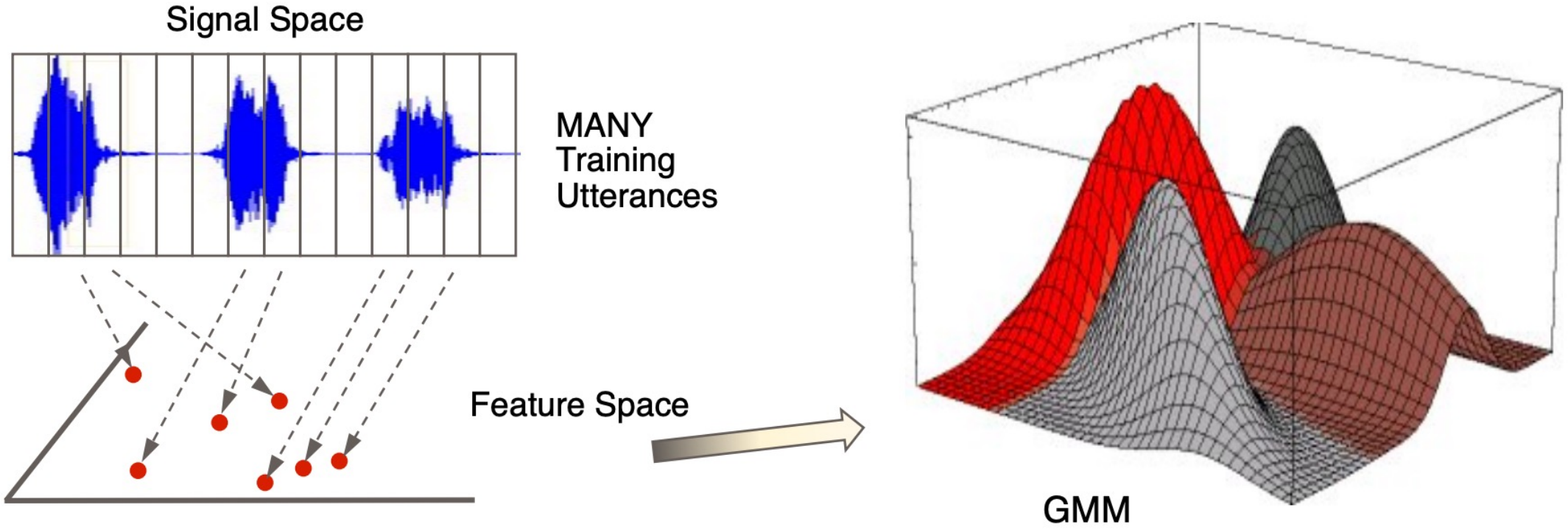
X-VECTORS: ROBUST DNN EMBEDDINGS FOR SPEAKER RECOGNITION

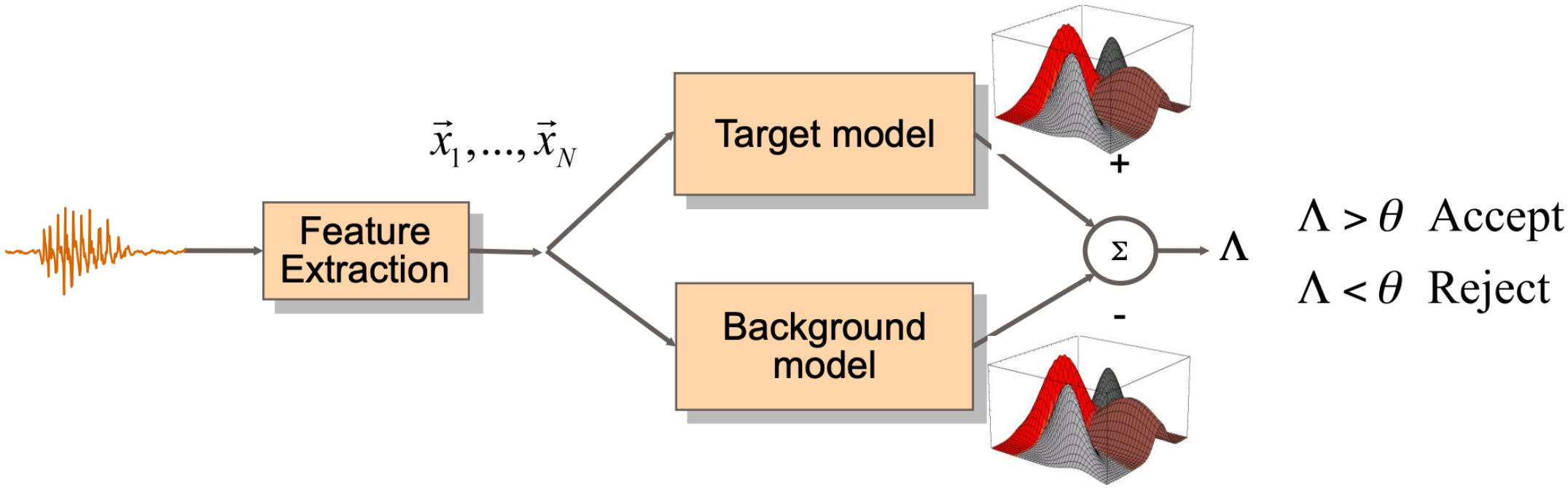
Deep Learning 이전의 기술들

초기 GMM



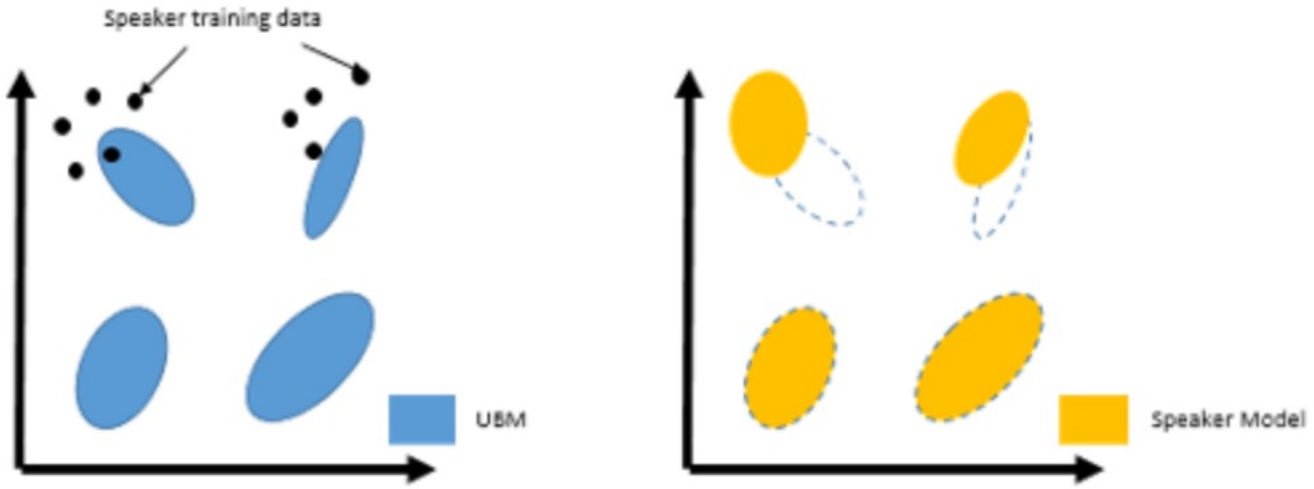
Deep Learning 이전의 기술들

GMM-UBM



Deep Learning 이전의 기술들

GMM-UBM



UBM은 Background model을 학습

Background model은 최대한 많은 화자 및 발음 음성을 수집 후 GMM으로 clustering
→ 인간의 평균 음성 특징이 나타남

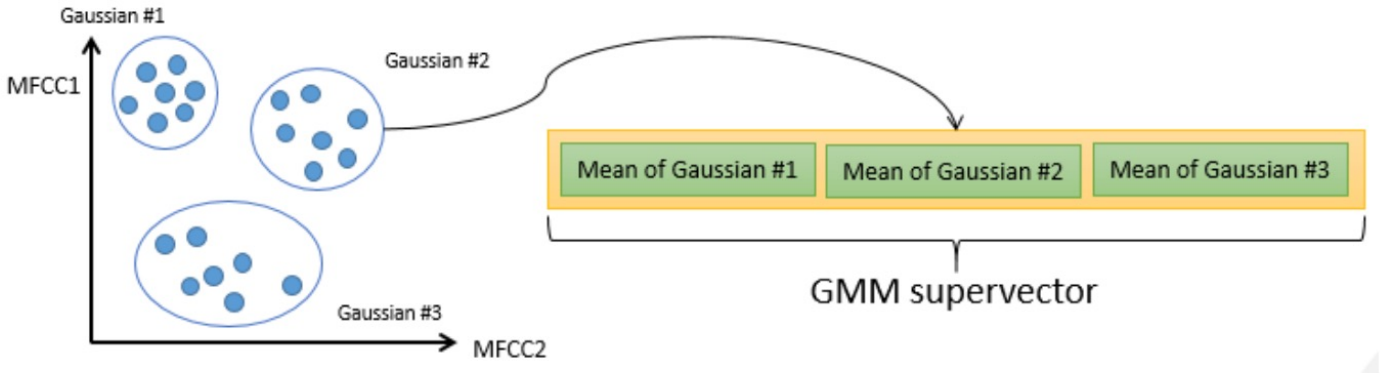
UBM의 파라미터를 초기 값으로 우리가 원하는 화자 별 GMM을 학습
→ Fine Tuning의 개념

이렇게 구한

출처 : https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=kwh1990&logNo=221485966657

Deep Learning 이전의 기술들

GMM-UBM



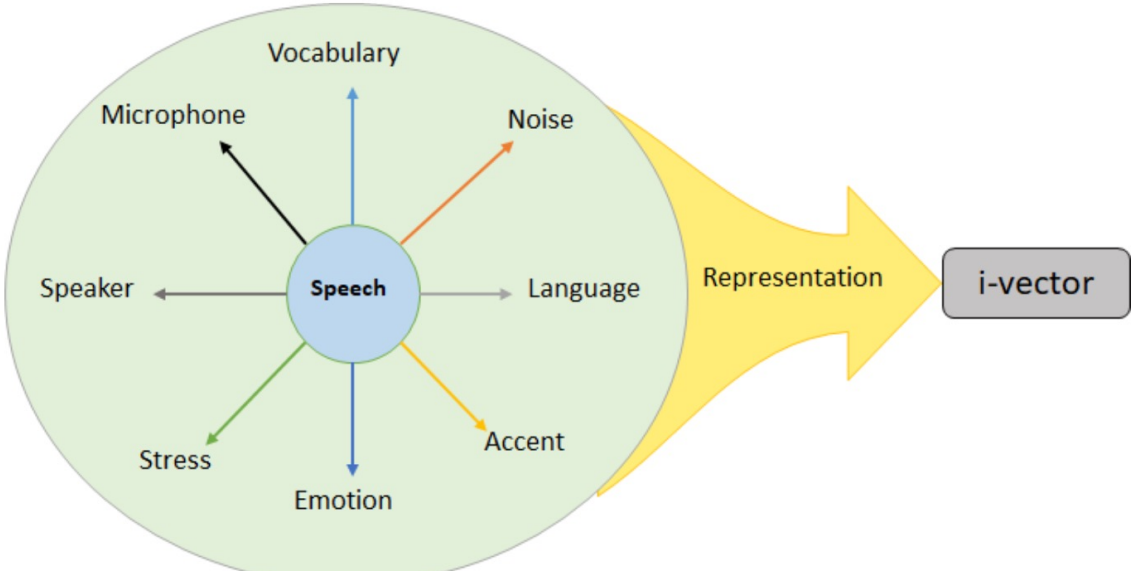
이후 각 Gaussian mixture별로 mean을 연결하여 supervector로 표현한다.

이 super vector를 SVM이나 DNN을 통해 학습시킨다.

출처 : https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=kwh1990&logNo=221485966657

Deep Learning 이전의 기술들

I - vector



$$\mathbf{m} = \mathbf{u} + \mathbf{Tw}$$

GMM supervector of the utterance

Utterance-independent GMM (UBM)

Total variability matrix (i-vector extractor)

i-vector (latent variable)

I-vector는 speaker, channel 등 모든 정보를 하나의 subspace로 표현

GMM-UBM을 기반으로 차원을 줄이고 다양한 정보를 담고있음

출처 : https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=kwh1990&logNo=221485966657

X-vector

abstract

DNN 기반 임베딩이 large-scale training dataset이 있을 때 더 효과적이였다

그러나 대량의 labeI된 dataset을 구하는건 어려운 일이다.

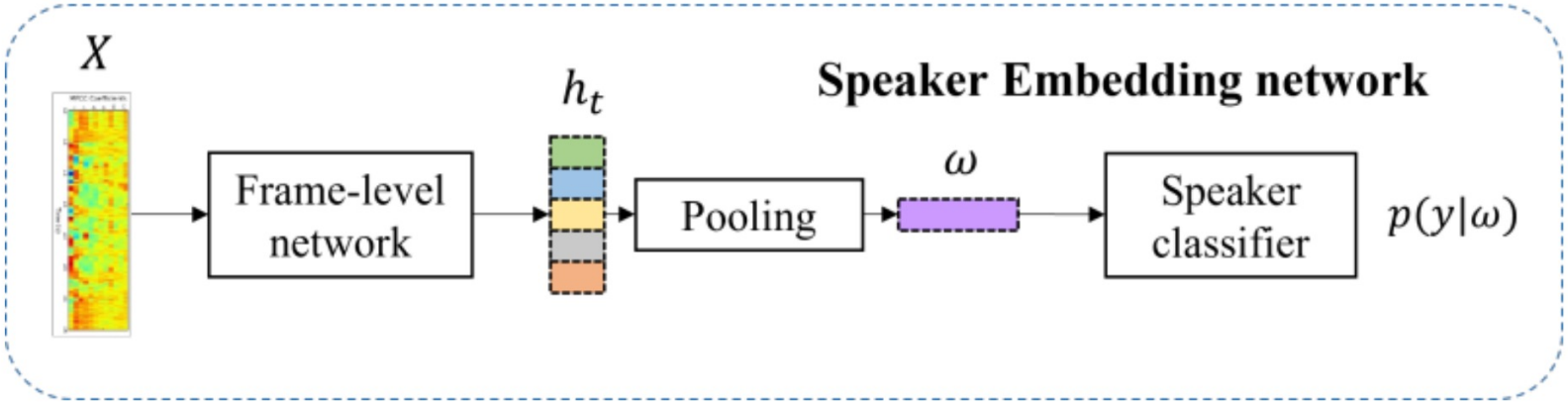
따라서 **added noise and reverberation**를 통한 augmentation으로 robustness를 올렸다.

Augmentation은 PLDA classifier에서는 효과적이였으나, i-vector에서는 아니였다.

DNN기반 X-vector에서는 supervise learning이기 때문에 augmentation의 효과를 봤다.

X-vector

X-vector 시스템 overview



X-vector

X-vector 시스템 overview

Layer	Layer context	Total context	Input x output
frame1	$[t-2, t+2]$	5	120x512
frame2	$\{t-2, t, t+2\}$	9	1536x512
frame3	$\{t-3, t, t+3\}$	15	1536x512
frame4	$\{t\}$	15	512x512
frame5	$\{t\}$	15	512x1500
stats pooling	$[0, T)$	T	1500Tx3000
segment6	$\{0\}$	T	3000x512
segment7	$\{0\}$	T	512x512
softmax	$\{0\}$	T	512xN

X-vector

X-vector 시스템 – 1. Filter Bank Input

Layer	Layer context	Total context	Input x output
frame1	$[t-2, t+2]$	5	120x512
frame2	$\{t-2, t, t+2\}$	9	1536x512
frame3	$\{t-3, t, t+3\}$	15	1536x512
frame4	$\{t\}$	15	512x512
frame5	$\{t\}$	15	512x1500
stats pooling	$[0, T)$	T	1500Tx3000
segment6	$\{0\}$	T	3000x512
segment7	$\{0\}$	T	512x512
softmax	$\{0\}$	T	512xN

1. DNN에 들어갈 acoustic feature를 추출한다.
→ Frame 단위로 MFCC같은 feature를 추출
→ VAD를 통해 speech만 이용
→ 논문에선 Frame당 24개의 feature를 사용

2. 첫 layer에는 t-2 ~ t+2까지 5개 frame을 넣는다.
→ 여기서 각 Layer는 TDNN layer를 사용한다.
(1D Conv) // 24 * 5 = 120 → 512

3. 두번째 layer는 첫번째 layer를 통과한 feature에 대하여 t-2, t, t+2 3개의 frame을 사용한다.
→ 512 * 3 = 1536

X-vector

X-vector 시스템 – 2. TDNN layer

Layer	Layer context	Total context	Input x output
frame1	$[t-2, t+2]$	5	120x512
frame2	$\{t-2, t, t+2\}$	9	1536x512
frame3	$\{t-3, t, t+3\}$	15	1536x512
frame4	$\{t\}$	15	512x512
frame5	$\{t\}$	15	512x1500
stats pooling	$[0, T)$	T	1500Tx3000
segment6	$\{0\}$	T	3000x512
segment7	$\{0\}$	T	512x512
softmax	$\{0\}$	T	512xN

4. layer3도 마찬가지이다
→ 여기는 t-3, t, t+3 frame 사용

5. layer 4,5는 t 시점의 frame만 사용한다.

X-vector

X-vector 시스템 – 3. statistics pooling

Layer	Layer context	Total context	Input x output
frame1	$[t-2, t+2]$	5	120x512
frame2	$\{t-2, t, t+2\}$	9	1536x512
frame3	$\{t-3, t, t+3\}$	15	1536x512
frame4	$\{t\}$	15	512x512
frame5	$\{t\}$	15	512x1500
stats pooling	$[0, T)$	T	$1500T$x3000
segment6	$\{0\}$	T	3000x512
segment7	$\{0\}$	T	512x512
softmax	$\{0\}$	T	512xN

일반적인 pooling이 아니라 stats pooling이라는 방식을 사용한다.

입력 사이즈가 일정하지 않기 때문에 sequence의 출력을 고정된 사이즈로 바꿔주는 역할을 한다.

5번째 layer 또한 frame 별로 output을 return 했을 것이다.

각 frame 마다 mean과 std를 구한다.

Mean과 std를 concat해서 pooling 한다.

1500 개의 mean과 std --> 3000 dim

X-vector

X-vector 시스템 – 4. segment layer

Layer	Layer context	Total context	Input x output
frame1	$[t-2, t+2]$	5	120x512
frame2	$\{t-2, t, t+2\}$	9	1536x512
frame3	$\{t-3, t, t+3\}$	15	1536x512
frame4	$\{t\}$	15	512x512
frame5	$\{t\}$	15	512x1500
stats pooling	$[0, T)$	T	1500Tx3000
segment6	$\{0\}$	T	3000x512
segment7	$\{0\}$	T	512x512
softmax	$\{0\}$	T	512xN

이후 TDNN Layer를 차례로 통과 한 후
Softmax로 speaker classification을 진행한다.

Embedding은 segment6 layer에서 추출한다.

X-vector

논문의 augmentation 방법

3.3. Data augmentation

Augmentation increases the amount and diversity of the existing training data. Our strategy employs additive noises and reverberation. Reverberation involves convolving room impulse responses (RIR) with audio. We use the simulated RIRs described by Ko et al. in [25], and the reverberation itself is performed with the multi-condition training tools in the Kaldi *ASpIRE* recipe [21]. For additive noise, we use the MUSAN dataset, which consists of over 900 noises, 42 hours of music from various genres and 60 hours of speech from twelve languages [26]. Both MUSAN and the RIR datasets are freely available from `http://www.openslr.org`.

We use a 3-fold augmentation that combines the original "clean" training list with two augmented copies. To augment a recording, we choose between one of the following randomly:

- **babble**: Three to seven speakers are randomly picked from MUSAN speech, summed together, then added to the original signal (13-20dB SNR).
- **music**: A single music file is randomly selected from MUSAN, trimmed or repeated as necessary to match duration, and added to the original signal (5-15dB SNR).
- **noise**: MUSAN noises are added at one second intervals throughout the recording (0-15dB SNR).
- **reverb**: The training recording is artificially reverberated via convolution with simulated RIRs.

X-vector

Classifier에 관한 문제점 → 논문 외 이야기

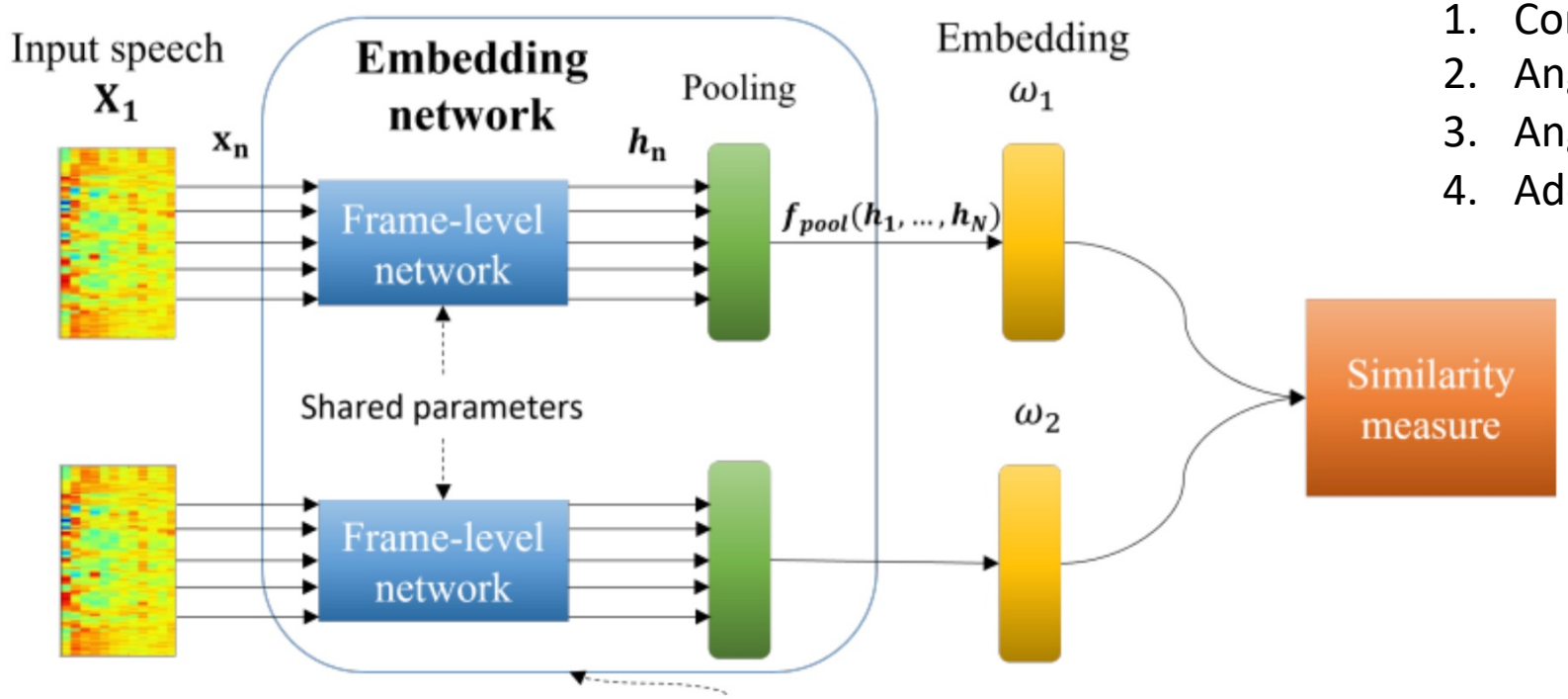
학습내에 존재하는 화자 중 입력 화자를 맞추는 Speaker Identification 같은 경우 softmax 기반 loss 학습이 적절하다.

하지만 시스템에 등록된 사용자의 목소리와 입력 목소리 둘다 학습 데이터내에 존재한다고 보장되지 않은 Verification에는 두 embedding간의 similarity를 계산한다.

softmax와 cross-entropy를 이용해도 embedding vector가 화자에 대한 정보를 최대한 추출하지만, metric mismatch는 성능을 제한할 수 있다.

X-vector

Classifier에 관한 문제점 → 논문 외 이야기



1. Contrastive loss
2. Angular softmax loss
3. Angular margin softmax loss
4. Additive angular margin softamx loss