Speaker Recognition

HLT Tutorial

Jesus Villalba

Roadmap

- Introduction:
  - Task, Terminology, Framework
- Neural Embeddings:
  - X-Vectors
- Encoder Architectures:
  - TDNN
  - ResNet, Res2Net
  - ECAPA-TDNN
- Pooling Mechanisms:
  - Statistics
  - MHAtt,
- Loss Functions:
  - Cross-Entropy
  - AM-Softmax, AAM-Softmax
- Back-end:
  - Cosine Scoring, PLDA
  - Score-Normalization, Calibration
- Other Topics in Speaker Recognition:
  - Spoofing Attacks
  - Adversarial Attacks

Introduction

Task, Terminology, Framework

Extracting Information from Speech

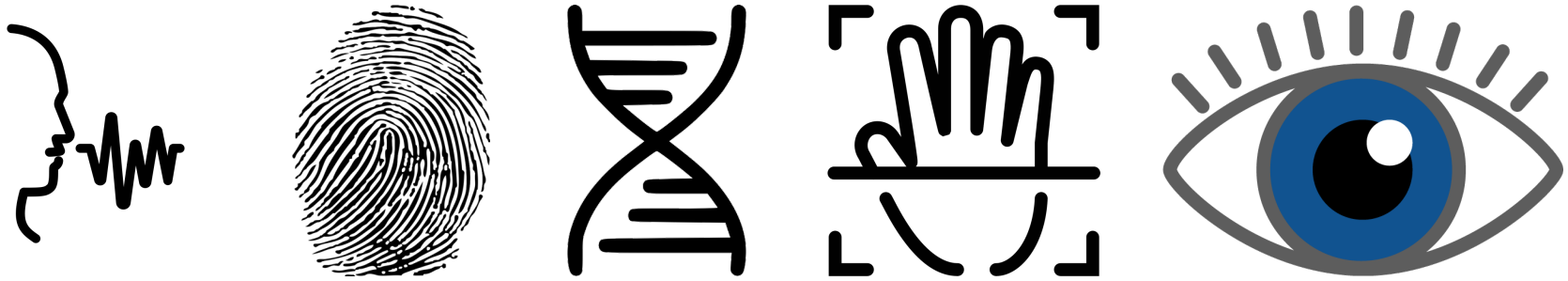
Goal: Automatically extract information transmitted in speech signal

Speech Signal

- Speech Recognition → Words
  "How are you?"
- Language Recognition → Language Name
  English
- Speaker Recognition → Speaker Name
  James Wilson
- Speaker Diarization → Who Speaks When
  Bob: Meeting tonight?
  Alice: yes!
- Emotion Recognition → Emotional State
  Happy: 90%
- Pathology Detection → Medical Conditions
  Parkinson's: 70%, Covid-19: 30%

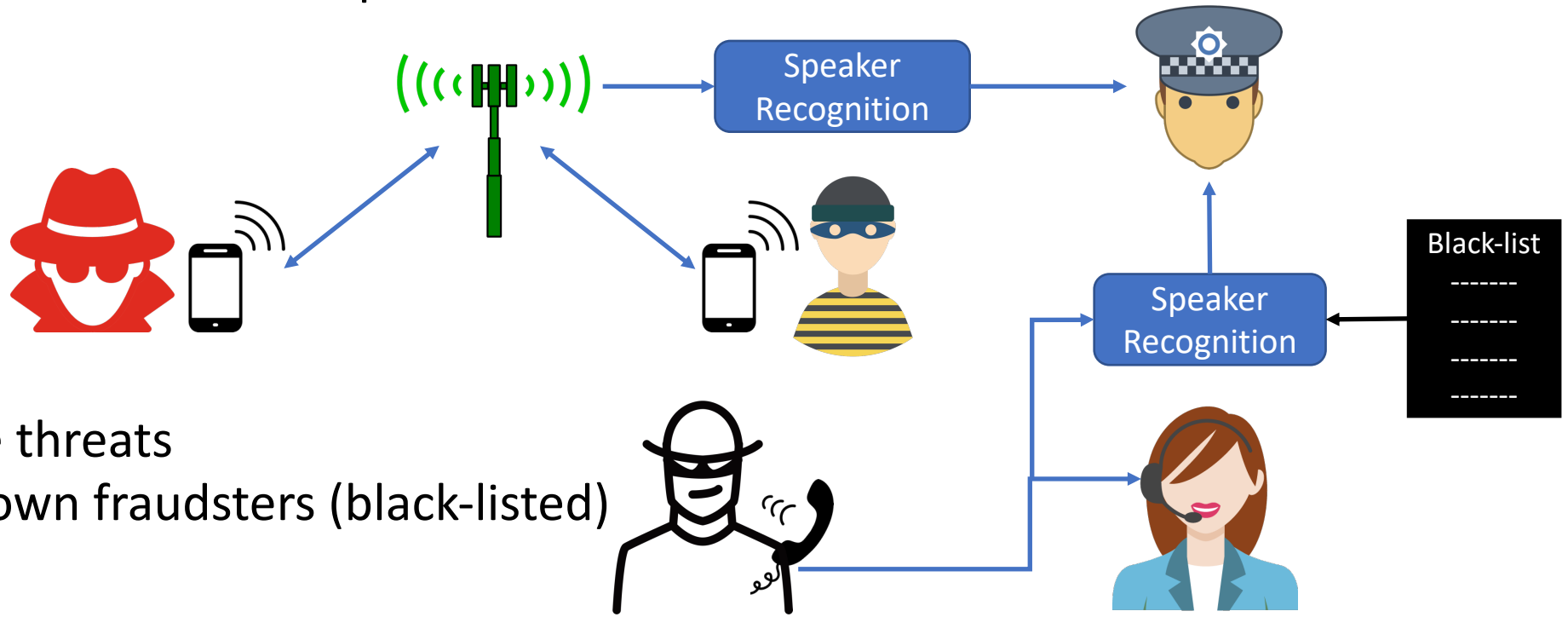
Voice Biometrics / Speaker Recognition

- Biometric modality consisting in recognizing people from the characteristics of their voices
- Properties of speech influenced by:
  - Anatomy:
    - Shape and size of voice production organs (vocal track, larynx, nasal cavity)
  - Behavioral patterns (Manner of Speaking):
    - Accent, rhythm, intonation style, pronunciation pattern, vocabulary
- Advantages:
  - Easy to use -> speech is a natural way of communication
  - Non-intrusive -> Well accepted by users



Speaker Recognition Applications

- Law Enforcement and Forensics
  - Search for criminals in telephone conversations

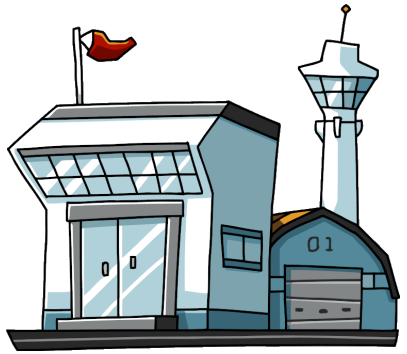


- Telephone threats
- Detect known fraudsters (black-listed)

- Voice sample taken at a crime scene can convict or discharge in court

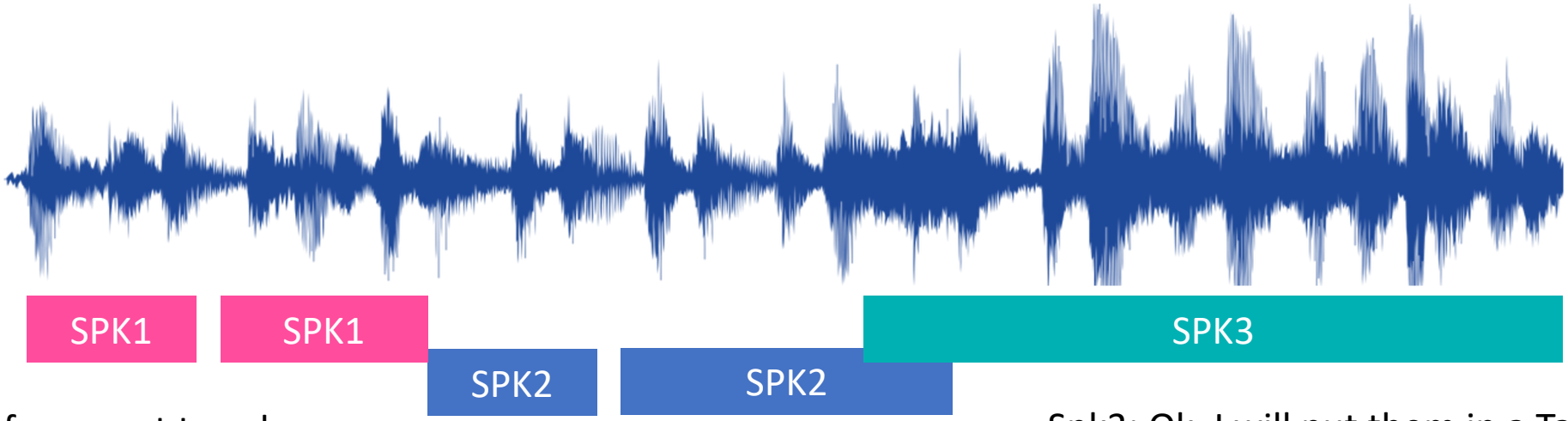
Speaker Recognition Applications

- Identity Authentication and Access Control
  - Access to high security physical facilities:
    - Military base, airports, government buildings
  - Computer networks, web services
  - Password Reset
  - Telephone/electronic banking



Speaker Recognition Applications

- Meeting Transcription
  - Enrich adding meta-data
  - Who said what in a meeting?



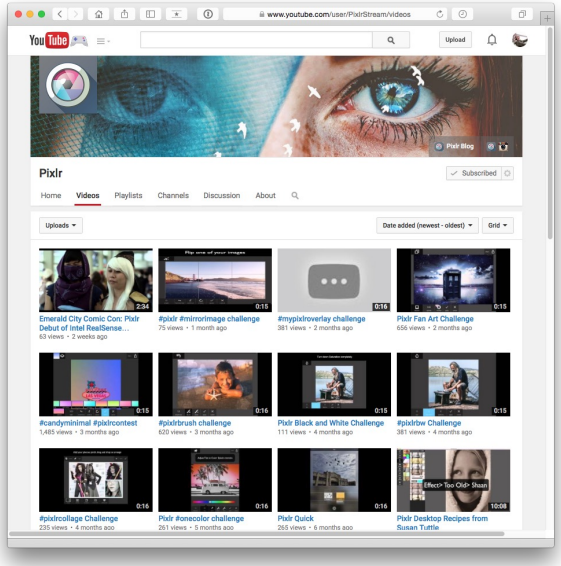
Spk1: If we want to solve automatic speech recognition

Spk2: You need to first breakdown those results.

Spk3: Ok, I will put them in a Table or graph. I will also detail the algorithm…

Speaker Recognition Applications

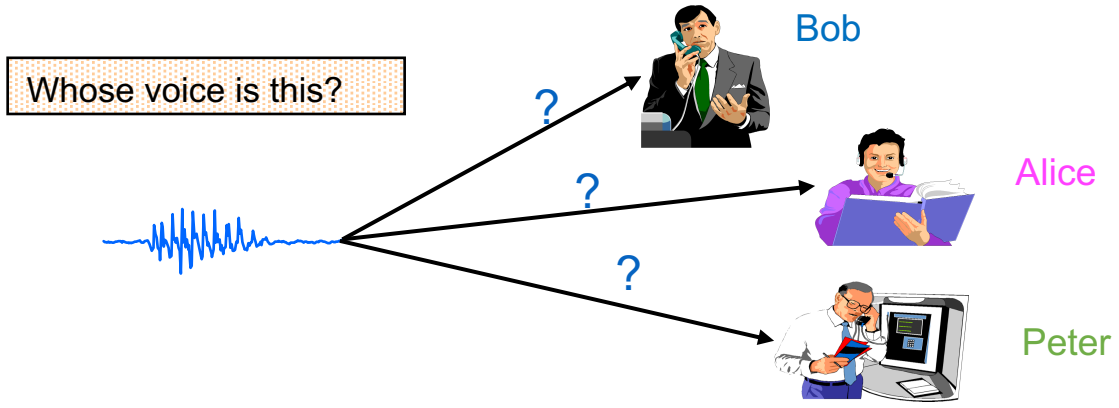
- Audio-Visual Media Indexing
  - Broadcast TV, YouTube
  - Add Meta-data
    - Who is speaking in this video
    - When is he/she speaking
  - Improve document indexing and search



- Personalization
  - Voice Assistant adapt to the user
    - Play user's favorite music
    - Read user's emails
    - Parental control

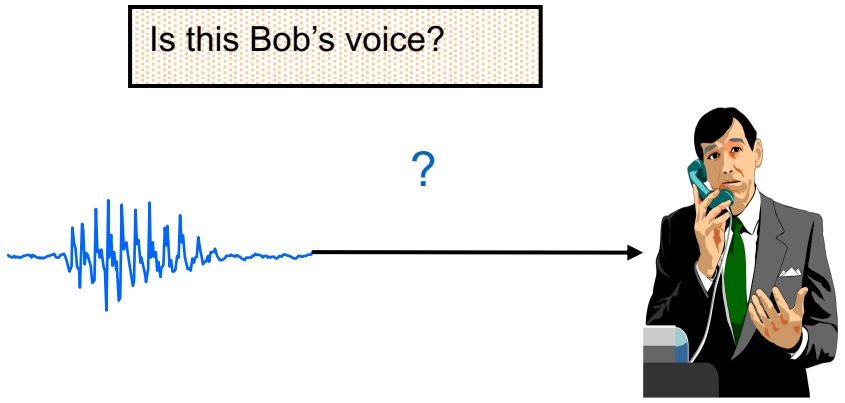
Speaker Classification/Identification

- Determine whether a test speaker matches one of a set of known speakers
- Referred as closed-set identification.

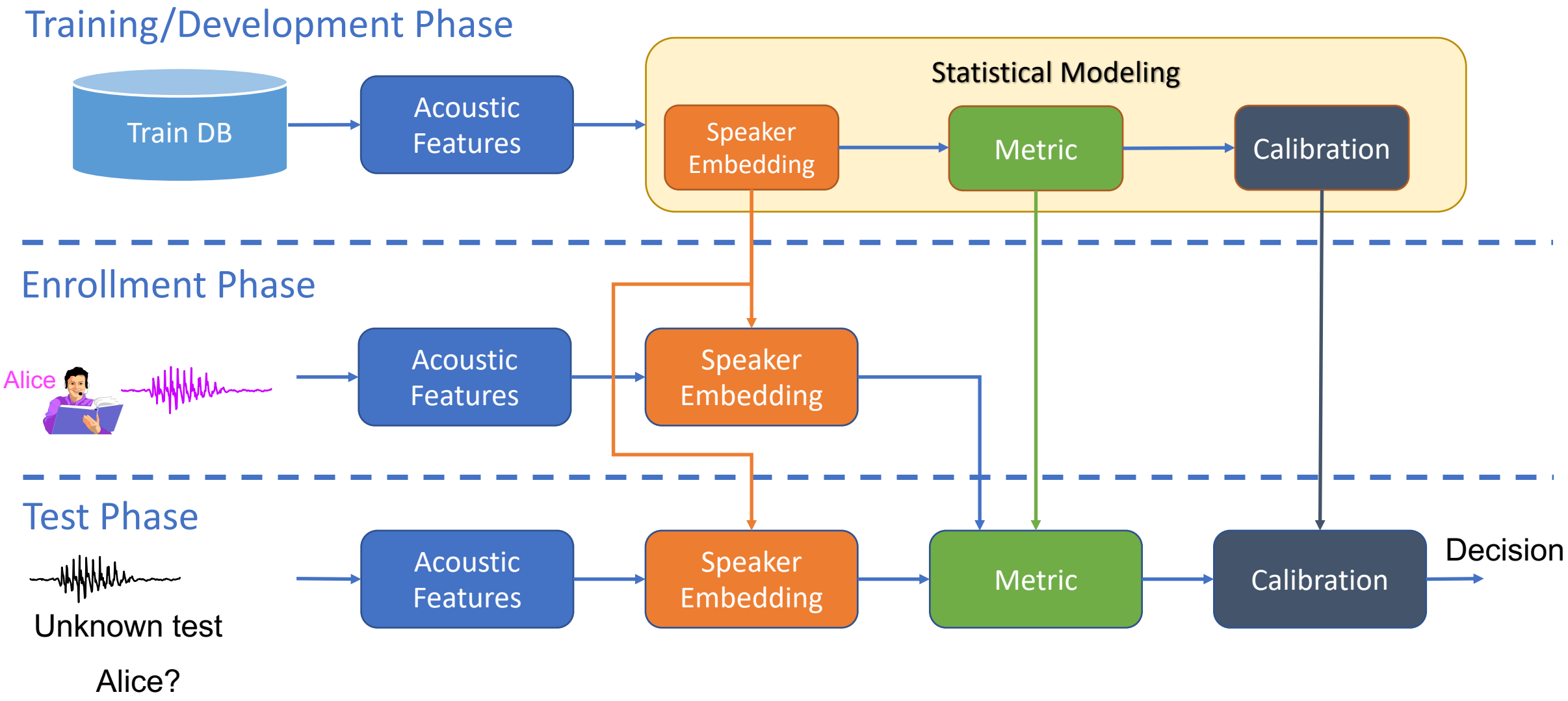


Speaker Verification

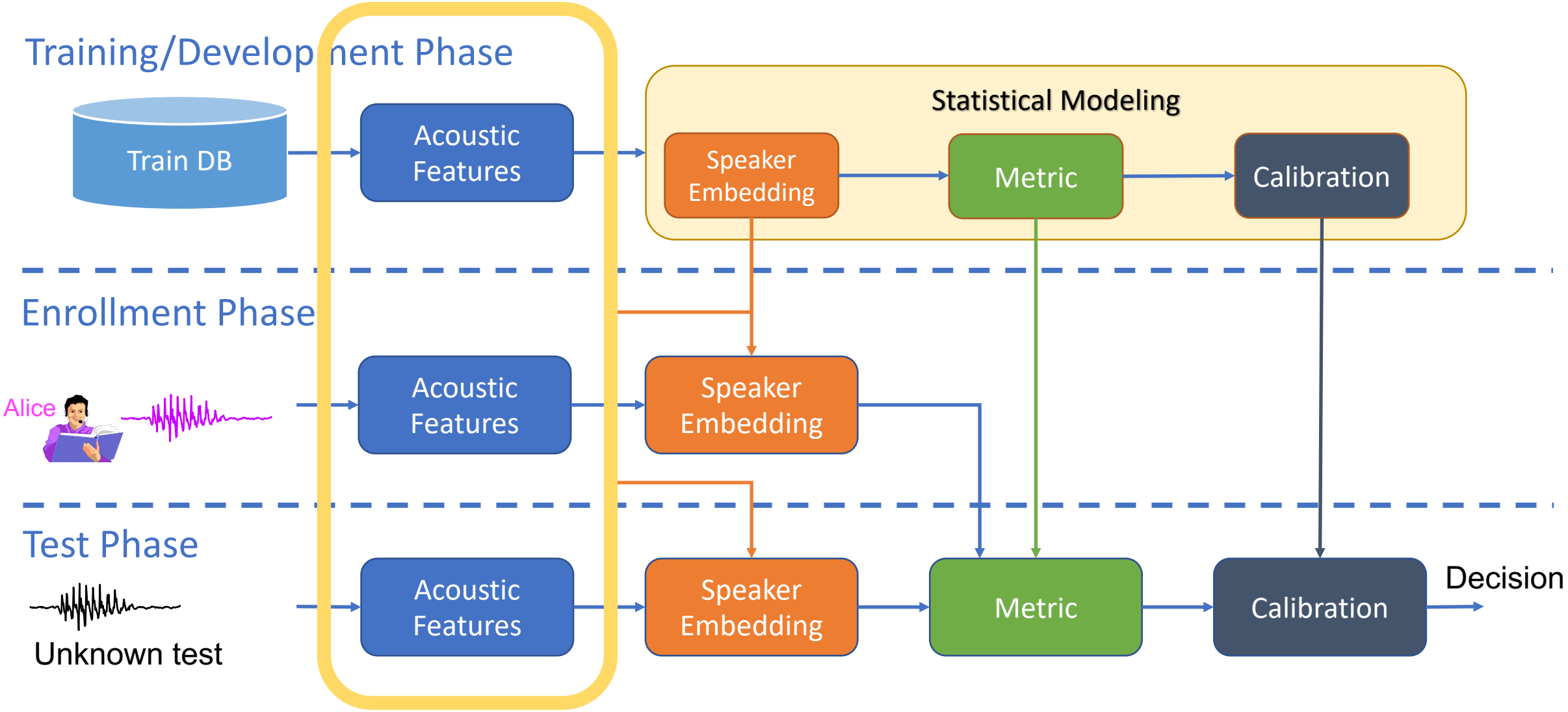
- Determine whether a test speaker matches a specific target speaker
- Unknown speech may come from a large set of unknown speakers – referred as open-set verification
- This is most common task in speaker recognition, close to real application.



Speaker Verification Pipeline

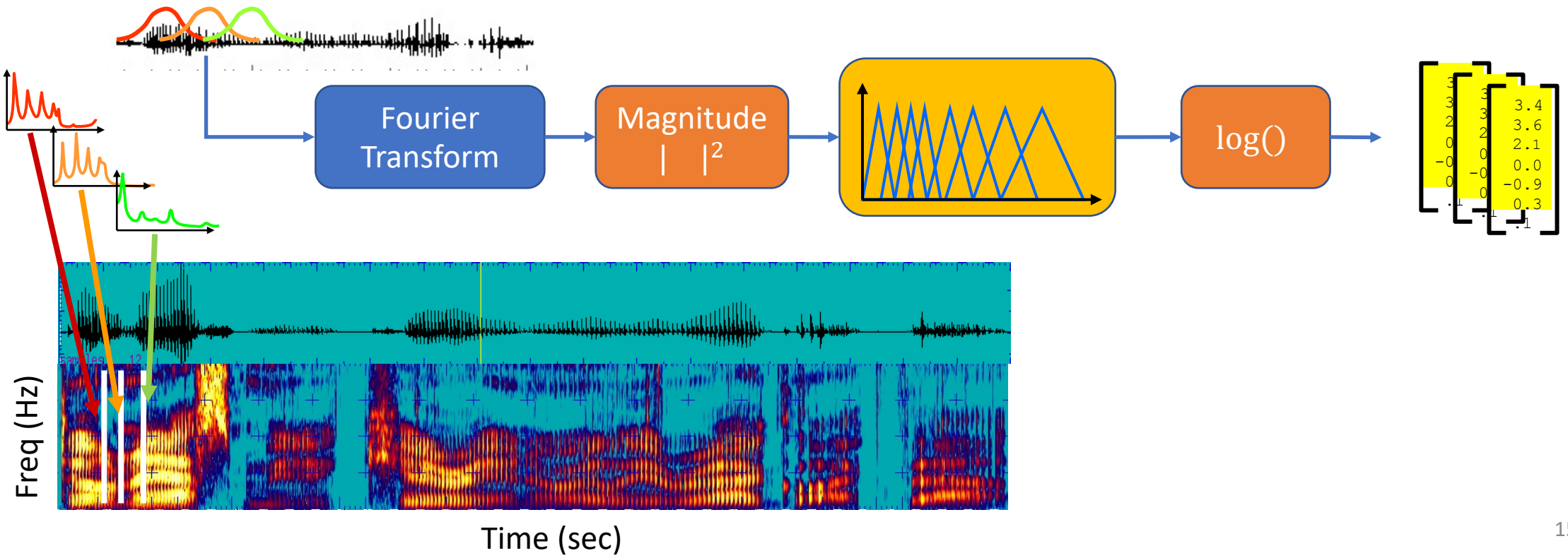


Speaker Verification Pipeline: Acoustic Features

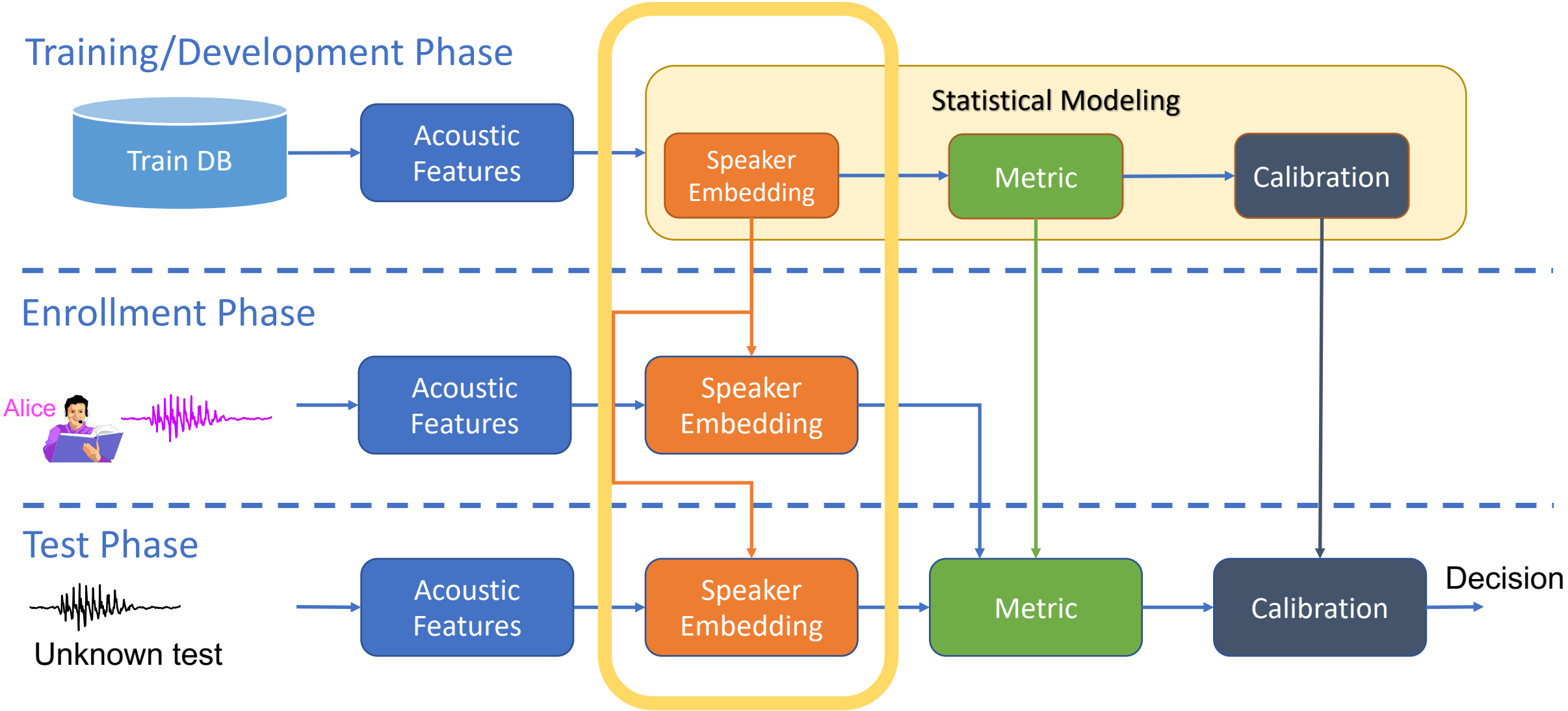


Acoustic Features

- Time sequence of acoustic features is needed to extract the speech information
  - Short-time spectral features are computed using a sliding window
  - Time-frequency representation of the signal
  - Filter bank in log Mel scale (Mel filtered spectrogram)



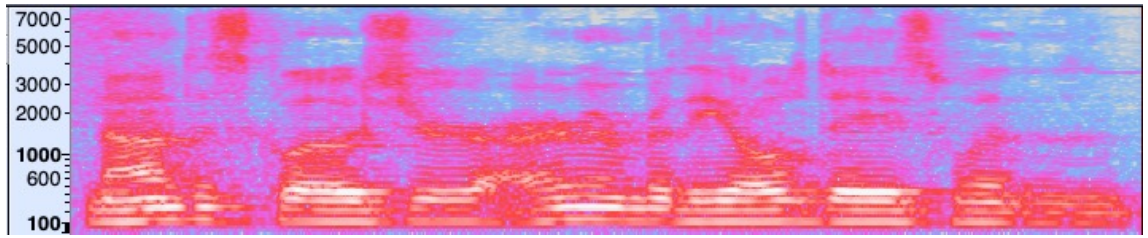
Speaker Verification Pipeline: Acoustic Features



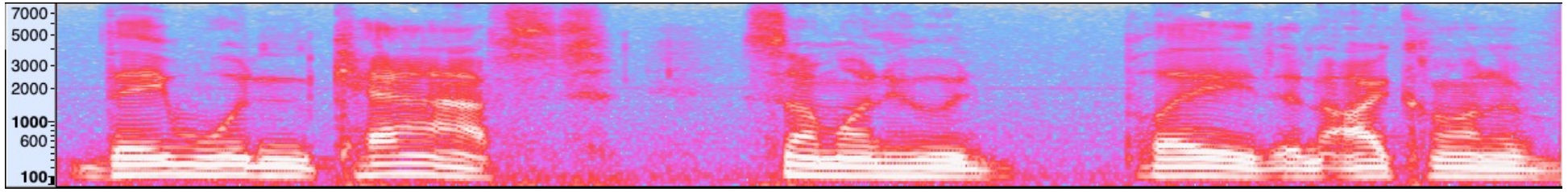
Speaker Embeddings

- Difficult to Compare Enrollment and Test recordings using Acoustic Features
- They have different durations, different number of feature vectors
  - Cannot calculate something like Euclidean distance between feature matrices
- The sequence of phonemes is different in each one of them
  - Early systems were text-dependent, same phrase in enrollment and test

Enroll.:

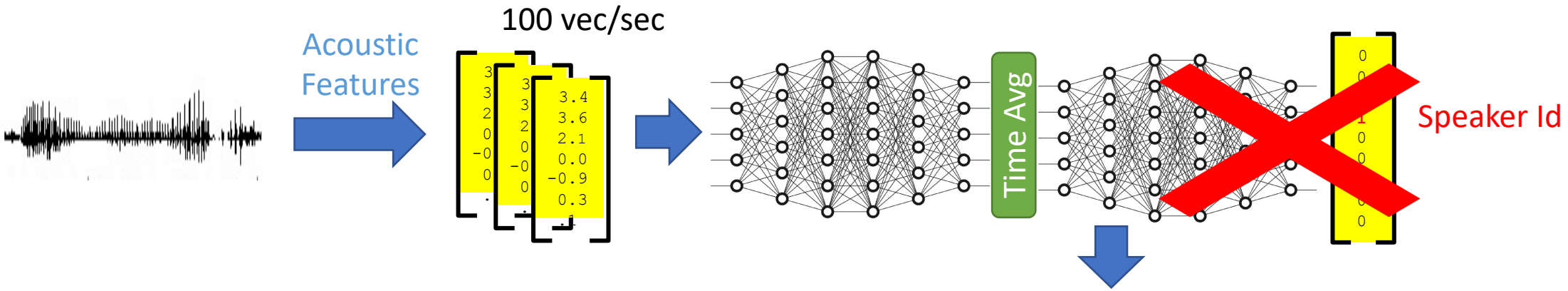


Test:



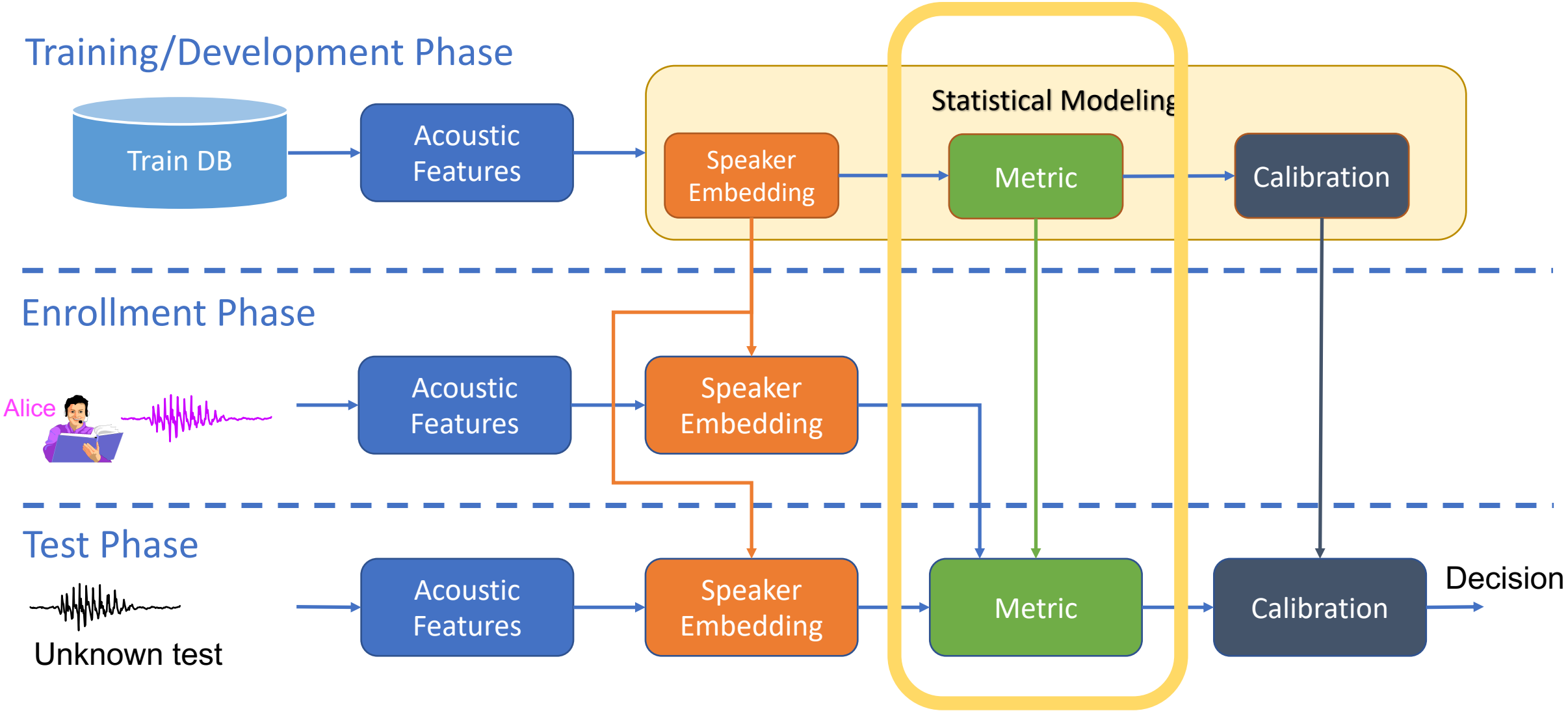
Speaker Embeddings

- Modern solution: **Speaker Embeddings**
  - Transform variable length recording into a single vector - Embedding
  - Embedding retains the speaker identity information



- Train the network to classify speakers
  - Large dataset > 1K speakers
- After training:
  - Take intermediate layer as speaker embedding

Speaker Verification Pipeline: Acoustic Features



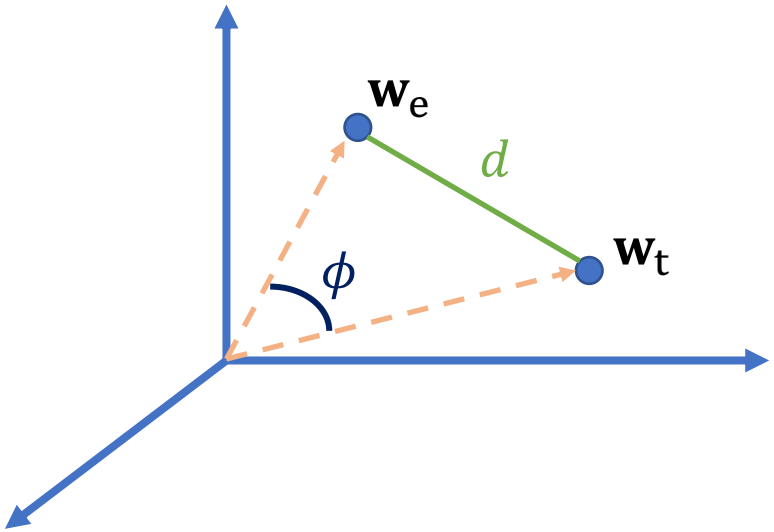
Metric (Back-end)

- Assume that:
  - $\mathbf{w}_e$ spk. embedding from enrollment utterance of speaker **X**
  - $\mathbf{w}_t$ spk. embedding from test utterance of person that claims to be speaker **X**

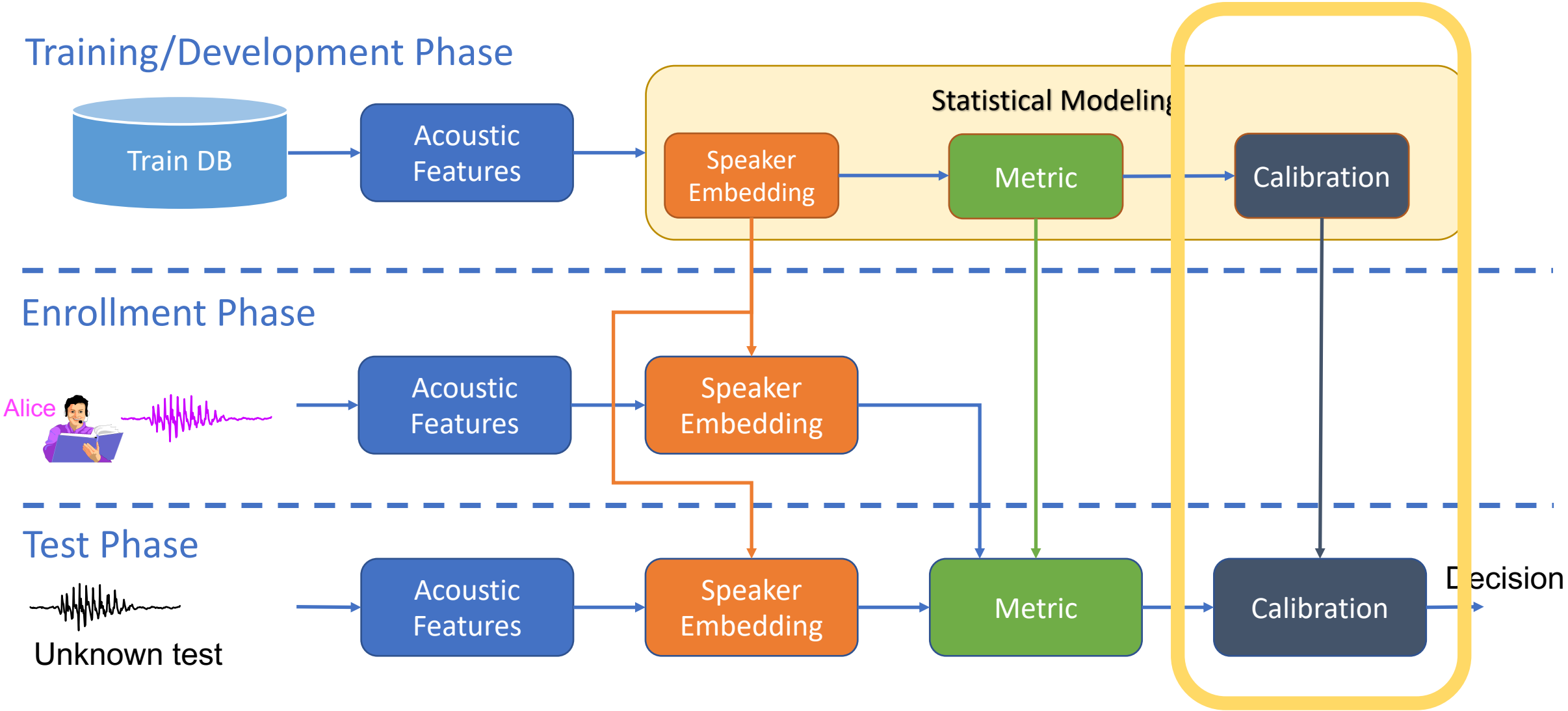
- The Metric compares enrollment and test embeddings $\mathbf{w}_e$, $\mathbf{w}_t$
  - Cosine scoring:

$$s = \cos(\phi) = \frac{\mathbf{w}_e^T \mathbf{w}_t}{\|\mathbf{w}_e\|_2 \|\mathbf{w}_t\|_2}$$

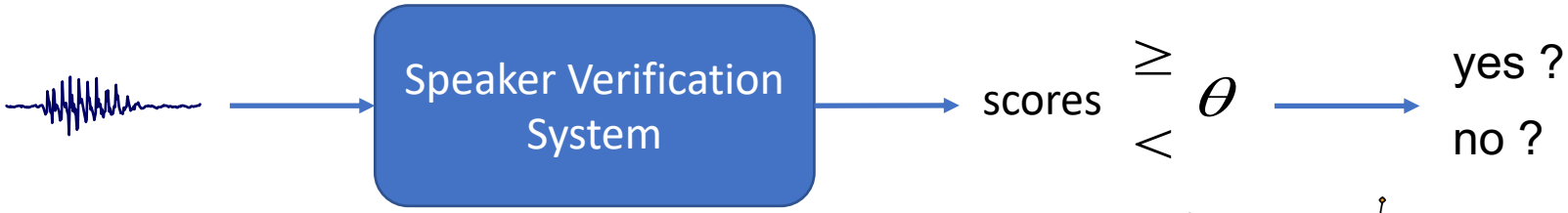
  - PLDA



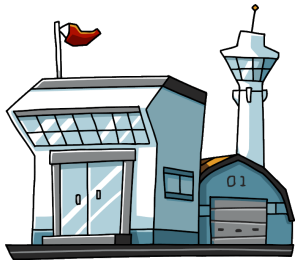
Speaker Verification Pipeline: Acoustic Features



Calibration, the Art of Choosing the Threshold

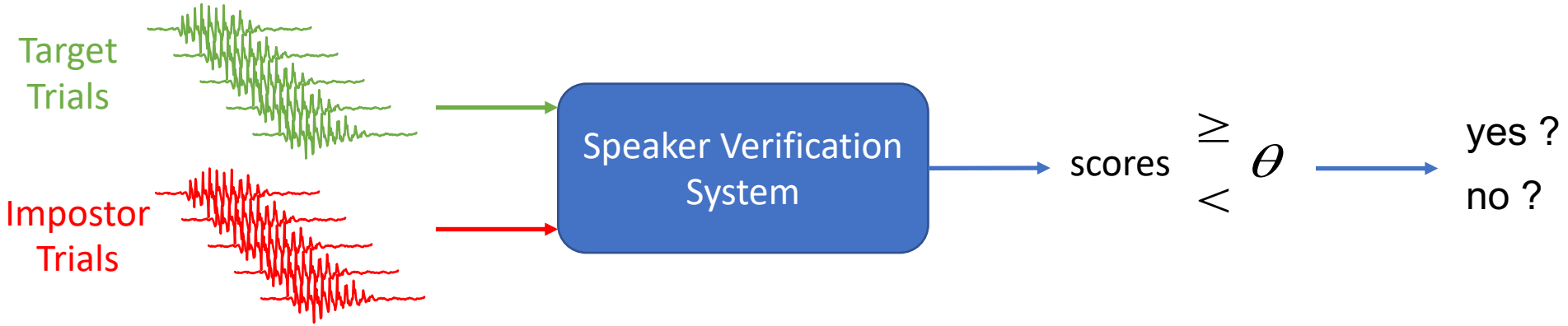


- How do we choose the decision threshold?

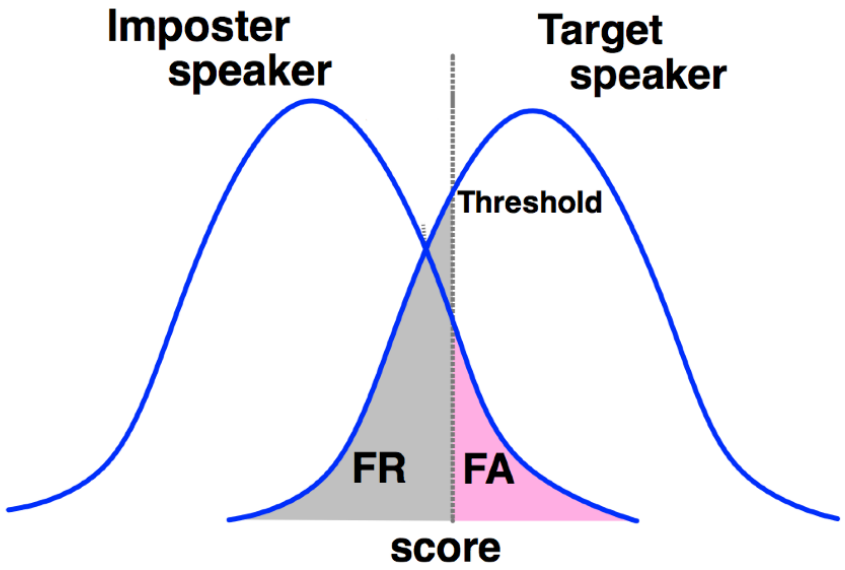


- High Security Application -> High decision threshold.
- Low Security Application: Low decision threshold

Performance Metrics

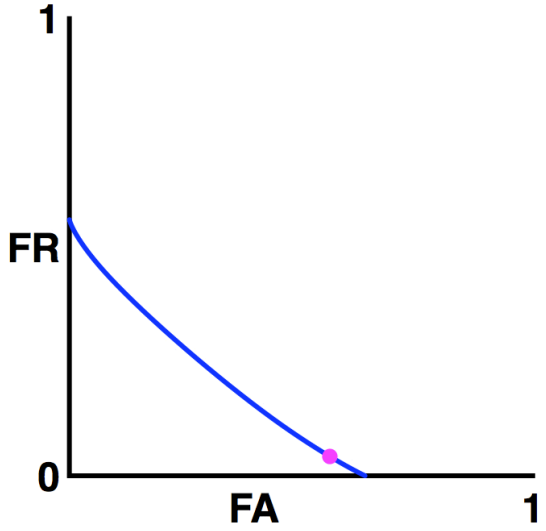


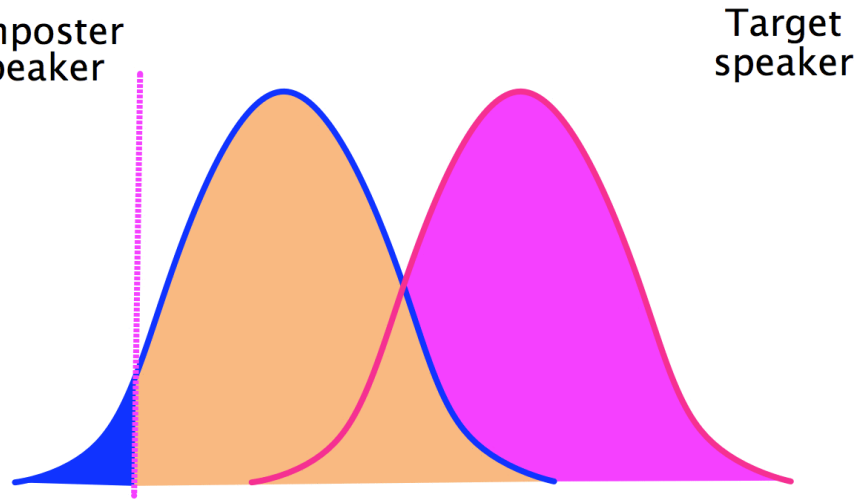
- Put >1k of target and impostor trials into the systems and count the errors
- Types of Errors:
  - Miss/False rejection:
    - True speaker is classified as impostor
    - Metric: Miss rate P_{Miss}
  - False alarm:
    - Impostor is classified as the true speaker
    - Metric: False alarm rate P_{FA}



Performance Metrics

- Detection Error Trade-off (DET)





- True positive
- True Negative
- Miss, false rejected
- False Accepted

- Equal Error Rate (EER)

$$P_{\text{Miss}}(\theta_{\text{EER}}) = P_{\text{FA}}(\theta_{\text{EER}})$$

$$\theta = \text{decision threshold}$$

- Detection Cost Function (DCF)

$$C_{\text{Det}}(\theta) = P_{\text{Miss}}(\theta) + \beta P_{\text{FA}}(\theta) \quad \text{with } \beta = \frac{1-P_{\text{target}}}{P_{\text{target}}}$$

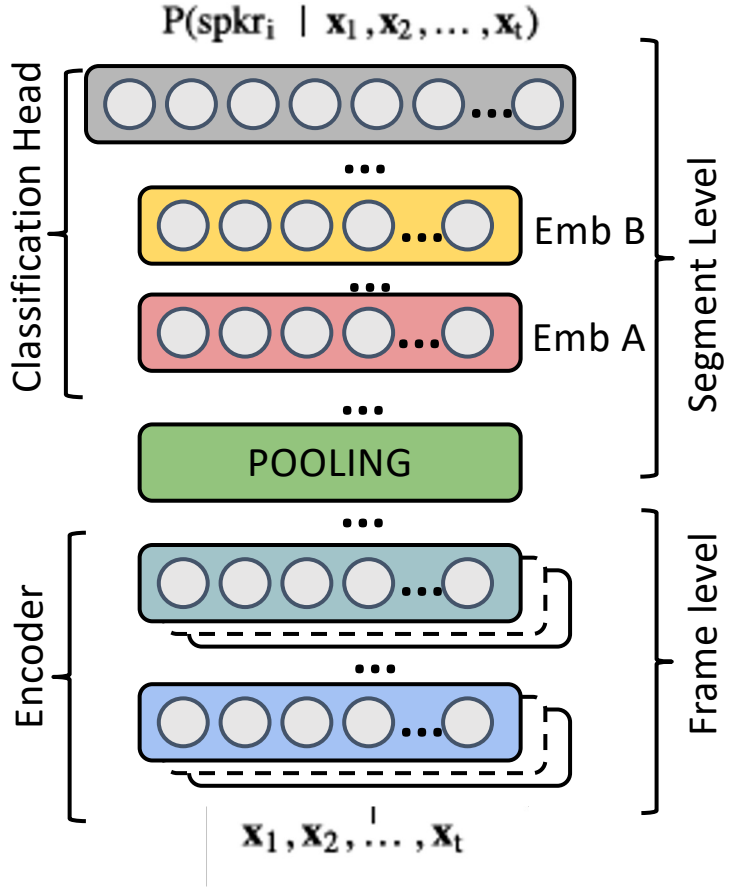
$$\text{Minimum } C_{\text{Det}} = \min_{\theta} C_{\text{Det}}(\theta)$$

Neural Embeddings

X-Vectors

X-Vectors

- X-Vector network has three parts:
  - Encoder:
    - Input: Acoustic features log-Mel spectrogram.
    - Output: frame level hidden representations.
  - Pooling:
    - Summarizes representations into a single vector / utt.
    - Mean, Mean+Stddev, …
  - Classification Head:
    - Predicts posterior probabilities for the training speakers.
    - Embedding extracted from middle layer.
- Categorical cross-entropy loss: $$L = -\sum_{i=1}^{M} \log P\left(y_i = t_i | \mathbf{X}_i\right)$$
  - Compares each utterance against all the speakers in the training data.
  - Does not need hard negative sampling.



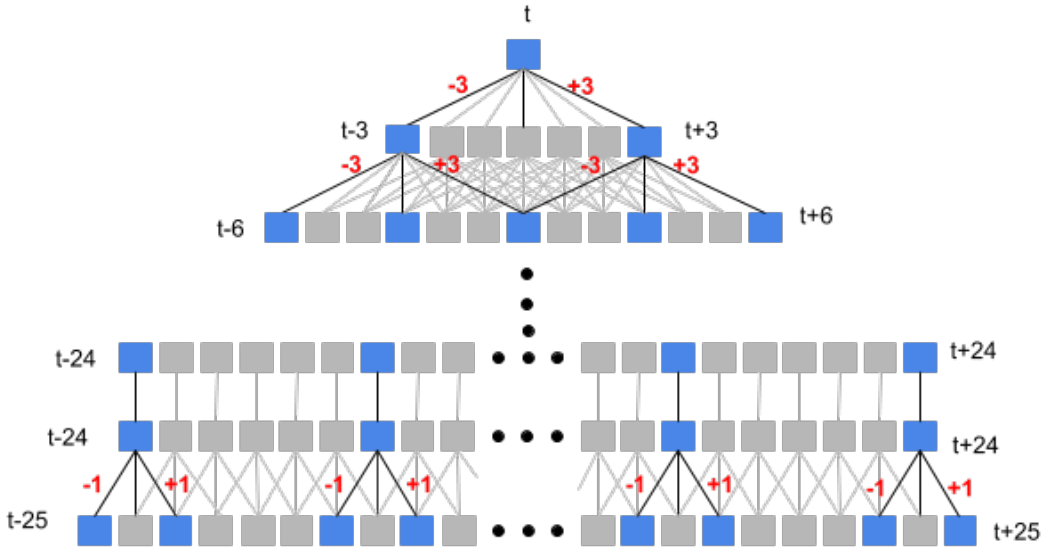
Snyder, David, et al. "Deep Neural Network Embeddings for Text-Independent Speaker Verification." *Interspeech*. 2017.
Snyder, David, et al. "X-vectors: Robust dnn embeddings for speaker recognition." *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018.

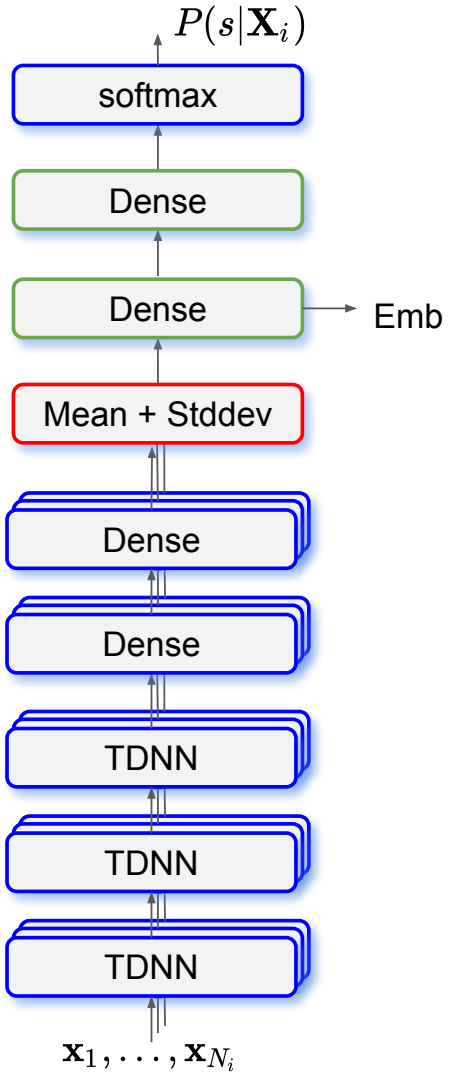
Neural Embeddings Architectures

TDNN, ResNet2D, ECAPA-TDNN

TDNN x-Vector

- TDNN Encoder for x-Vector
  - TDNN layer: 1d Dilated Convolution
  - Aggregates information over a larger receptive field as it gets deeper.
  - Dilation makes the receptive field to grow faster.



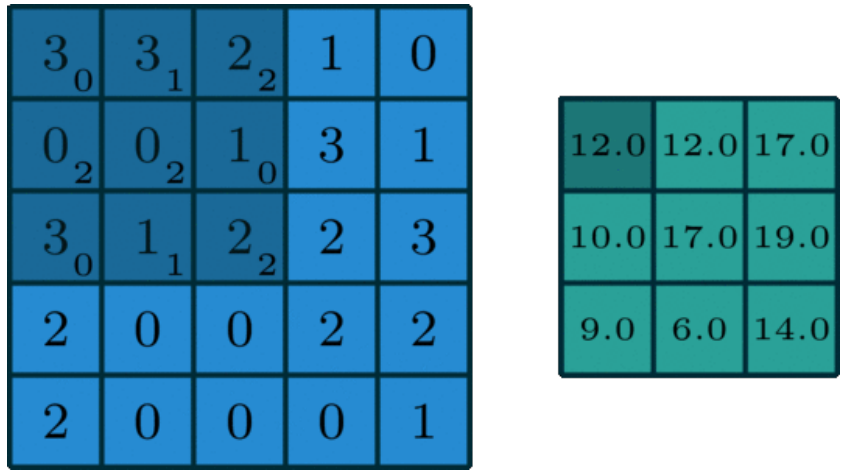


Snyder, David, et al. "Deep Neural Network Embeddings for Text-Independent Speaker Verification." *Interspeech*. 2017.

Residual Networks with 2D Convs

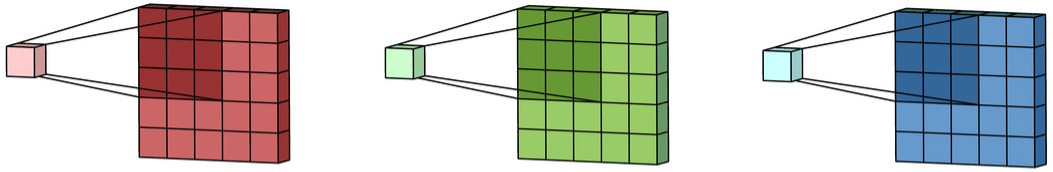
2D Conv with 1 in/out channels

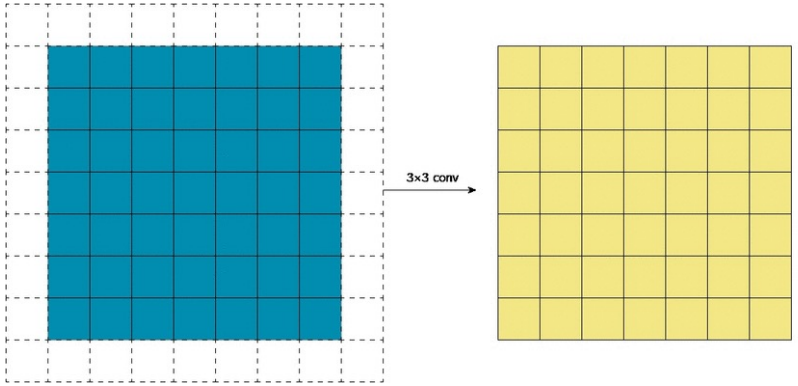
- Feature Map: (B, 1, F, T) -> (B, 1, F, T)



2D Conv with Cin/Cout channels

- Feature Map: (B, Cin, F, T) -> (B, Cout, F, T)
- Kernel : (Cout, Cin, k, k)





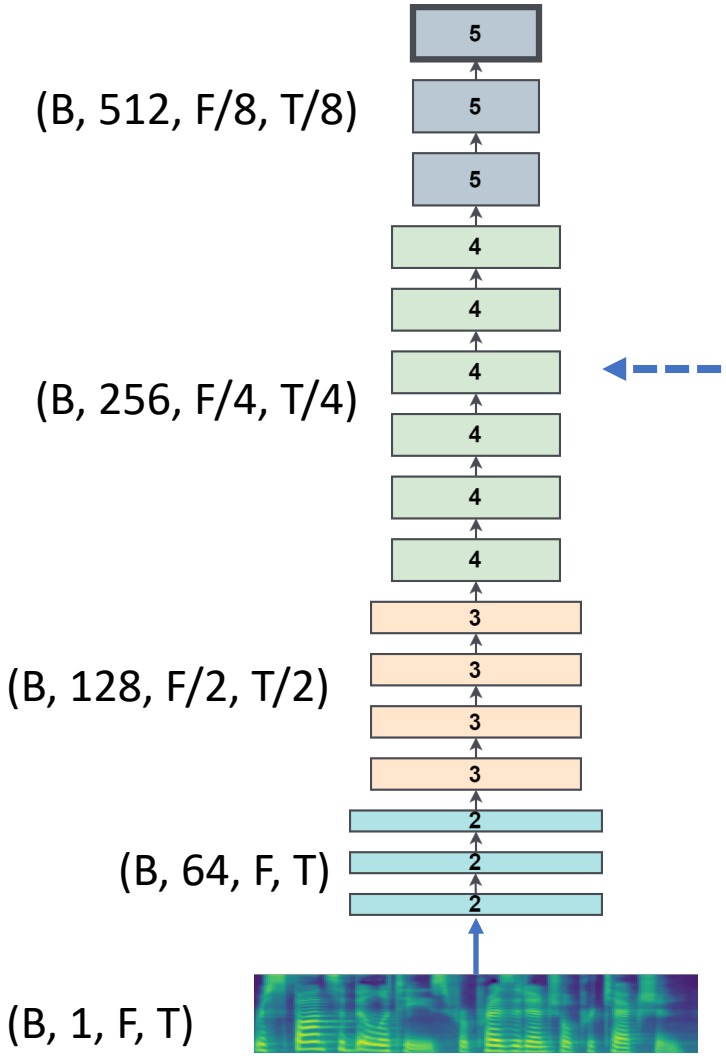
Strided Convolution

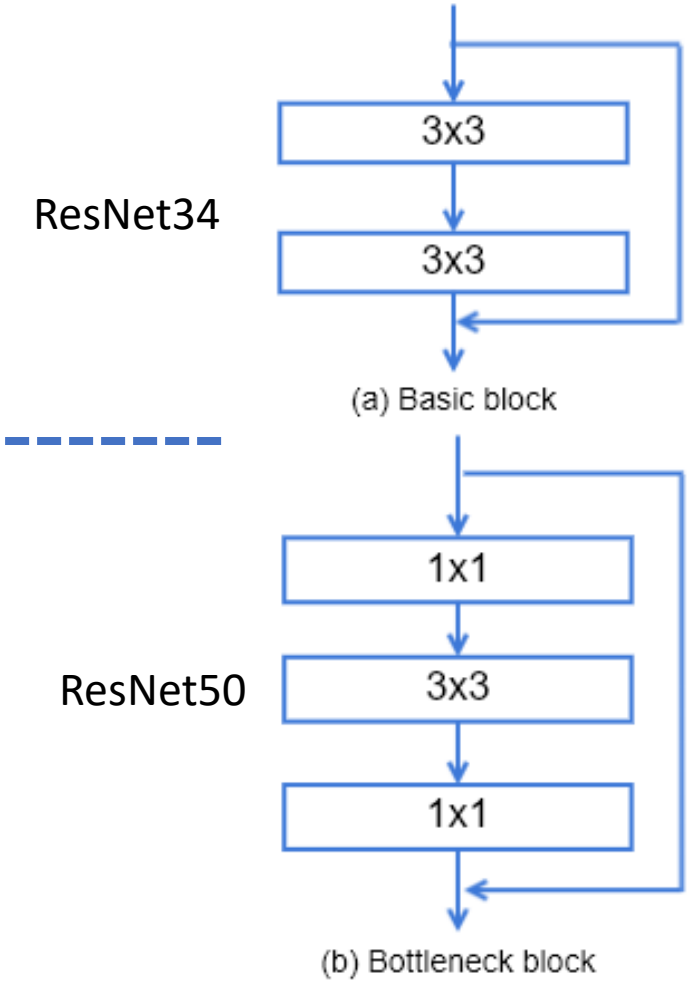
- Downsamples feature maps
- Feature Map: (B, Cin, F, T) -> (B, Cout, F/stride, T/stride)
- The receptive field of the next layers is multiplied by the stride.

Animations [from https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1](https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1)

ResNet 2D

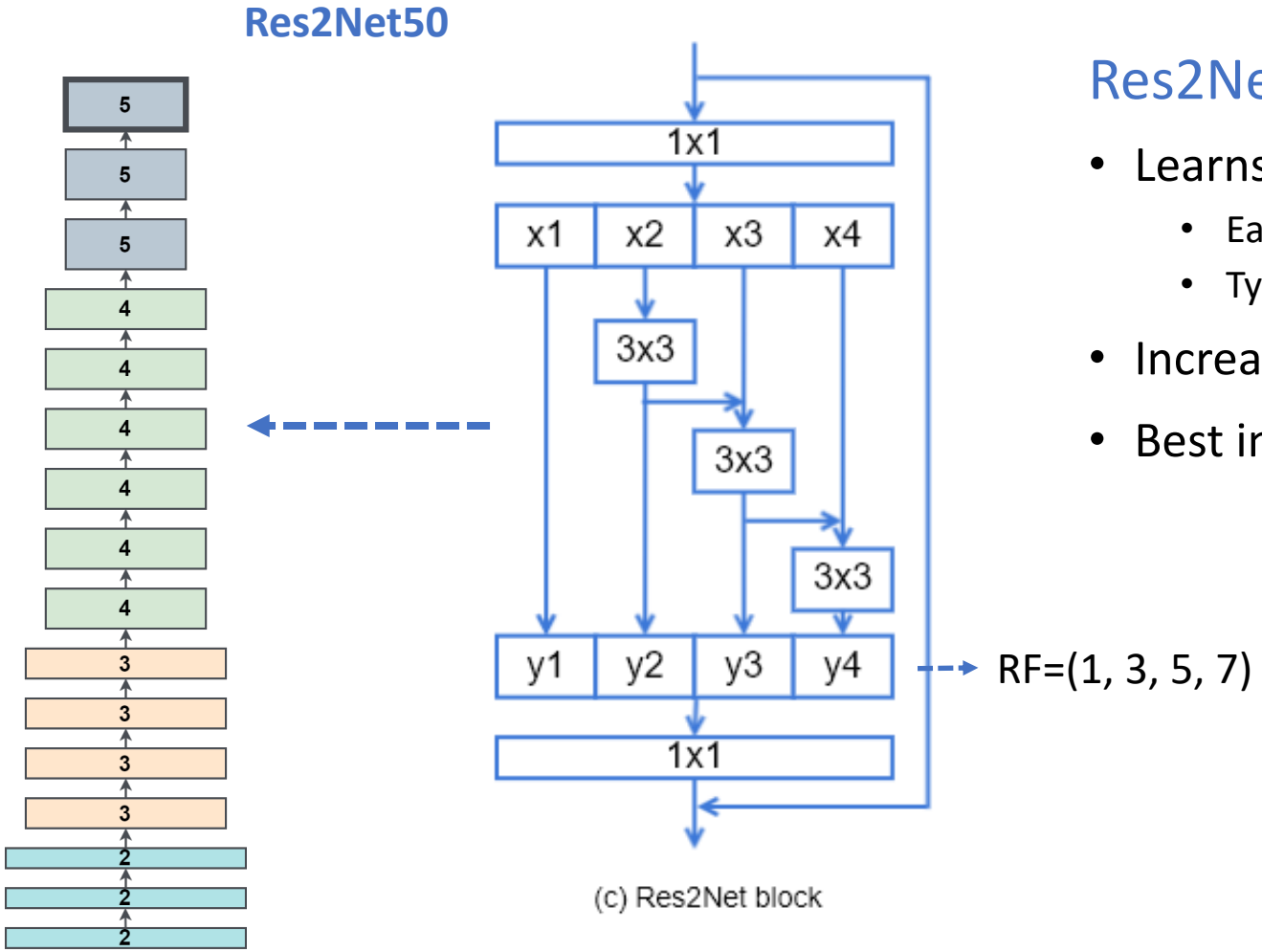
ResNet Scheme





He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

Res2Net



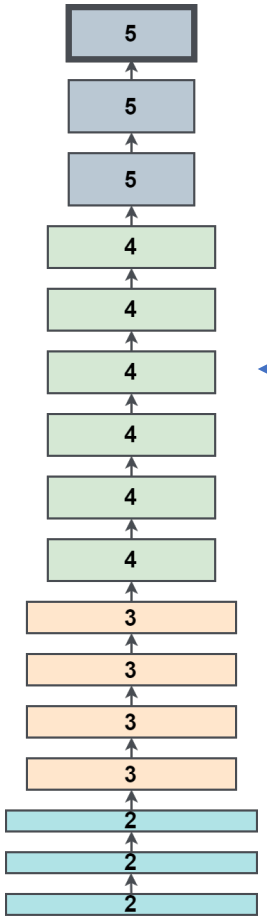
Res2Net

- Learns multi-scale features:
  - Each channel group observes a different receptive field
  - Typically use scale=4 or 8
- Increases global receptive field in the full network
- Best in VoxCeleb, SRE19AV, SRE21

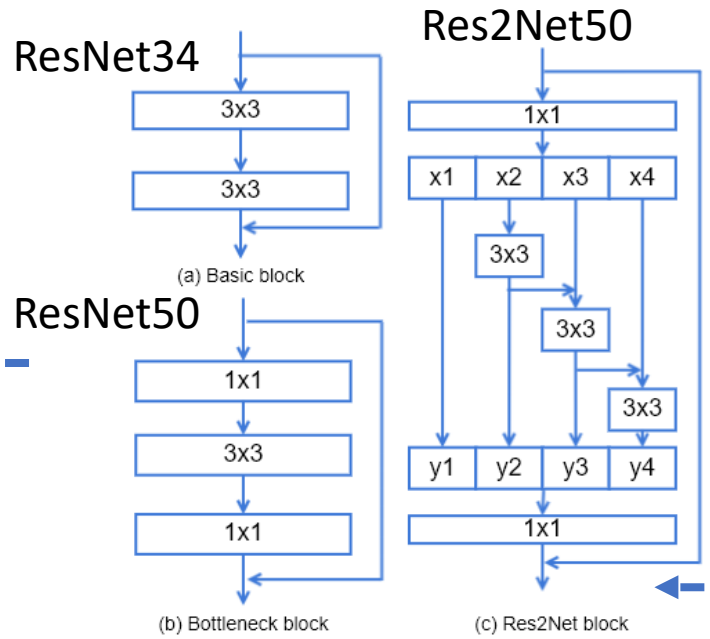
	Receptive Field (secs.)
ResNet34	2.4
ResNet50	1.2
Res2Net50 scale=4	3.6
Res2Net50 sale=8	8.3

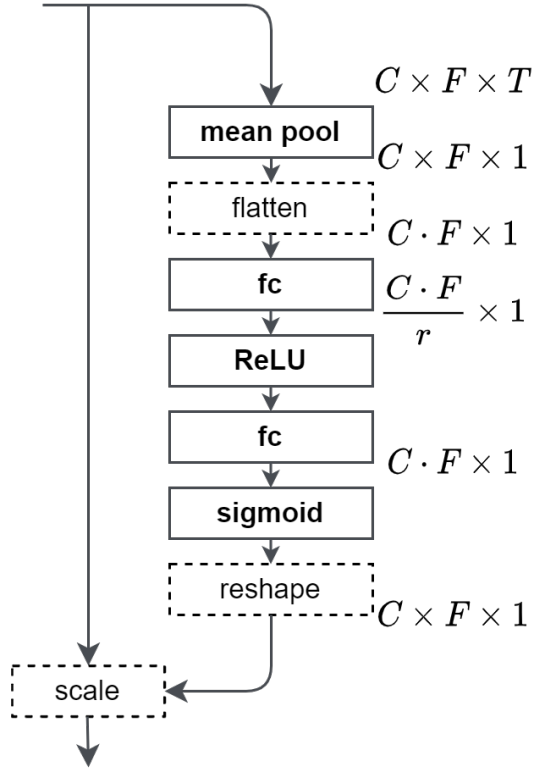
Gao, Shang-Hua, et al. "Res2net: A new multi-scale backbone architecture." *IEEE transactions on pattern analysis and machine intelligence* 43.2 (2019): 652-662.

Squeeze-Excitation ResNet



ResNet Scheme



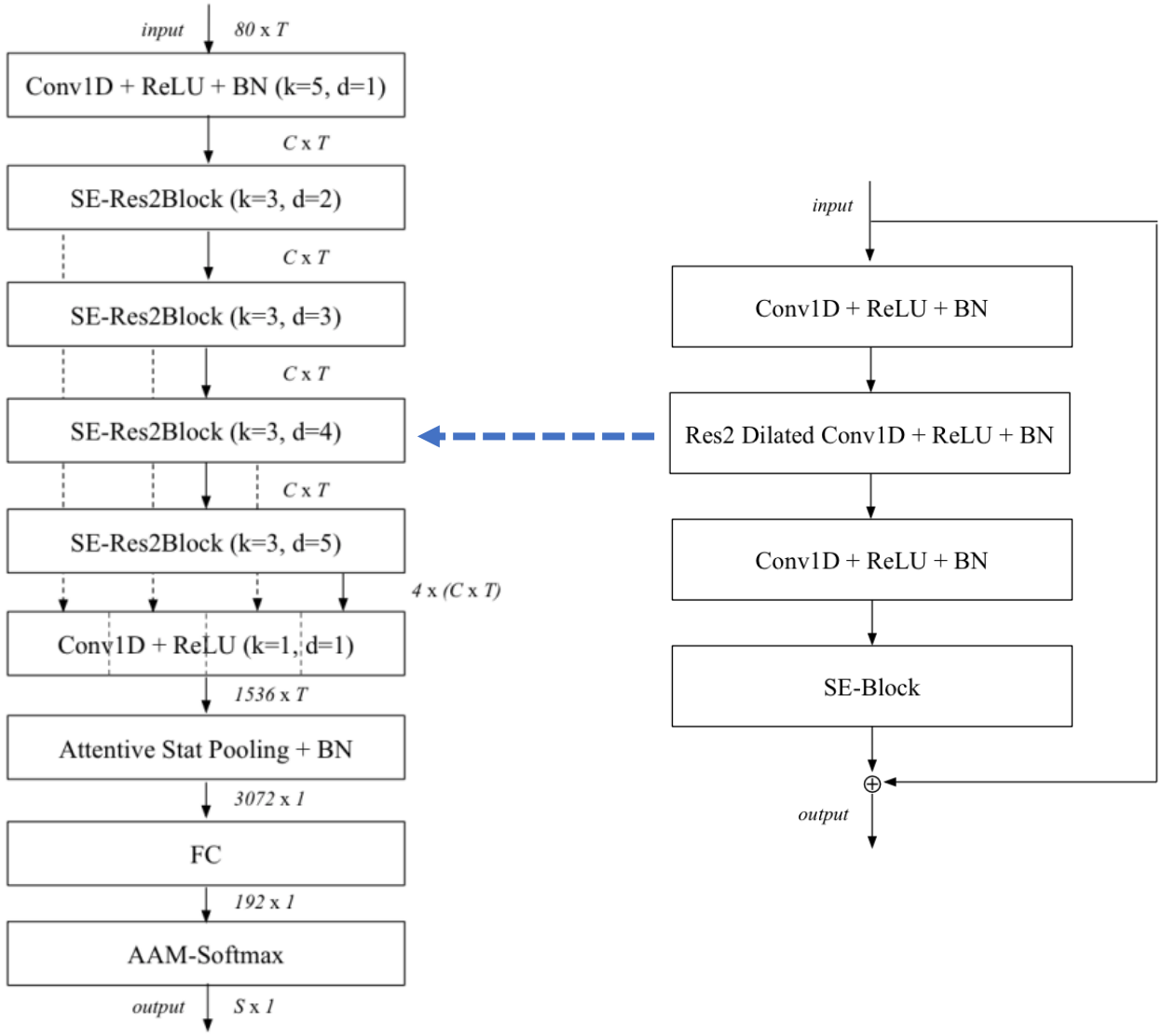


Temporal Squeeze-Excitation

- Pooling only on Time dimension
- Applies different scaling at each channel and frequency dimension

Hu, Jie, Li Shen, and Gang Sun. "Squeeze-and-excitation networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

ECAPA-TDNN



- ECAPA-TDNN:
  - Res2Net 1d
  - Squeeze-Excitation
  - Dilated convolutions in the bottleneck layer to increase receptive field
  - No downsamplings feature maps

N.Net.	Num. layers	Layer dim.	Receptive Field (secs.)
Small	3	512	0.23
Large	4	2048	0.33

Neural Embeddings Takeaways

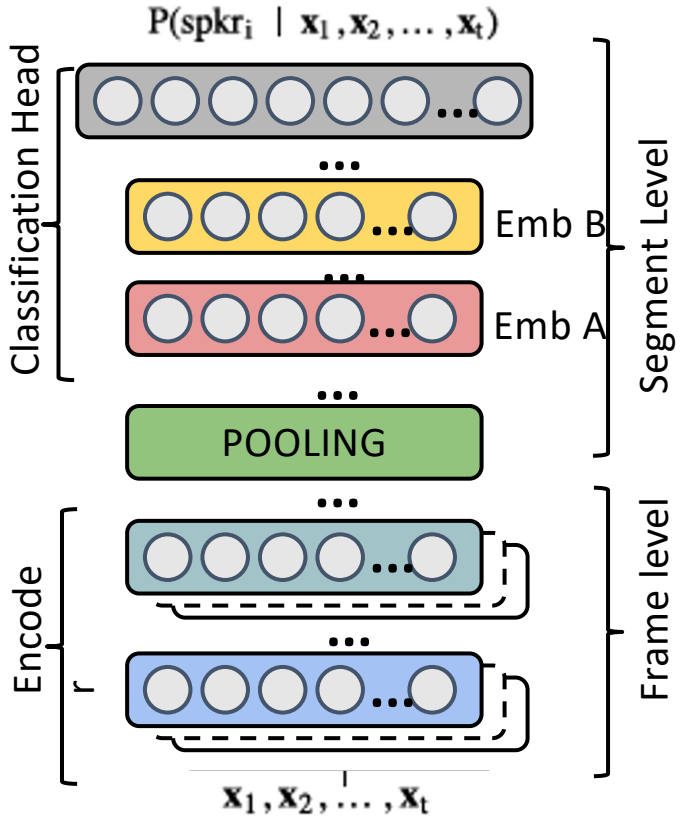
CENTER FOR LANGUAGE AND SPEECH PROCESSING

- SOTA approaches based on the x-Vector scheme:
  - Categorical Cross Entropy
- Architectures based on 1D, 2D Convolutions and Transformers.
- Best Architectures:
  - Res2Net with 2D Convolutions
  - ECAPA-TDNN: Res2Net with 1D Dilated Convolutions.

Pooling Mechanisms

Mean and Statistics Pooling

- Global Average Pooling
  - Mean of Encoder representations along time dimension.
  - For 1D Encoders: (B, C, T) -> (B, C)
  - For 2D Encoders: (B, C, F, T) -> (B, C x F, T) -> (B, C x F)
- Global Statistics Pooling
  - Concatenate:
    - Mean along time dimension.
    - Standard Deviation along time dimension.
  - For 1D Encoders: (B, C, T) -> (B, 2 x C)
  - For 2D Encoders: (B, C, F, T) -> (B, C x F, T) -> (B, 2 x C x F)



Snyder, David, et al. "Deep Neural Network Embeddings for Text-Independent Speaker Verification." *Interspeech*. 2017.

Attentive Statistics Pooling

- Statistics Pooling with different weight for each frame.
  - More weight for most important frames
  - Weight calculus:
    - Weights sum up to one in time dimension
    - f(x) = tanh(x)

$$e_t = \boldsymbol{v}^T f(\boldsymbol{W}\boldsymbol{h}_t + \boldsymbol{b}) + k,$$

$$\alpha_t = \frac{\exp(e_t)}{\sum_\tau^T \exp(e_\tau)}.$$

  - Weighted mean and std. dev:

$$\tilde{\boldsymbol{\mu}} = \sum_t^T \alpha_t \boldsymbol{h}_t.$$

$$\tilde{\boldsymbol{\sigma}} = \sqrt{\sum_t^T \alpha_t \boldsymbol{h}_t \odot \boldsymbol{h}_t - \tilde{\boldsymbol{\mu}} \odot \tilde{\boldsymbol{\mu}}},$$

Okabe, Koji, Takafumi Koshinaka, and Koichi Shinoda. "Attentive statistics pooling for deep speaker embedding." *arXiv preprint arXiv:1803.10963* (2018).

Multi-Head Attention

- Attentive Statistics Pooling with *H* heads
  - Each heads look at different types of frames.
  - Computes a different set of weights per head.
    - The weights of each head sum up to 1 in the time dimension.

$$\mathbf{e}_{h,t} = \mathbf{v}_h^{\mathrm{T}} f(\mathbf{W}\mathbf{h}_t + \mathbf{b}) + k_h \qquad \alpha_{h,t} = \frac{\exp(\mathbf{e}_{h,t})}{\sum_t^T \exp(\mathbf{e}_{h,t})}$$

  - Computes weighted statistics per head

$$\mu_h = \sum_t^T \alpha_{h,t}\mathbf{h}_t \qquad \sigma_h = \sqrt{\sum_t^T \alpha_{h,t}\mathbf{h}_t \circ \mathbf{h}_t - \mu_h \circ \mu_h}$$

  - Concatenate mean and std. dev of all heads.
    - Feature Maps: (B, C, T) -> (B, 2 x C x H)

Pooling Mechanism Take Aways

- Summarizes Sequence of Encoder representations into single vector.

- Best techniques uses some form of attention.

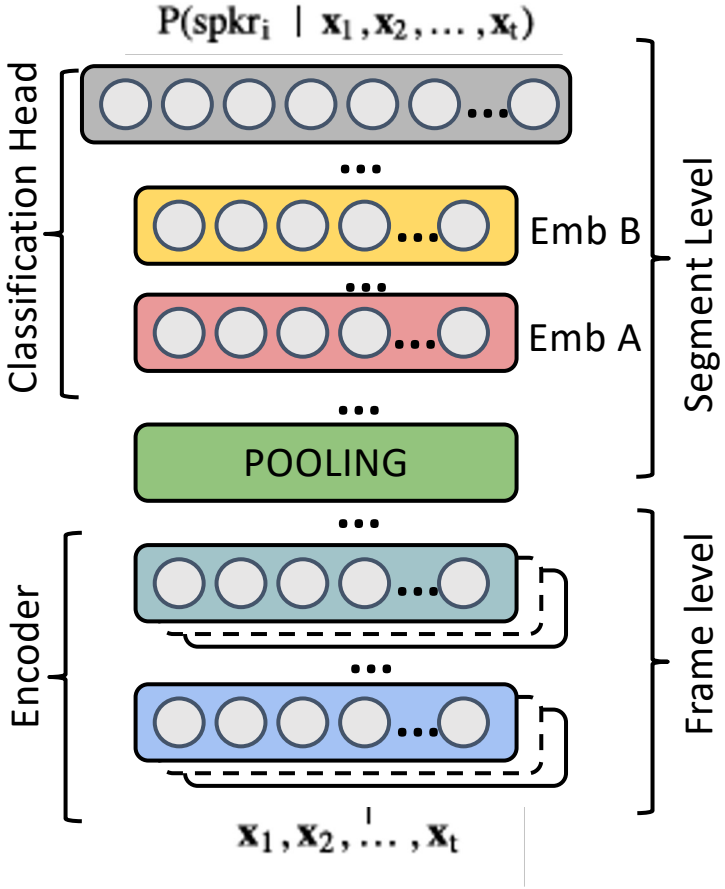
Loss Functions

Multi-class Cross-Entropy

- Categorical Cross-Entropy is the basic loss function for speaker embeddings.

$$L_i = -\log\left(\frac{e^{\boldsymbol{W}_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_j e^{\boldsymbol{W}_j^T \boldsymbol{x}_i + b_j}}\right)$$

  - W is the reference vector for speaker i in the training set.
  - Compares h versus all the training speakers



Additive Margin Softmax (AMSoftmax, CosFace)

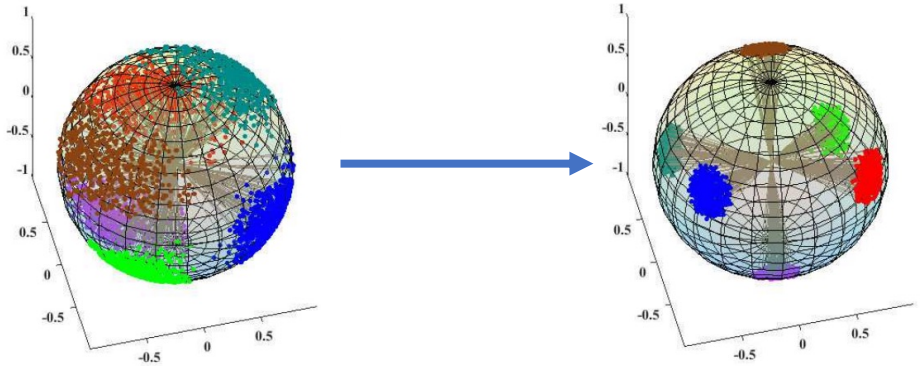
CENTER FOR LANGUAGE AND SPEECH PROCESSING

Softmax Cross-Entropy

$$L_i = -\log\left(\frac{e^{\boldsymbol{W}_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_j e^{\boldsymbol{W}_j^T \boldsymbol{x}_i + b_j}}\right) \xrightarrow[\|x_i\| = 1, \forall i]{\|\boldsymbol{W}_j\| = 1, \forall j} L_{lmc} = \frac{1}{N}\sum_i -\log\frac{e^{s(\cos(\theta_{y_i,i}) - m)}}{e^{s(\cos(\theta_{y_i,i}) - m)} + \sum_{j \neq y_i} e^{s\cos(\theta_{j,i})}}$$

$$= -\log\left(\frac{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\cos(\theta_{y_i,i}) + b_{y_i}}}{\sum_j e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|\cos(\theta_{j,i}) + b_j}}\right)$$

- Score for each class is cosine of the angle between input and the class vector.
  - The norm of the input vector, is normalized to 1.
  - Multiplies by scale value s ~ 30 to sharpen the posterior probabilities
- Subtract Margin m from the score of the true class
  - Creates Stronger Gradients for Correctly classified classes
  - Increase separation between Embedding of Different Classes

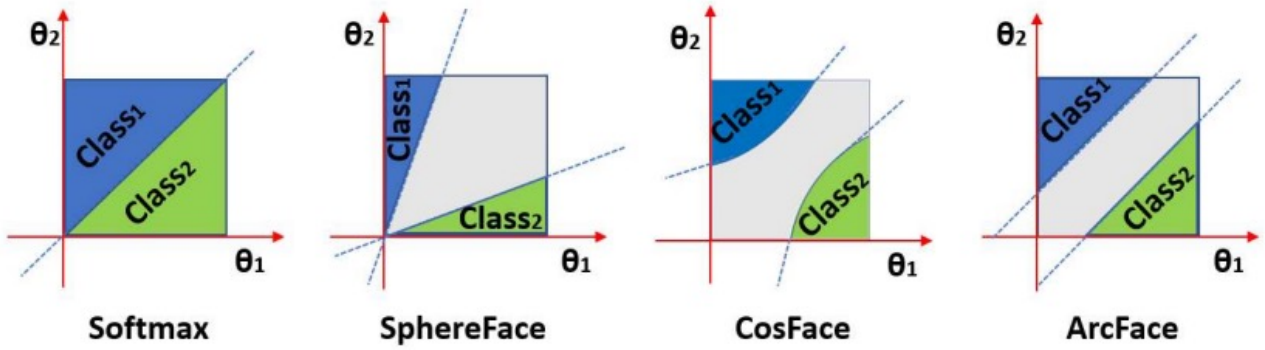


Wang, Hao, et al. "Cosface: Large margin cosine loss for deep face recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

Additive Angular Margin Softmax (Arc-Face)

$$L_3 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}$$

- Decision Margins:



θ_i Angle between Embedding and Reference Vector for class i

		EER(%)	minDCF08	minDCF10
Kaldi [2]		3.10	0.0169	0.4977
Softmax		2.34	0.0122	0.3754
Modified Softmax		2.62	0.0131	0.4146
ASoftmax	$m_1 = 2$	2.18	0.0119	0.3791
	$m_1 = 4$	2.15	0.0113	0.3108
ArcSoftmax	$m_2 = 0.20$	2.14	0.0119	0.3610
	$m_2 = 0.25$	2.03	0.0120	0.4010
	$m_2 = 0.30$	2.12	0.0115	0.3138
	$m_2 = 0.35$	2.23	0.0123	0.3622
AMSoftmax	$m_3 = 0.15$	2.13	0.0113	0.3707
	$m_3 = 0.20$	2.04	0.0111	0.2922
	$m_3 = 0.25$	2.15	0.0119	0.3559
	$m_3 = 0.30$	2.18	0.0115	0.3152

Deng, Jiankang, et al. "Arcface: Additive angular margin loss for deep face recognition." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019.

Liu, Yi, Liang He, and Jia Liu. "Large Margin Softmax Loss for Speaker Verification}}." *Proc. Interspeech 2019* (2019): 2873-2877.

Losses Takeaways

- Large Margin Losses Boost Performance in Speaker and Face Recognition

- AM-Softmax and AAM-Softmax Best

Back-end

Metric (Back-end)

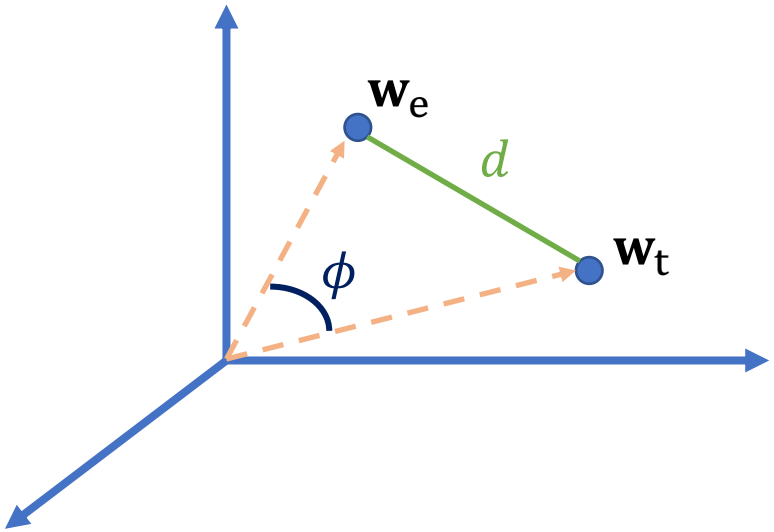
- Assume that:
  - $\mathbf{w}_e$ spk. embedding from enrollment utterance of speaker **X**
  - $\mathbf{w}_t$ spk. embedding from test utterance of person that claims to be speaker **X**

- The Metric compares enrollment and test embeddings $\mathbf{w}_e$, $\mathbf{w}_t$

- Cosine scoring:

$$s = \cos(\phi) = \frac{\mathbf{w}_e^T \mathbf{w}_t}{\|\mathbf{w}_e\|_2 \|\mathbf{w}_t\|_2}$$

  - Simplest Metric
  - Large Margin Losses optimize a cosine score metric
  - Works well in many tasks:
    - VoxCeleb
    - Eval Data is in-domain



Linear Discriminant Analysis

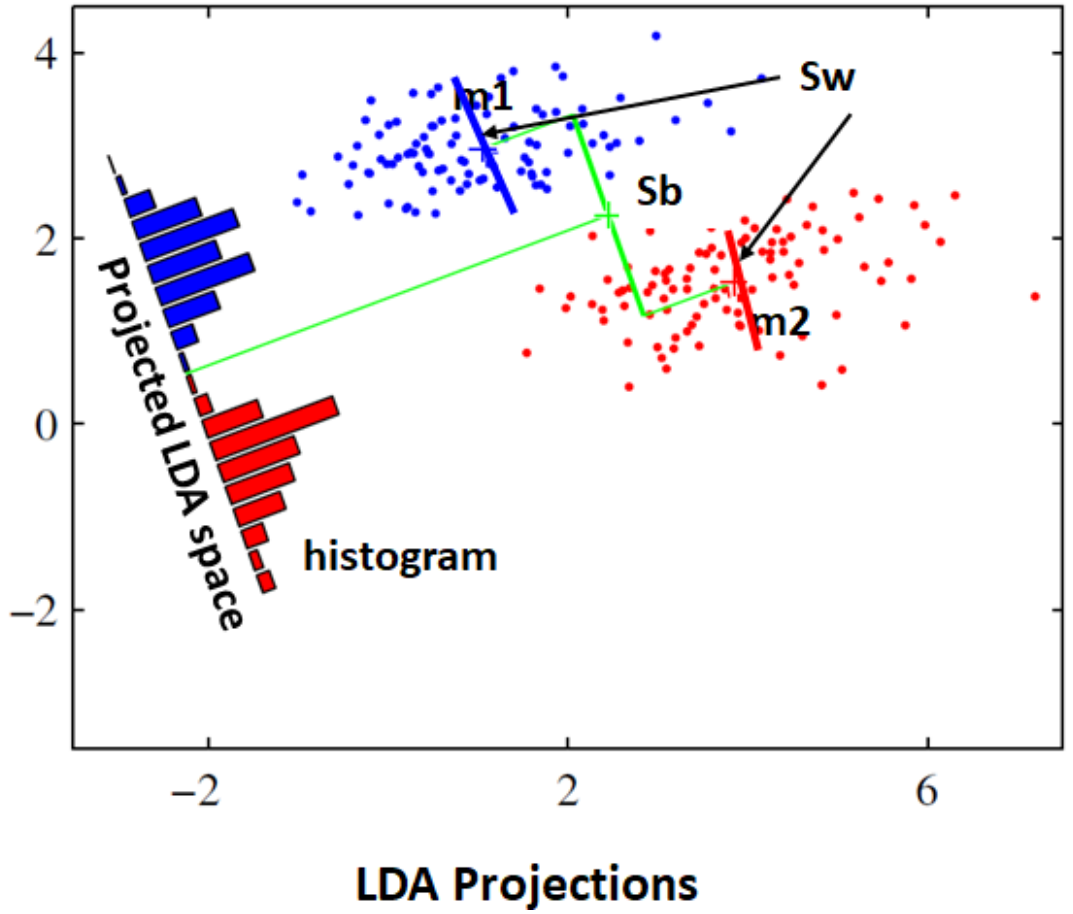
- LDA projects embeddings into a new vector space:
  - Maximize the separation between classes.
  - Minimizes the covarianze within each class
- Between/Within-class covariances:

$$\mathbf{S}_b = \frac{1}{M}\sum_{i=1}^{M}(\mu_i - \mu)(\mu_i - \mu)^{\mathrm{T}}$$

$$\mathbf{S}_w = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_i}\sum_{j=1}^{N_i}(\mathbf{x}_{ij} - \mu_i)(\mathbf{x}_{ij} - \mu_i)^{\mathrm{T}}$$

- Solve the generalized eigenvalue problem:

$$\max_{\mathbf{v}:\,\|\mathbf{v}\|=1} \frac{\mathbf{v}^{\mathrm{T}}\mathbf{S}_b\mathbf{v}}{\mathbf{v}^{\mathrm{T}}\mathbf{S}_w\mathbf{v}} \quad \rightarrow \quad \mathbf{S}_b\mathbf{v} = \lambda\mathbf{S}_w\mathbf{v}$$



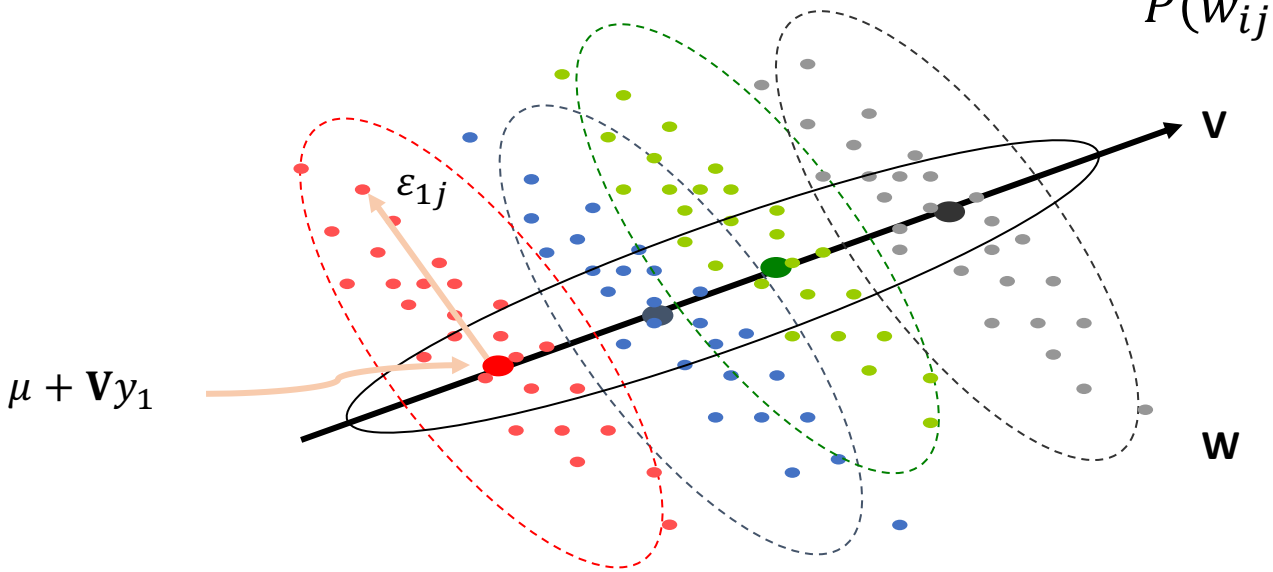
Probabilistic Linear discriminant Analysis (PLDA)

- Probabilistic version of LDA
- x-vector *j* of class *i* is decomposed as a sum of several terms

$$w_{ij} = \mu + \mathbf{V}y_i + \epsilon_{ij}$$

  - μ is the class-independent mean of all the i-vectors
  - **V** is low rank matrix defining the inter-class variability space
  - $y_i \sim \mathrm{N}(0, \mathrm{I})$ are the coordinates of the speaker in the space defined by **V**
  - $\epsilon_{ij} \sim \mathrm{N}(0, \mathbf{W})$ where W is the intra-class covariance.

$$P(w_{ij}|y_i) = N(w_{ij}|\mu + \mathbf{V}y_i, \mathbf{W})$$



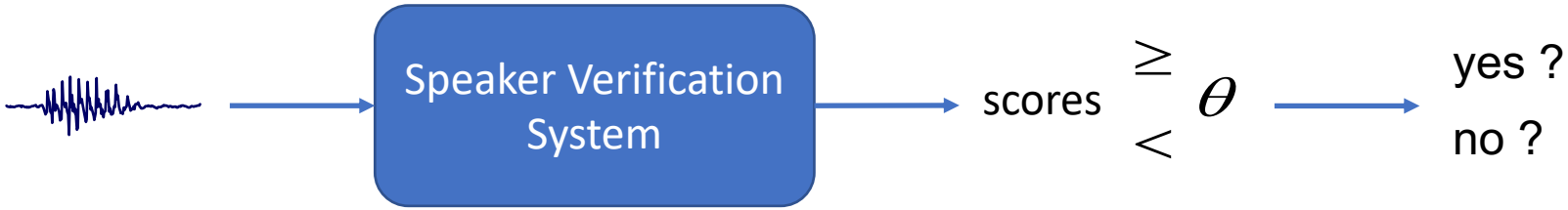
PLDA Evaluation

- Evaluates a Log-likelihood ratio test between two hypothesis:
  - Probability for enrollment and test vectors were generated by the same speaker
    - Have the same y
  - Probability for enrollment and test vectors were generated by different speakers
    - Have the same y

$$\text{LLR} = \log \frac{P(w_1, w_2|same)}{P(w_1, w_2|diff)} = \log \frac{\int P(w_1|y)P(w_2|y)P(y)dy}{\int P(w_1|y)P(y)dy \int P(w_2|y)P(y)dy} = \log \frac{N\left(\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \middle| \begin{bmatrix} \mu \\ \mu \end{bmatrix}, \begin{bmatrix} \mathbf{V}\mathbf{V}^T + \mathbf{W} & \mathbf{V}\mathbf{V}^T \\ \mathbf{V}\mathbf{V}^T & \mathbf{V}\mathbf{V}^T + \mathbf{W} \end{bmatrix}\right)}{N\left(w_1 \middle| \mu, \mathbf{V}\mathbf{V}^T + \mathbf{W}\right) N\left(w_2 \middle| \mu, \mathbf{V}\mathbf{V}^T + \mathbf{W}\right)}$$

S. J. D. Prince and J. H. Elder, "Probabilistic Linear Discriminant Analysis for Inferences About Identity," *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1-8, doi: 10.1109/ICCV.2007.4409052.

Score Calibration



- How do we choose the decision threshold?
- Posterior probability for target trial:

$$P(\text{Tar}|\mathbf{x}_1,\mathbf{x}_2) = \frac{P_{\text{Tar}}P(\mathbf{x}_1,\mathbf{x}_2|\text{Tar})}{P_{\text{Tar}}P(\mathbf{x}_1,\mathbf{x}_2|\text{Tar}) + (1-P_{\text{Tar}})P(\mathbf{x}_1,\mathbf{x}_2|\text{NonTar})} =$$

$$\frac{1}{1+\exp\left(-\log\frac{P_{\text{Tar}}P(\mathbf{x}_1,\mathbf{x}_2|\text{Tar})}{(1-P_{\text{Tar}})P(\mathbf{x}_1,\mathbf{x}_2|\text{NonTar})}\right)} = \text{sigmoid}\left(\log\frac{P_{\text{Tar}}P(\mathbf{x}_1,\mathbf{x}_2|\text{Tar})}{(1-P_{\text{Tar}})P(\mathbf{x}_1,\mathbf{x}_2|\text{NonTar})}\right) \geq 0.5$$

  - Prior probability of finding a target trial P_{Tar}
- Derive the threshold for the log-likelihood ratio:

$$\log\frac{P(\mathbf{x}_1,\mathbf{x}_2|\text{Tar})}{P(\mathbf{x}_1,\mathbf{x}_2|\text{NonTar})} \geq -\log\frac{P_{\text{Tar}}}{(1-P_{\text{Tar}})} = -\text{logit}(P_{\text{Tar}}) = \text{thr}$$

Score Calibration

- Back-end Scores s need calibration
  - Cosine scoring does not produce a log-likelihood ratio.
  - PLDA does produce a log-likelihood ratio but not always is well calibrated.
- Linear calibration: $\mathrm{LLR} = a\ s + b$
- Trained with Weighted binary cross-entropy:

$$L = \frac{P_{\mathrm{Tar}}}{N_{\mathrm{Tar}}} \sum_{D \in \mathrm{Tar}} \log P(\mathrm{Tar}|D) + \frac{1 - P_{\mathrm{Tar}}}{N_{\mathrm{Non}}} \sum_{D \in \mathrm{Non}} \log P(\mathrm{Non}|D) =$$

$$\frac{P_{\mathrm{Tar}}}{N_{\mathrm{Tar}}} \sum_{s \in \mathrm{Tar}} \log(\mathrm{sigmoid}(as + b + \mathrm{logit}P_{\mathrm{Tar}})) + \frac{1 - P_{\mathrm{Tar}}}{N_{\mathrm{Non}}} \sum_{s \in \mathrm{Non}} \log(1 - \mathrm{sigmoid}(as + b + \mathrm{logit}P_{\mathrm{Tar}}))$$

  - Train on a held-out set of target/non-trials

Brümmer, Niko, and Johan Du Preez. "Application-independent evaluation of speaker detection." *Computer Speech & Language* 20.2-3 (2006): 230-275.

Back-end Pipeline

- The common back-end pipeline may involve all the blocks:

Back-end Takeaways

- Cosine scoring works well for easy tasks
  - Training and Test data are from the same domain

- PLDA is better when the embedding training data is out-of-domain
  - Some in-domain data can be used to train the PLDA

- Calibration allow us to set the threshold given a target prior probability.

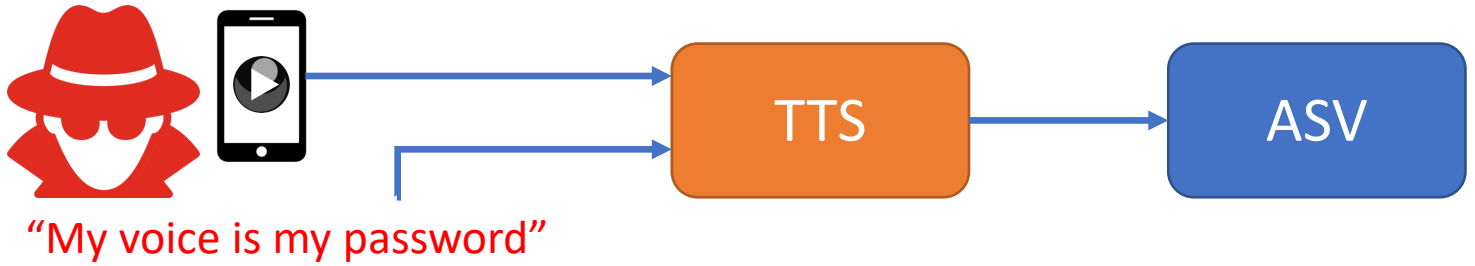
Other Topics in Speaker Recognition

Spoofing Attacks

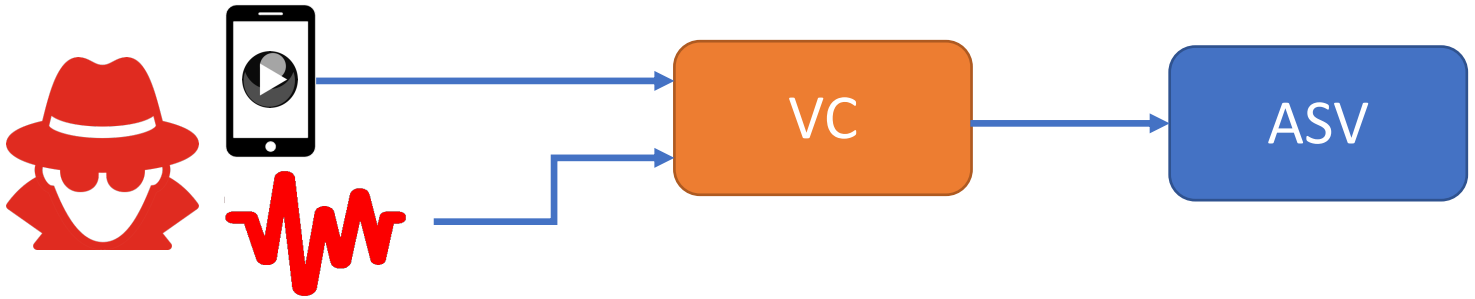
- Impostor tries to Impersonate a legitimate user
- Acquire Victim's Voice Sample

- Replay Attack

- Text-to-Speech

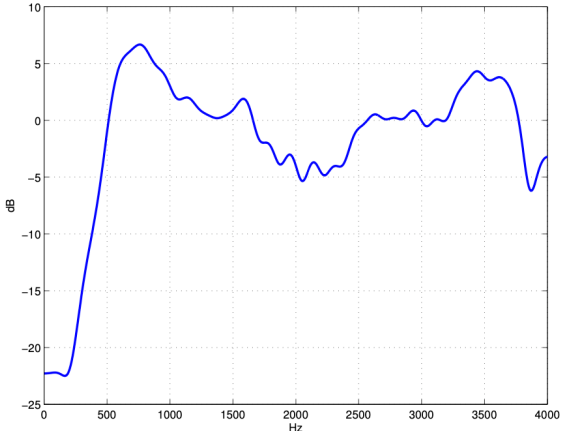


- Voice Conversion



Spoofing Detectors

- Physical access:
  - Spoofing audio played over air channel
  - Focus: detect loudspeaker and far-field artifacts

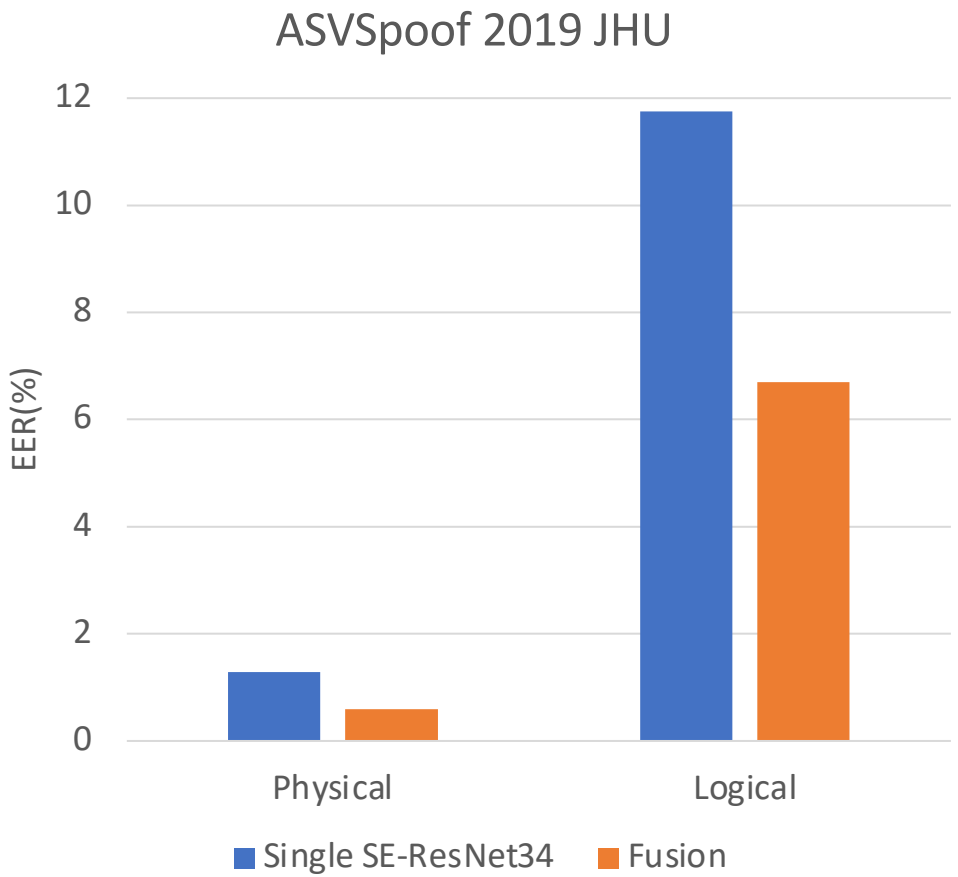


Loudspeaker frequency response

- Logical Access:
  - Signal injected in ASV digitally (no air-channel)
  - Focus: detect TTS and VC vocoder artifacts
- From pure signal processing to Deep learning detectors

JHU System for ASVSpoof2019 challenge is fusion of

- ResNets, Squeeze-Excitation ResNet and Dilated ResNet

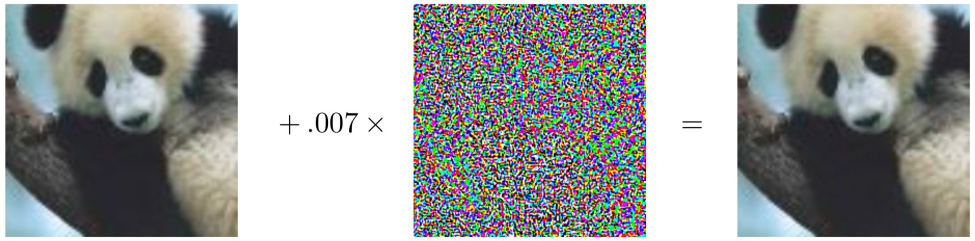


Villalba, J., Lleida, E.. Preventing Replay Attacks on Speaker Verification Systems. In *IEEE ICCST 2011*

Lai, C.-I., Chen, N., Villalba, J., Dehak, N. ASSERT: Anti-Spoofing with Squeeze-Excitation and Residual neTworks. In*, INTERSPEECH 2019*

Adversarial Attacks

- Adversarial Attacks add a small perturbation to the signal, which is imperceptible for humans but changes the output of the ML systems



Panda 60% Gibbon 99%

- Few studies in speaker verification (most works are in image domain)

Types of Adversarial Attacks

- Goal of the attacker:
  - ***Impersonation***
  - ***Evasion***

- Knowledge of the attacker:
  - White-box:
    - Attacker has full knowledge of the victim system, architecture and parameters
  - Black-box:
    - Attacker doesn't have access to the victim model

- Optimization method used to obtain the adversarial sample:
  - Fast gradient sign method (FGSM), Iterative FGSM, Carlini-Wagner

FGSM Attacks

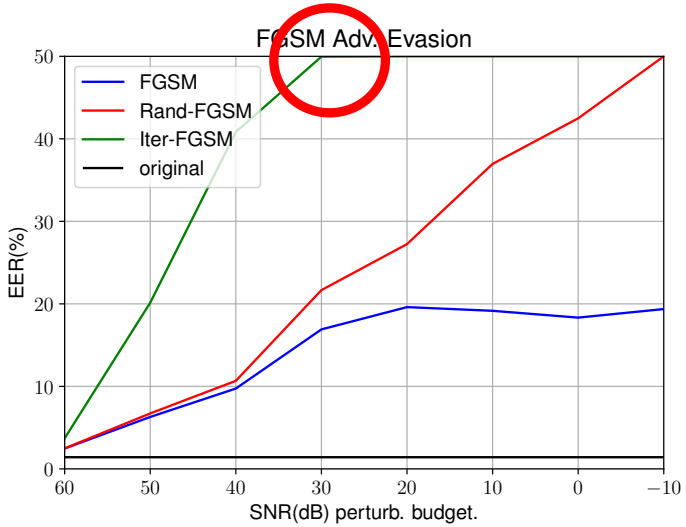
- Fast Gradient Sign Method (FGSM) [Goodfellow et al. 2015]

  - J is the objective function (Binary cross-entropy for the speaker verification task)
  - ε is equal to the perturbation L_∞ norm
  - Fast method (1 forward/backward pass), not optimal
- Iter-FGSM:
  - Multiple FGSM iterations, stronger attacks
- Others:
  - Carlini-Wagner-L2/Linf/L0, PGD-Linf/L2/L1, etc

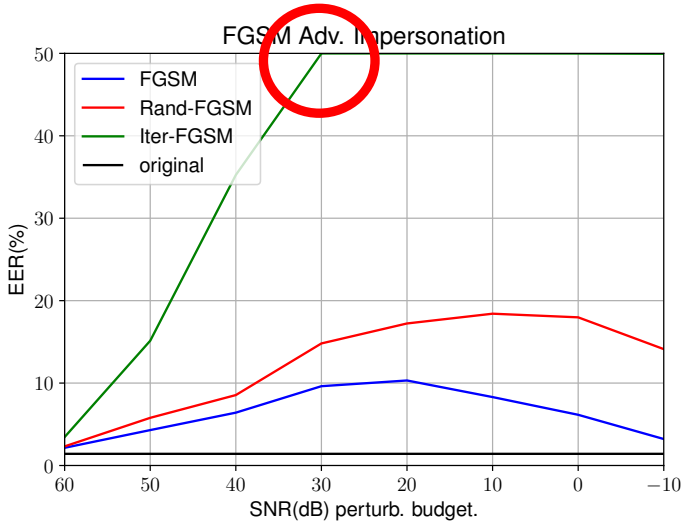
Goodfellow et al. “Explaining and harnessing adversarial examples”, ICLR 2015

Results White-box Attacks

Evasion



Impersonation



- Benchmark SOTA Speaker Verification against adv. attacks
- EER(%) vs SNR
  - $\text{SNR} = 10 \log_{10} \frac{P_{\text{benign speech}}}{P_{\text{adv. perturbation}}}$ (dB)

- Iter-FGSM method achieves 50 % EER for SNR>30dB

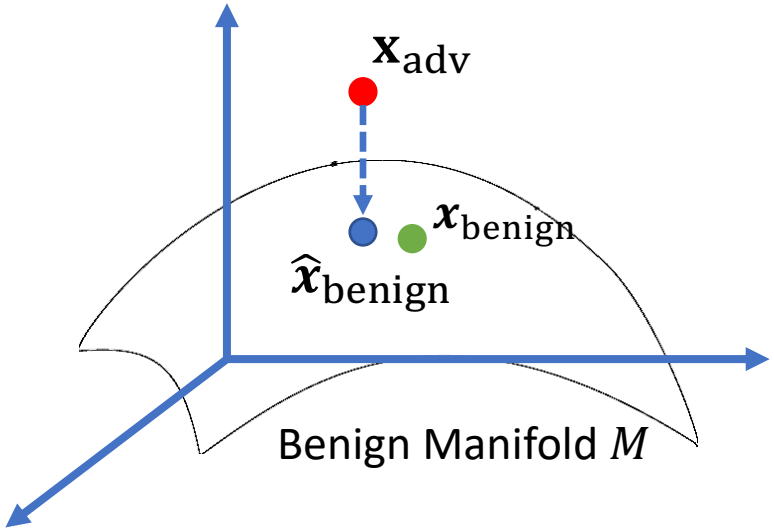
Villalba, J., Zhang, Y., Dehak, N. x-Vectors Meet Adversarial Attacks : Benchmarking Adversarial Robustness in Speaker Verification. In *Interspeech 2020*.

Generative Models as Defenses

- Investigating how to use Generative Models as Defenses

- Generative model defines the manifold of benign signals
  - Trained on clean data
  - VAE, GAN

- Assumption:
  - Adversarial signals are outside of the manifold

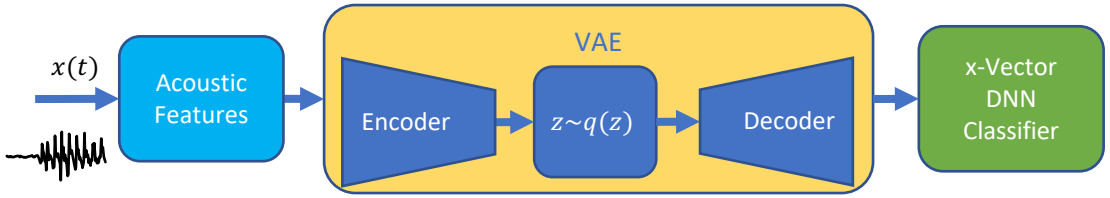


- Project adversarial sample into the benign manifold

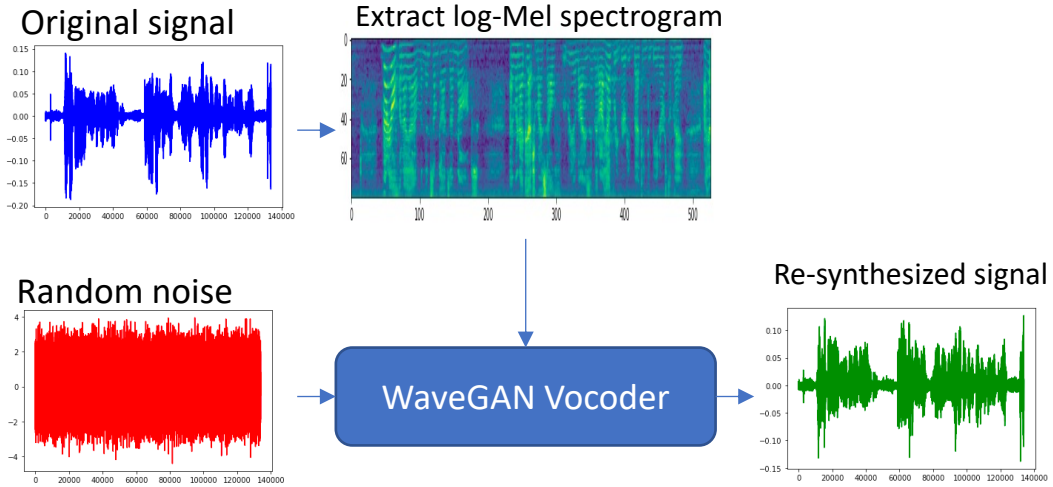
Joshi, S., Villalba, J., Zelasko, P., Moro, L., Dehak, N. Adversarial Attacks and Defenses for Speaker Identification Systems, Submitted to IEEE Transactions of Information Forensics and Security

VAE/WaveGAN Defense

- Variational Autoencoder Defense
  - Applied in log-Mel Spectrogram domain

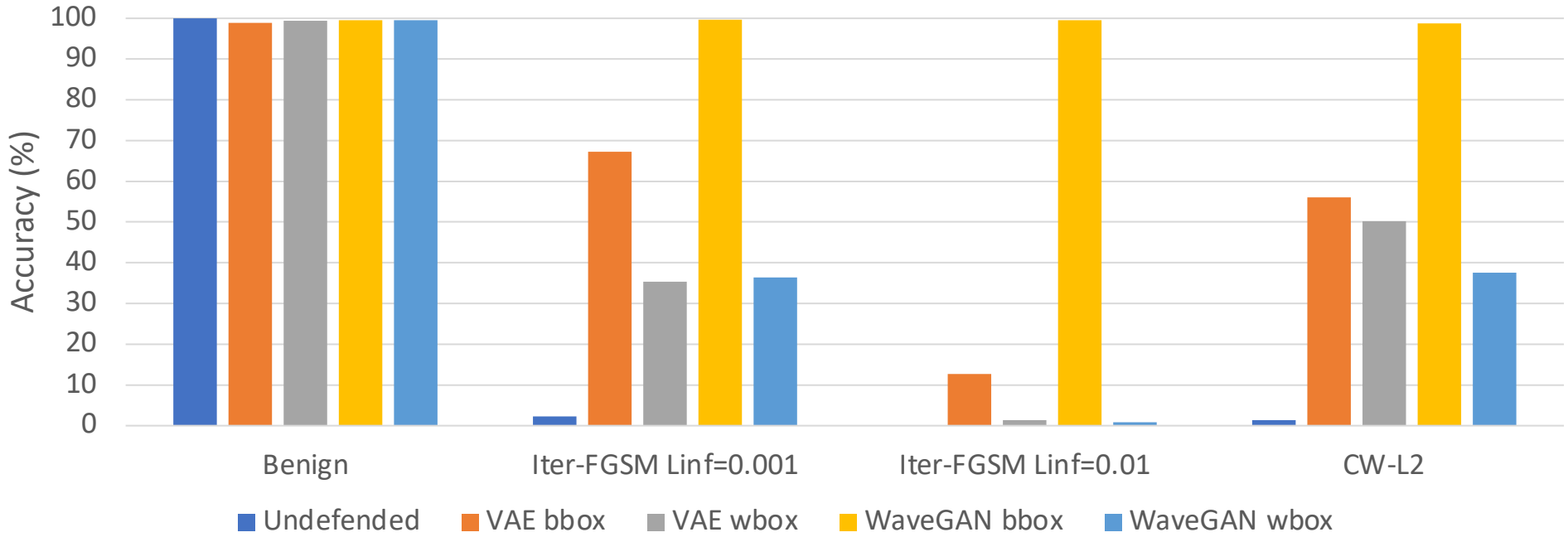


- ParallelWaveGAN vocoder
  - Re-Synthetize Speech Waveform given the Mel-Spectrogram
  - Trained on mix of adversarial and multi-scale short-time Fourier transform losses



Defenses Result

- Evaluate on a 40 speaker's Speaker Identification task based on LibriSpeech
- Two cases:
  - Black-box Defense: The adversary doesn't know that there is a defense
  - White-box Defense: The adversary can adapt to the defense
- WaveGAN performs very in the black-box case, need to improve in the white-box case




ARE THERE ANY QUESTIONS?