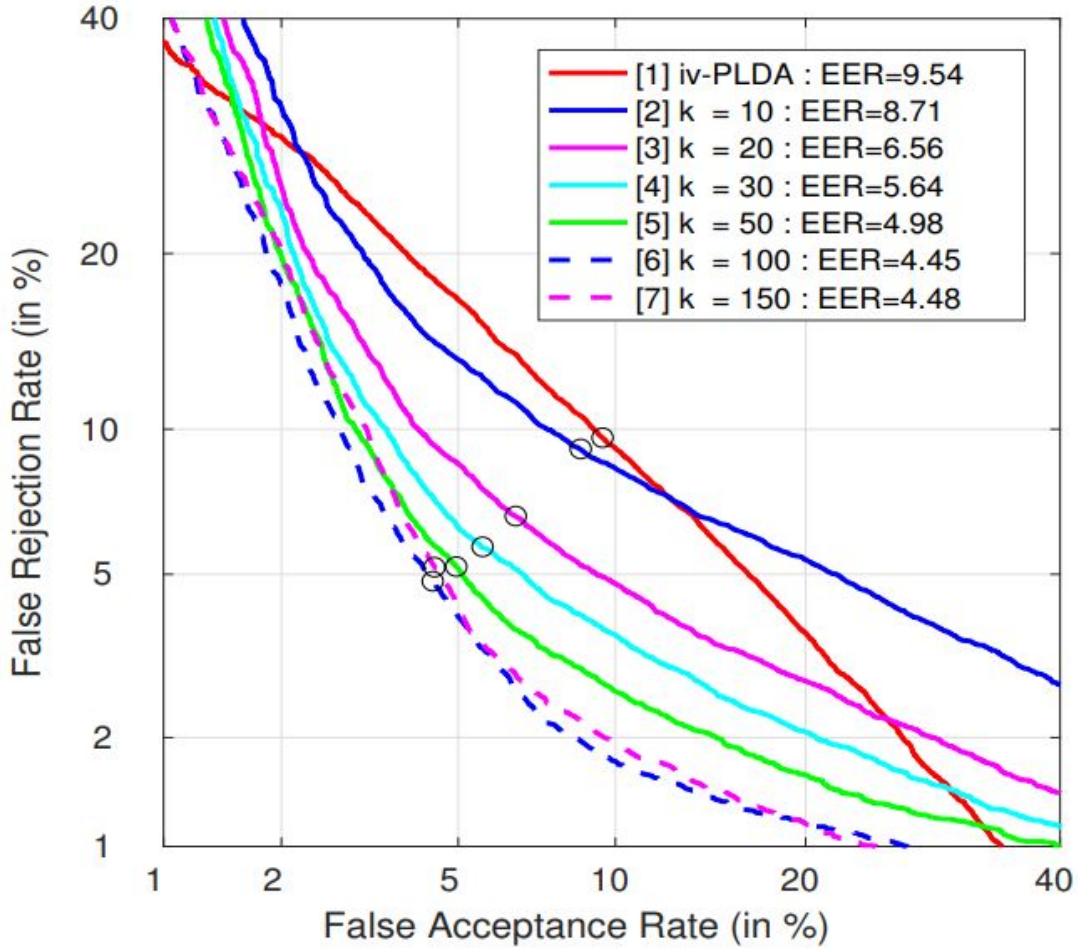
Results

EER(%) for the proposed vectors using both CD and MSE losses with different values of k.
The EER(%) for i-vector/PLDA is equal to 9.54

k	CD Loss	MSE Loss
10	8.81	8.70
20	6.60	6.56
30	5.68	5.54
50	4.97	4.98
100	4.84	**4.45**
150	6.53	4.48

Results

DET Curve for the proposed vectors using MSE loss with different values of k.



Results

- Our proposed approach has obtained:
  - 25% relative improvement over x-vectors.
  - 53% relative improvement over i-vectors.
- Main advantage: No speaker labels required.
- Disadvantage: Background data is used in the testing phase.
- The good results are obtained mainly due to the usage of nearest neighbors for the testing i-vectors.

Conclusion

- We proposed a post processing for i-vectors, in order to increase their discriminative power.
- We trained a DNN using nearest neighbor i-vectors.
- The nearest neighbors were selected in unsupervised manner.
- We transformed the test i-vectors into a new speaker vectors.
- The results have shown that our proposed speaker vectors outperform the baseline systems.
- Avoids speaker labels at the cost of using the background data in the testing part.

Thanks