Improving Fairness in Speaker Recognition and Speech Recognition

Andreas Stolcke, *Amazon Alexa AI*

Georgia Tech, March 31, 2023

amazon | science

Overview

Introduction to group fairness

ASR fairness – Fairness cohort discovery and mitigation

ASR fairness – Geographic disparity reduction

Speaker verification fairness – Group-adapted fusion

Speaker verification fairness – Adversarial reweighting

ASR fairness – Synthetic data for robustness to stuttered speech

Summary and Conclusions

What is Group Fairness?

- Bing Chat (GPT-4) says:
  - "Group fairness is a concept in machine learning that measures how a group of individuals with certain **protected attributes** (like gender or race) is impacted differently from other groups. It aims to achieve the same outcomes across different demographics or a set of protected population classes"
- But what about "non-protected" groups/attributes?
  - For example: age, regional accent, tenure with a voice assistant
  - Goal is to make speech-enabled AI systems perform about equally well for all speakers/attributes

$$P(f(x) \geq \theta \mid A(x)) \approx P(f(x) \geq \theta | \neg A(x)),$$

for a performance metric $f(x)$, threshold θ , and all attributes $A(x)$ we care about.

  - Attributes are often not available due to data privacy, so impossible to verify in general!

In this talk

- Focus on algorithmic approaches that reduce disparities in performance
  - For speech recognition (ASR)
  - For speaker recognition (speaker verification - SV)
- Metrics will be ad-hoc, depending on task
  - Absolute or relative differences in metric between groups
  - Word error rate (WER) for ASR
  - Equal error rate (EER) for speaker verification

Fairness and representation

- Empirically, group underperformance in ML systems is typically associated with underrepresentation in the training set
  - Training objective is to minimize loss over the entire dataset
  - It “pays” more to minimize loss for the majority
- Example:

  If nonnative speakers are a minority in the data, we expect ASR models to perform poorly for them
- Remedy:

  Increase the underrepresented group’s aggregated contribution to the loss function

Mitigation: Improving representation

- “Target group” = group that is underrepresented / seeing sub-par performance
- How to increase representation of the target group in training loss?
- Many different approaches:
  - Give extra weight to target group samples in the loss computation
  - Oversample the target group
  - Adapt / fine-tune model on the target group
  - Use group-specific models (and combine them)
  - Use modified loss function that penalizes disparities
  - Fabricate data for the target group

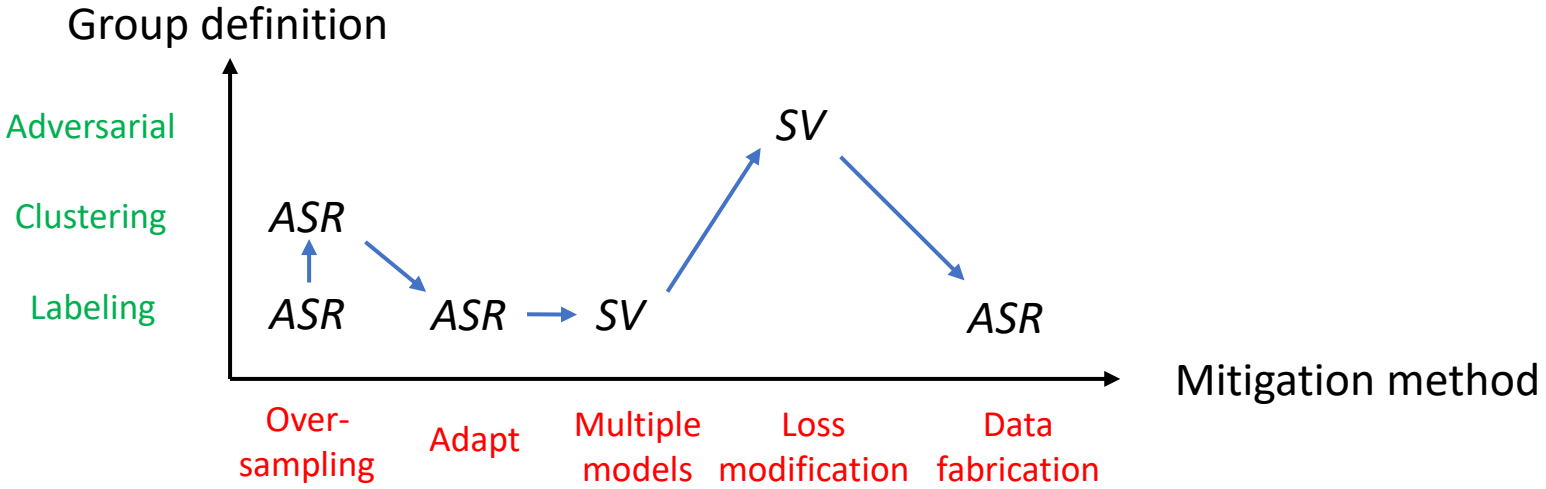
How to define/identify groups?

- Again, several approaches:
  - By pre-existing categories, e.g., demographic labels, metadata, …
    - Proxy labels (e.g., ZIP codes for demographics)
  - By automatic discovery / clustering
  - By an adversarial model (implicitly)

- Getting labels is a challenge in itself
  - Especially for protected / demographic attributes

- Methods that require no group labels in training are preferred, other things being equal
  - We then only need labels on the test data for evaluation

Fairness roadmap



- We will visit all points along both axes, but not map out the entire space!
- Along the way, we have some detours, e.g., to look at human performance disparities

Fairness Cohort Discovery and Mitigation

Pranav Dheram, **Murugesan Ramakrishnan**, A. Raju, I-F. Chen, B. King, K. Powell, M. Saboowala, K. Shetty, A. Stolcke, Toward Fairness in Speech Recognition: Discovery and mitigation of performance disparities, *Proc. Interspeech*, 2022

Human and Machine Performance Disparities

- Performance Disparity
  - Measure across two cohorts in this work: termed **bottom** and **top** performing cohort
- Machine Performance Disparity
  - ASR model confidence score: probability of the model output sequence being correct
  - Word error rate (WER) gap

$$WER\text{-}gap = \frac{WER_{bottom} - WER_{top}}{WER_{top}}$$

Human and Machine Performance Disparities

- Human Performance Disparity
  - Human transcription errors impact WER
  - Transcription process: 3 blind-pass + 1 adjudicator
  - Measure consensus between 3 independent transcribers
    - Inter-annotator agreement rates (IAA): 1-1-1 (all disagree) / 2-1 (two agree) / 3-0 (all agree)
    - Pairwise disagreement rates (PDR) computed from IAA

$$PDR = Percentage_{1\text{-}1\text{-}1} \cdot \frac{3}{3} + Percentage_{2\text{-}1} \cdot \frac{2}{3}$$

Manual cohort discovery: geodemographic

- Geodemographic cohorts – ZIP codes as a proxy label
  - Analyzed geodemographic characteristics: ZIP codes and US census data
  - Identified low ASR accuracy cohort:
    - for all census attributes, partition ZIP codes into those with ≥ 75% of population sharing that attribute, versus all other ZIP codes
    - select partition with largest ASR disparity
  - **Bottom cohort:** set of low-accuracy ZIP codes
  - **Top cohort:** all other ZIP codes

Manual cohort discovery: geodemographic

Table 1: Hybrid RNN-HMM ASR model performance disparities

Geolocation-based Cohorts	#ZIPs	#Hrs (K)	ASR Conf. Relative
Overall	41696	4513	Baseline
Bottom	431	48	-11.7%

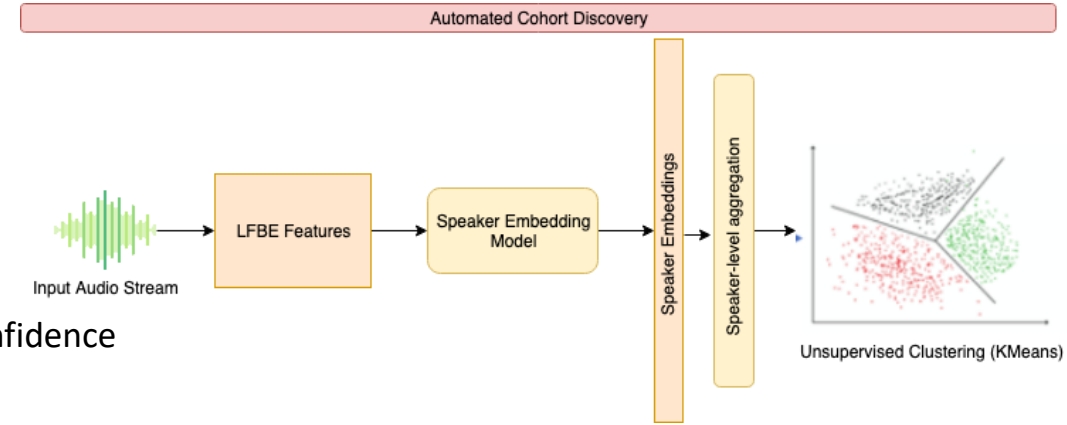
Table 2: Inter-annotator agreement rates of three blind passes

Cohorts	1-1-1 (All 3 disagree)	2-1 (Two agree)	3-0 (All 3 agree)	PDR	Rel. PDR increase in Bottom
Top	16.9%	32.2%	51.0%	27.6%	-
Bottom	26.2%	33.4%	40.5%	37.3%	35.1%

Automated cohort discovery

- **Problems with manual cohort discovery**
  - Not scalable
  - Small cohort sizes
  - Limited resolution

- **Cohort discovery**
  - Clustered speaker embeddings
  - Bottom cohort
    - 10% of clusters with lowest ASR confidence scores



Automated cohort discovery

Table 3: Hybrid RNN-HMM ASR model performance disparities

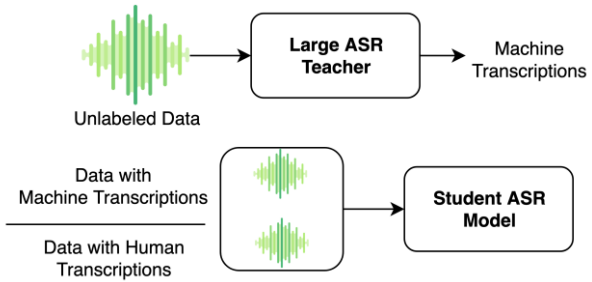
Cohort Discovery	WER-gap (%)	Bottom cohort share (%)
Geodemographic	41.7	0.8
Automatic	65.0	10.0

Table 4: Inter-annotator agreement rates of three blind passes for automatic cohorts

Cohorts	1-1-1 (All 3 disagree)	2-1 (Two agree)	3-0 (All 3 agree)	PDR	Rel. PDR increase in Bottom
Top	11.8%	24.6%	63.6%	20.0%	-
Bottom	16.7%	30.0%	53.5%	26.7%	33.5%

Mitigation: Oversampling bottom cohort data

- Bottom cohort underrepresented in training data
  - 0.8% for geodemographic cohorts
- Solution: Oversample bottom cohort data
- Semi-supervised learning
  - Identify unlabeled bottom cohort data using ZIP codes
  - Obtain machine transcriptions using a teacher model



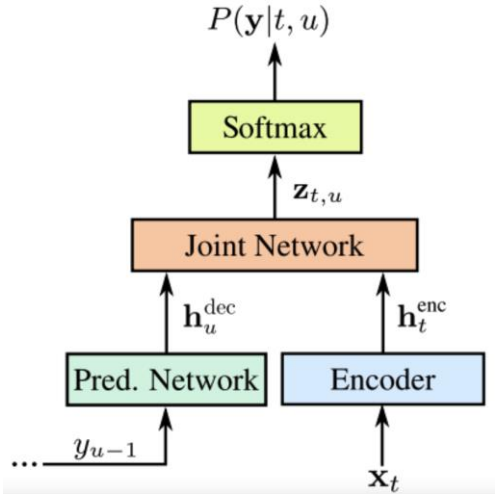
Unsupervised group labeling: Cohort embeddings

- Prior work [1]
  - One-hot accent embedding inputs to ASR improves accented speech accuracy
- Why?
  - One-hot embedding as an adapting bias in first layer
- Extend to cohorts
  - One-hot cohort embedding as an additional input to our ASR model
  - Help the model learn linguistic difference between top and bottom cohort

[1] M. Grace, M. Bastani and E. Weinstein. 2018. Occam's Adaptation: A Comparison of Interpolation of Bases Adaptation Methods for Multi-Dialect Acoustic Modeling with LSTMS. *Proc. IEEE Spoken Language Technology Workshop (SLT)*, pp. 174-181.

Experimental Setup

- Training Data
  - Baseline: 100,000+ hours of de-identified voice-assistant data
- Evaluation Data
  - Bottom cohort: 2,040 utterances
  - Top cohort: 31,199 utterances
- ASR model
  - End-to-end RNNT model architecture
- Metrics
  - $\text{WERR} = \frac{\text{WER}_{baseline} - \text{WER}_{exp}}{\text{WER}_{\text{baseline}}}$ $\quad\quad$ $\text{WER gap} = \frac{\text{WER}_{bottom} - \text{WER}_{top}}{\text{WER}_{top}}$



Results: Mitigating disparities

- Oversampling bottom cohort data improves performance
  - Bottom cohort WER: 5% relative improvement
  - WER-Gap reduced from: 56.3% to 46.2%
- Impact of Cohort embedding
  - Cohort embedding reduces WER gap from 56.3% to 38.5%.

Table 5: Impact of performance disparity mitigation on geodemographic cohorts

Exp.No	Model	Relative WER-Gap (%)	Bottom WERR (%)	Top WERR (%)	% Bottom cohort samples in training
E0	Baseline	56.3	0	0	0.8
E1	E0 + SSL	46.2	5.0	-1.6	9.0
E2	E0 + Cohort embedding	38.5	10.0	-1.6	0.8
E3	E0 + Cohort embedding + SSL	40.0	9.0	-1.6	9.0

Summary

- Cohort discovery
  - Manually identified performance disparities using geodemographic factors
  - Proposed automatic cohort discovery: larger bottom cohort, larger discrepancy
- ZIP-code-based grouping of speakers identifies ASR performance disparities
  - Group with higher ASR error also more difficult for human transcribers (higher disagreement)
- Tested two effective methods to help close ASR performance gap
  - Oversampling data from underperforming cohort
  - Encode cohort membership in model inputs

Reducing Geographic Disparities in ASR

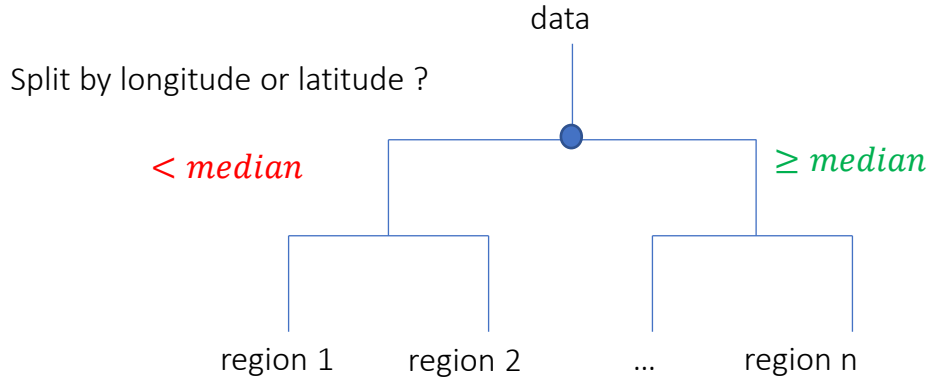
Viet Anh Trinh, P. Ghahremani, B. King, J. Droppo, A. Stolcke, R. Maas, Reducing Geographic Disparities in Automatic Speech Recognition via Elastic Weight Consolidation, *Proc. Interspeech*, 2022

Geolocation for speaker grouping

- ASR performance is affected by geography (e.g., regional accent, socio-economics)
- Instead of ZIP codes and human population attributes, use geolocation directly for grouping
- Given a pretrained ASR model, cluster speakers by geolocation to identify areas of high error rate
- Mitigation:
  - Adapt ASR model to reduce the performance gap against these high error regions
  - Without degrading average performance for all regions
  - Without access to the data of the pretraining stage

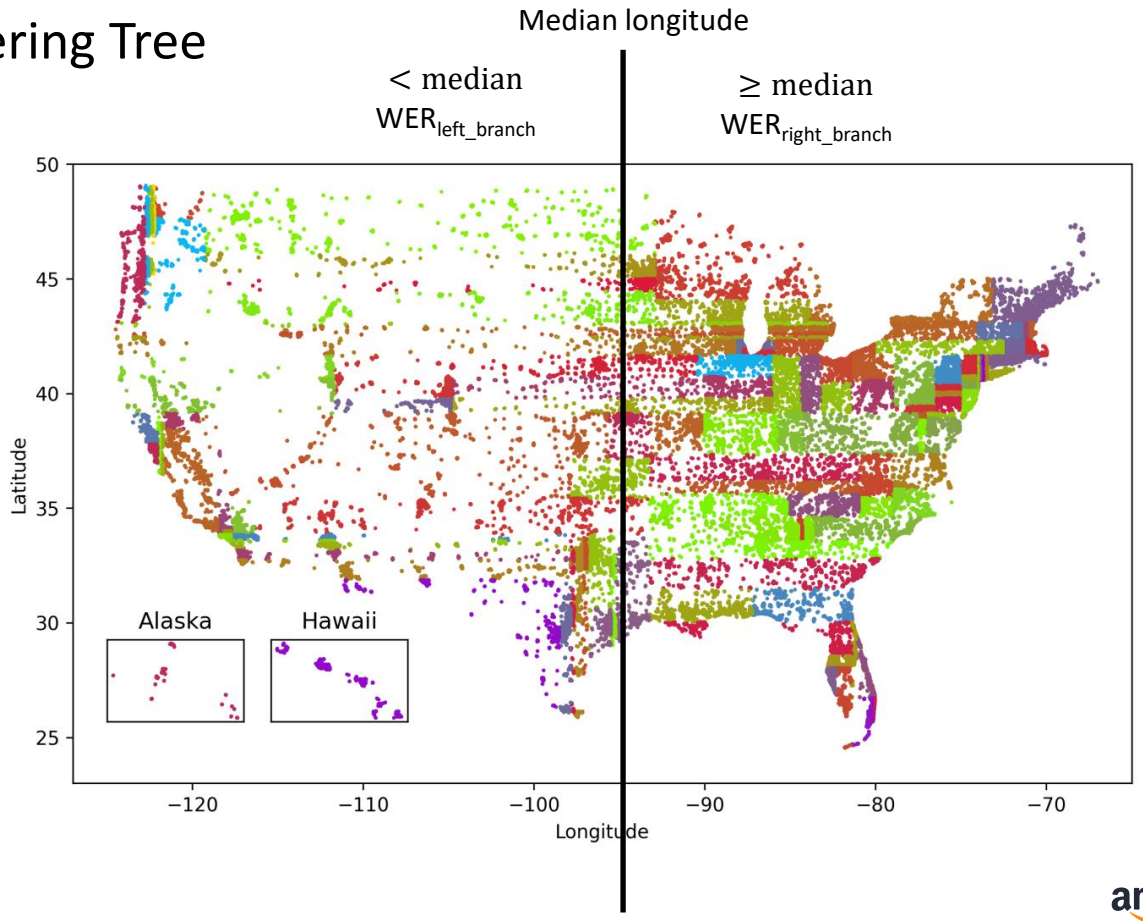
Geographical Clustering by ASR Accuracy

- Use clustering tree to split US data into regions while maximizing word error rate (WER) differences between regions
- $WER\_diff = (WER_{left\_branch} - WER_{right\_branch})^2$
- Split the data by longitude if WER_diff_longitude > WER_diff_latitude and vice versa
- Repeat while the number of devices in each leaf ≥ threshold *x* (ensure each region has at least *x* devices)



Result: Clustering Tree

$$WER\_diff\_longitude = (WER_{left\_branch} - WER_{right\_branch})^2$$



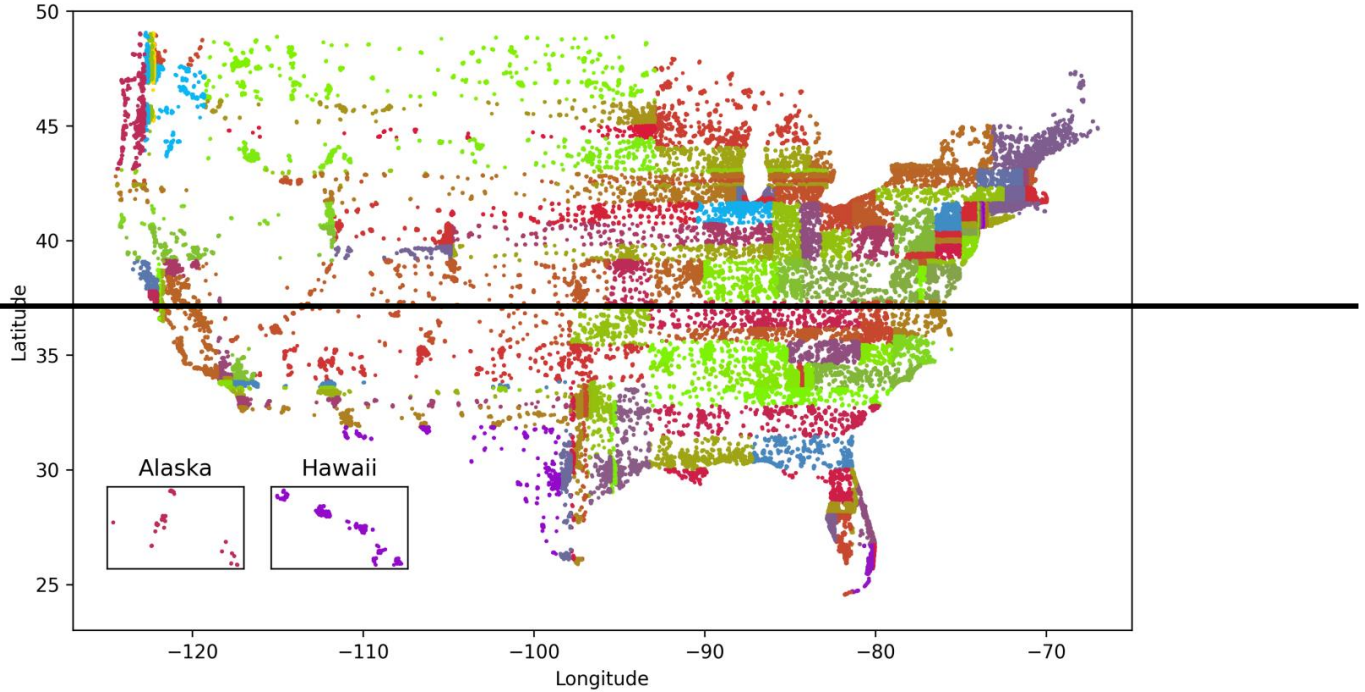
Result: Clustering Tree

WER_diff_latitude = $(WER_{left\_branch} - WER_{right\_branch})^2$

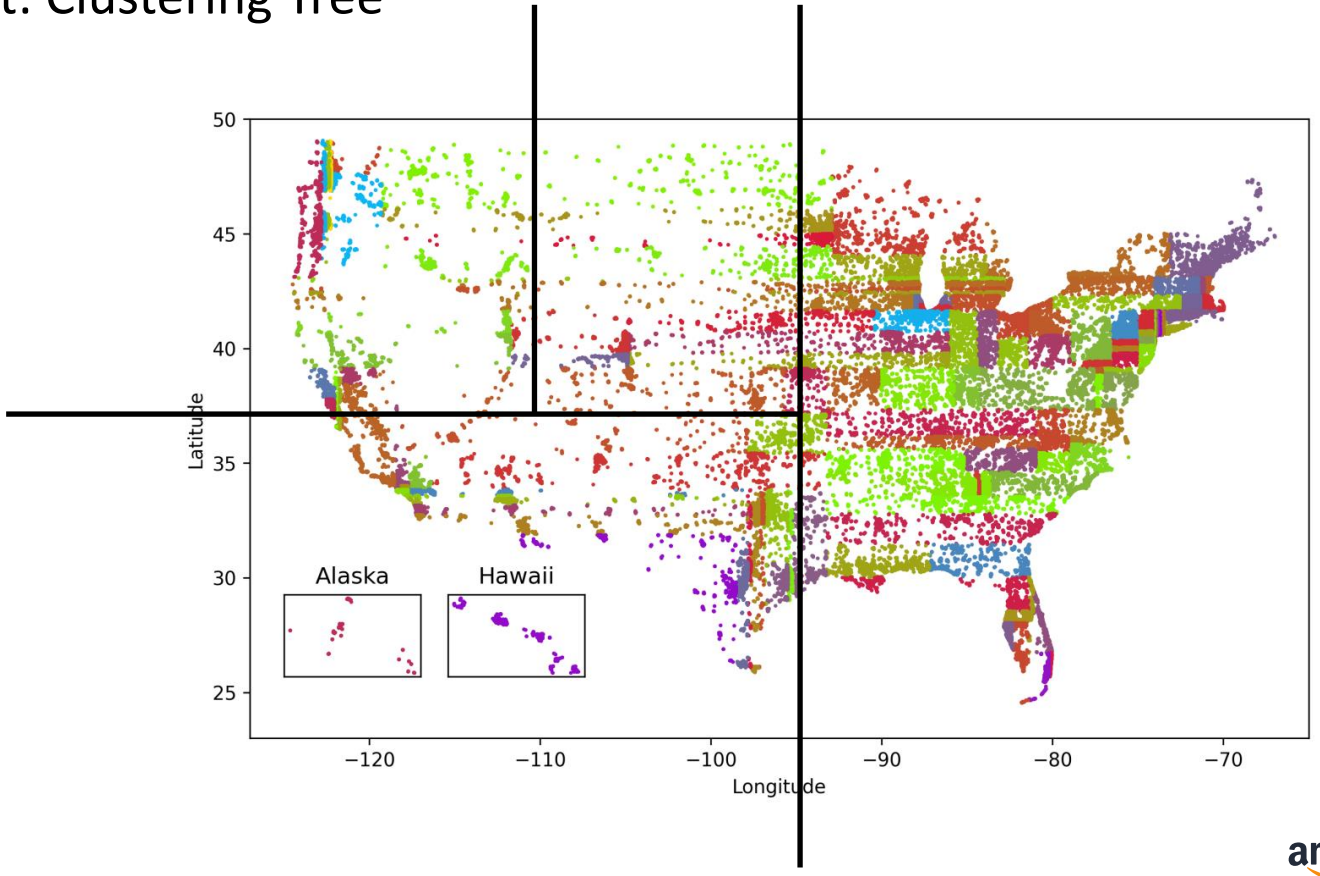
$\geq$ median WER_{right_branch}

Median latitude

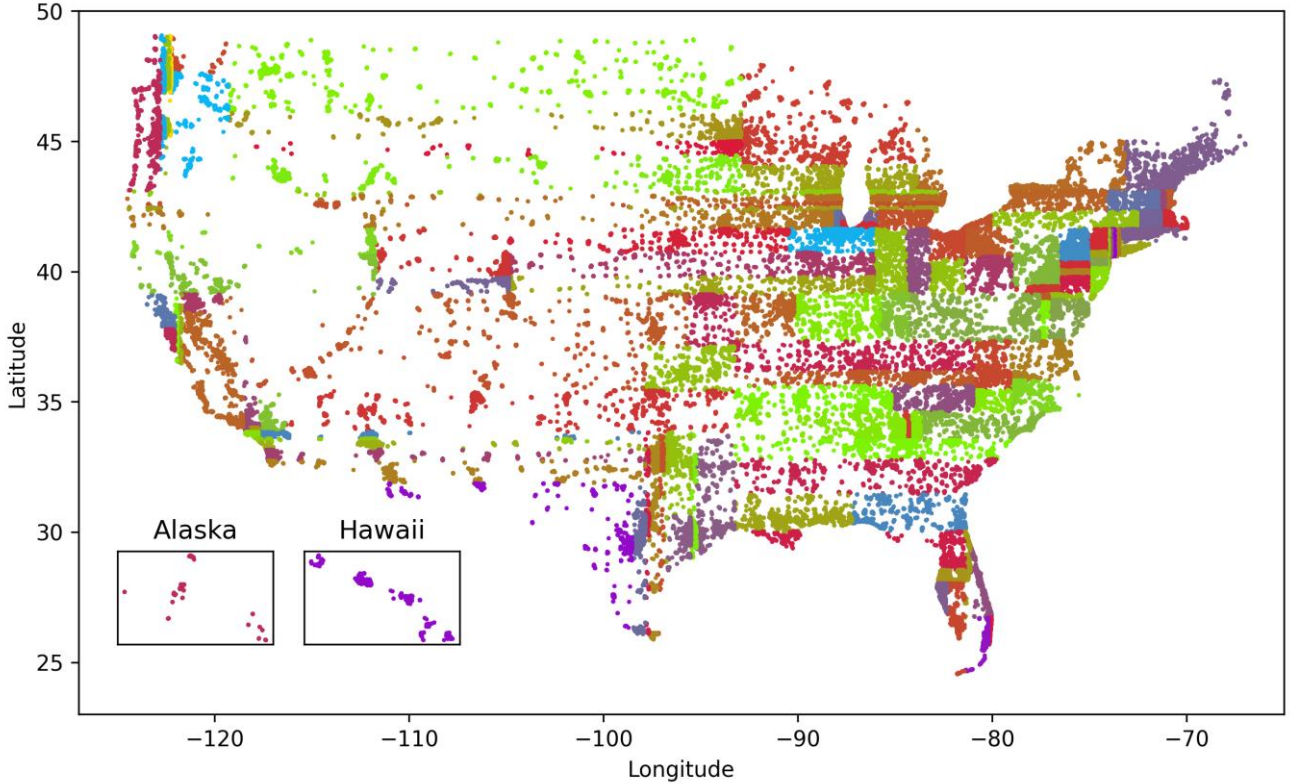
$<$ median WER_{left_branch}



Result: Clustering Tree

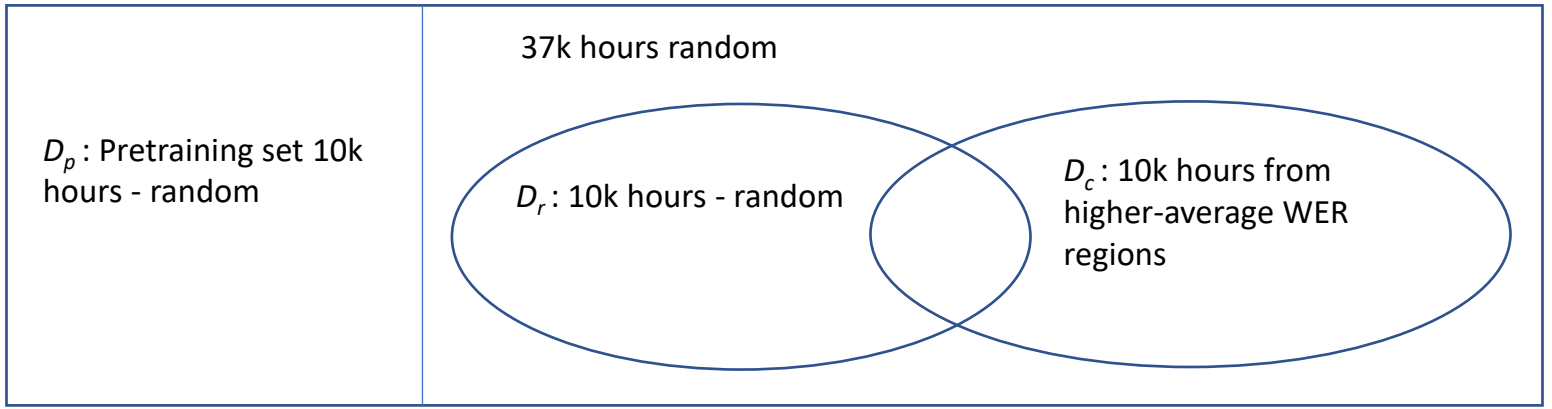


Result: Clustering Tree



Dataset

- De-identified user data from a commercial voice-enabled AI assistant
- Mitigation: collect more data from regions with high WER



Elastic Weight Consolidation

- We propose a loss function that is a combination of the Elastic Weight Consolidation (EWC [1]) loss and the RNN-T loss

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{ASR}}(\theta) + \frac{\lambda}{2}\sum_{\mathrm{i}} \mathrm{F_i}\,(\theta_{\mathrm{i}} - \theta^{*}_{\mathrm{p,i}})^2$$

- Force ASR parameters θ to **be close to the best parameters of the pre-trained model $\boldsymbol{\theta_p^*}$, along the directions that are important to the pretrained task** (based on Fisher information)

Standard transfer learning	EWC
Assign binary importance score (freeze, no freeze)	Assign continuous score (F_i)
Score for each layer	Score for each parameter in each layer
Based on researcher experience, trial and error	Use mathematical criterion (Fisher information) to assign score

[1] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks", *Proc. National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017

Experimental Setup

- Transfer learning: train on $\mathrm{D_p}$, then fine-tune on $\mathrm{D_c}$
  - Exp 1: Train on random 10k hours D_p
  - Exp 2: No freeze
  - Exp 3: Freeze all encoder layers
  - Exp 4: Freeze all predictor layers
  - Exp 5: Freeze 3/5 lowest encoder layers & 1/2 predictor layer
  - Exp 6: Proposed method
- Joint learning
  - Exp 7: train on both D_p and D_c, as empirical bound
  - Exp 8: train on both D_p and D_r

Results

Experiment	Description	Data	Region WER reduction (%)				Overall WER reduction (%)
			variance	mean	max	min	
Experiment 1	Baseline	D_p	0	0	0	0	0
Experiment 2	No freeze	D_c	-5.3	-0.9	-2.9	-4.6	-1.1
Experiment 3	Freeze Encoder	D_c	-1.8	0.0	-1.4	-5.4	-0.1
Experiment 4	Freeze Predictor	D_c	1.8	-0.3	-1.3	-8.5	-0.4
Experiment 5	Freeze 3 lowest encoder layers and 1 predictor layer	D_c	-0.9	-0.5	-2.5	-2.7	-0.4
Experiment 6	Proposed method	D_c	-7.9	-1.1	-3.2	-5.8	-1.3
Experiment 7	Empirical bound	$D_p + D_c$	-5.3	-1.2	-2.3	0.2	-1.0
Experiment 8		$D_p + D_r$	-12.3	-2.3	-0.9	-7.3	-2.1

Summary

- Use geolocation to partition speakers into low and high-WER regions (using a clustering tree based on latitude and longitude attributes)
- Use an RNN-T loss function combining the standard ASR loss with Elastic Weight Consolidation (EWC) regularization loss
  - EWC helps model keep good performance for the user population overall, while reducing WER for underperforming regions
- Our proposed method reduces the WER in the region with highest WER by 3.2% relative and reduces the overall WER by 1.3% relative

Group-adapted Fusion Network for Speaker Verification Fairness

Hua Shen, Y. Yang, G. Sun, R. Langman, E. Han, J. Droppo, A. Stolcke, Improving Fairness in Speaker Verification via Group-Adapted Fusion Network, *Proc. ICASSP*, 2022

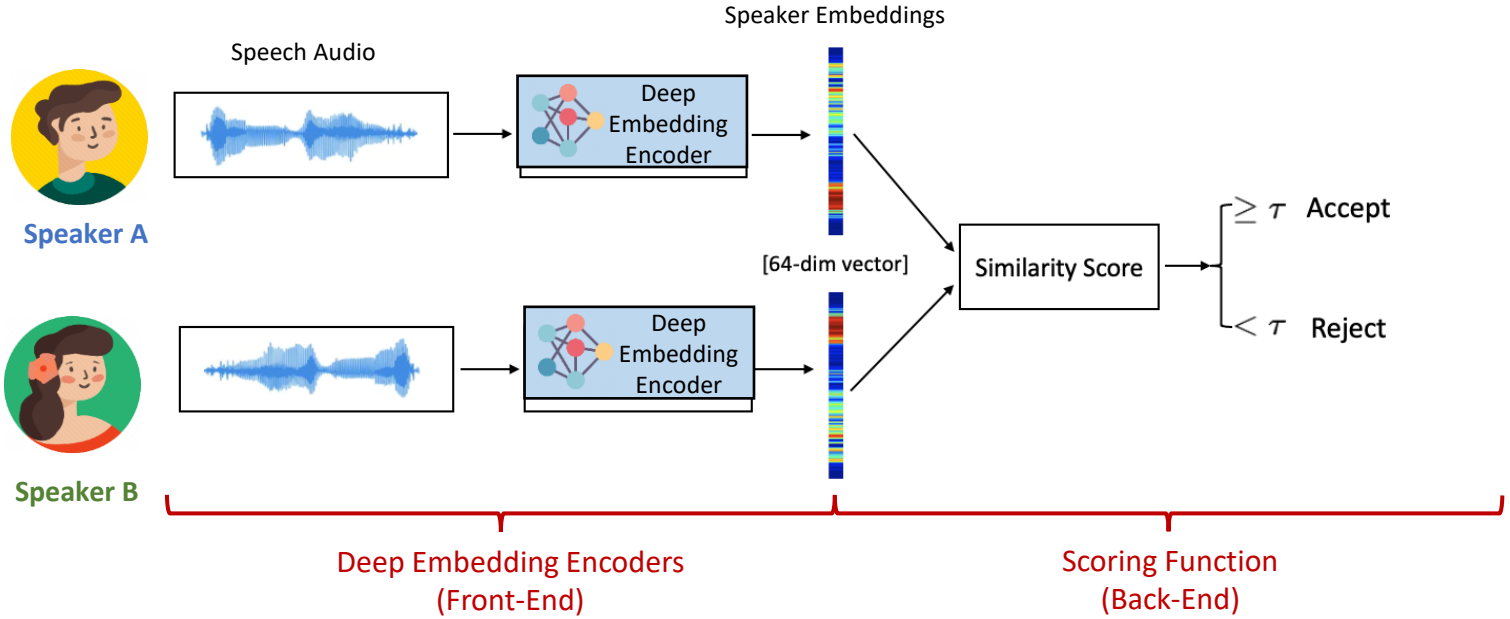
amazon | science

Speaker Verification

Model Architecture

The performance of **speaker verification systems** has dramatically improved due to both **deep learning algorithms** and **large-scale datasets**. The state-of-the-art **speaker verification models** typically have two stages:

1. **Deep embedding encoders (Front-end):** compute speaker embeddings from speech audio;
2. **Scoring function (Back-end):** compute similarity score between two embeddings.

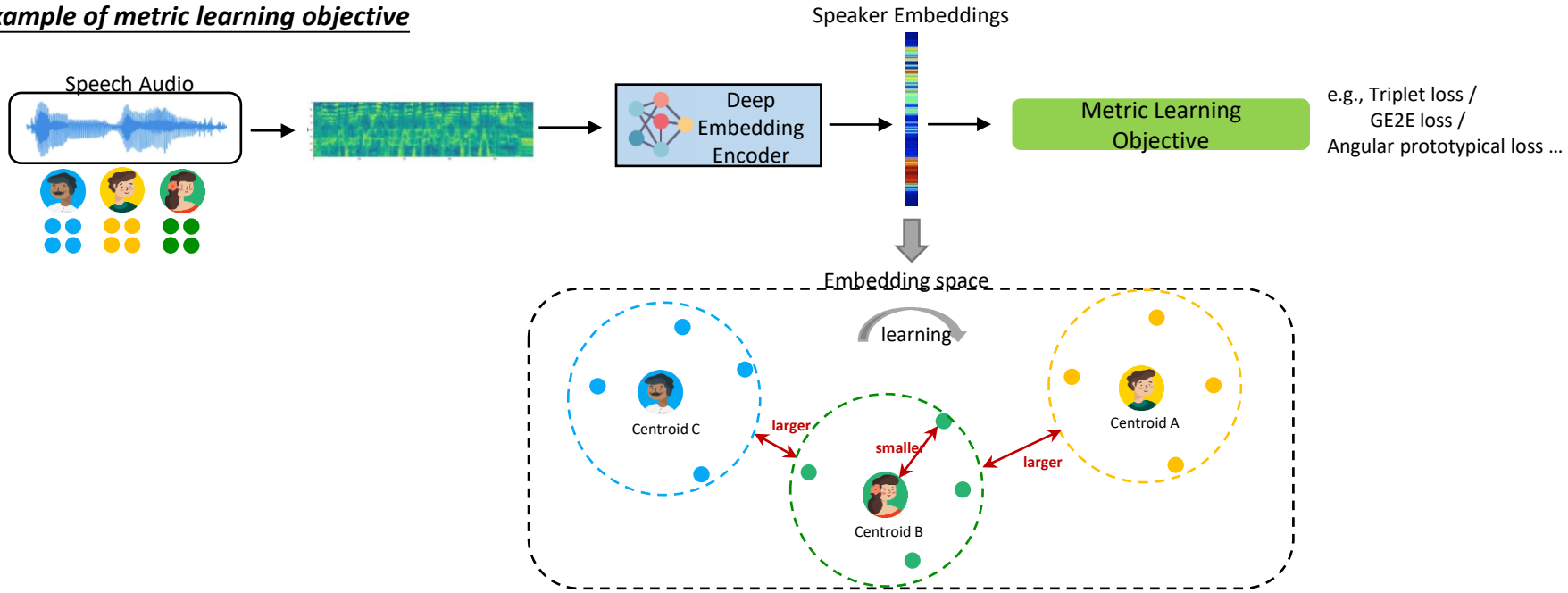


Speaker Verification

Training Process

We commonly **train** the Front-end **deep embedding encoders** with **classification** or **metric learning** objectives.

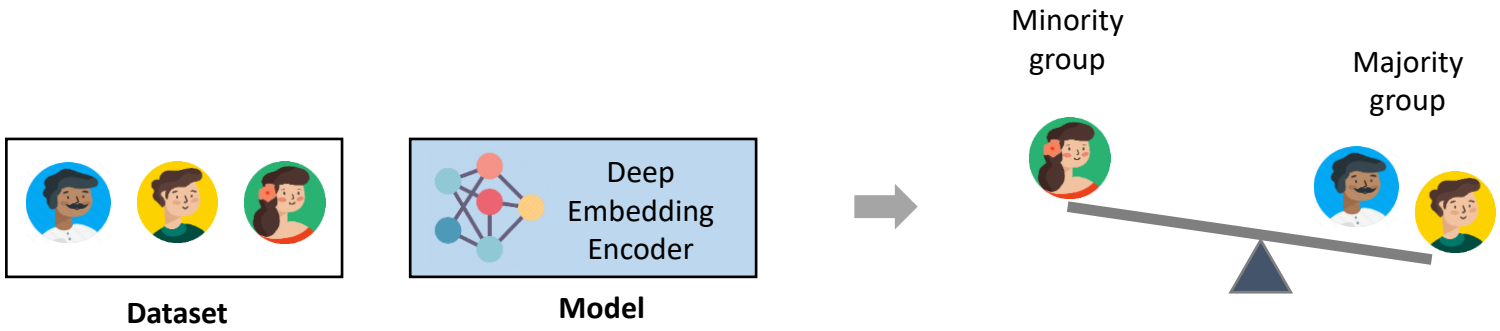
Example of metric learning objective



Learn to optimize the embedding to get:

- **smaller** distance between **same** speakers
- **larger** distance with **different** speakers.

Motivation



However, this learning process can potentially lead to **model unfairness across groups, because:**

- **Training:** Models **minimize average loss** over the full datasets, which might ignore the voice characteristics of **underrepresented groups;**
- **Evaluation:** The **performance metrics** (e.g., EER) typically measure **overall performance**, which does **not reflect performance over different subgroups**.

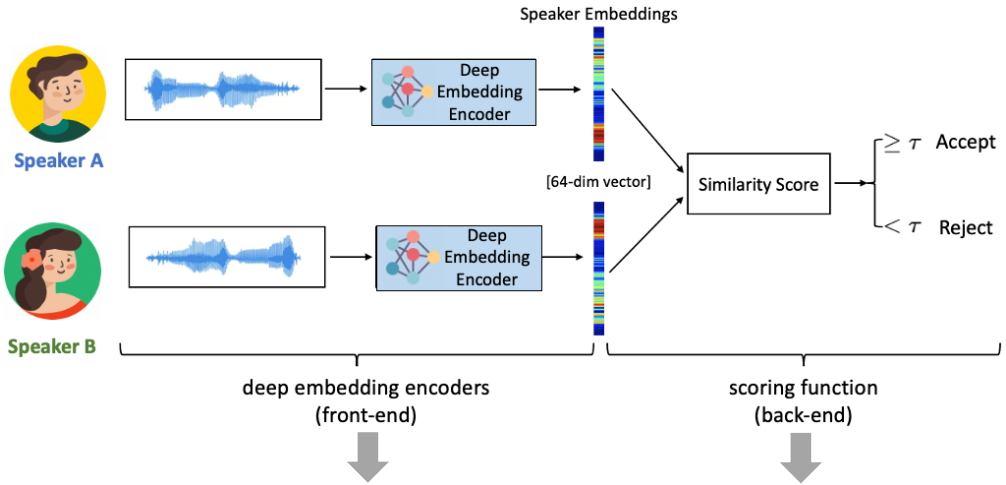
Study Objective

- Rigorously **analyze model unfairness** in speaker verification systems
- Offer a generalizable **solution to mitigate model unfairness**

Contributions

1. We originally **crafted training and evaluation datasets,** and **evaluation metrics,** to rigorously evaluate and analyze model fairness performance.
2. We provide direct evidence showing that **group-imbalanced training dataset can lead to model unfairness** to underrepresented groups.
3. We **propose a flexible, modular model** based on group embedding adaptation and score fusion to **alleviate model unfairness**.

Core Idea



Group Embedding Adaptation

Use **group-wise adaptation encoders** to capture voice characteristics specific to each group

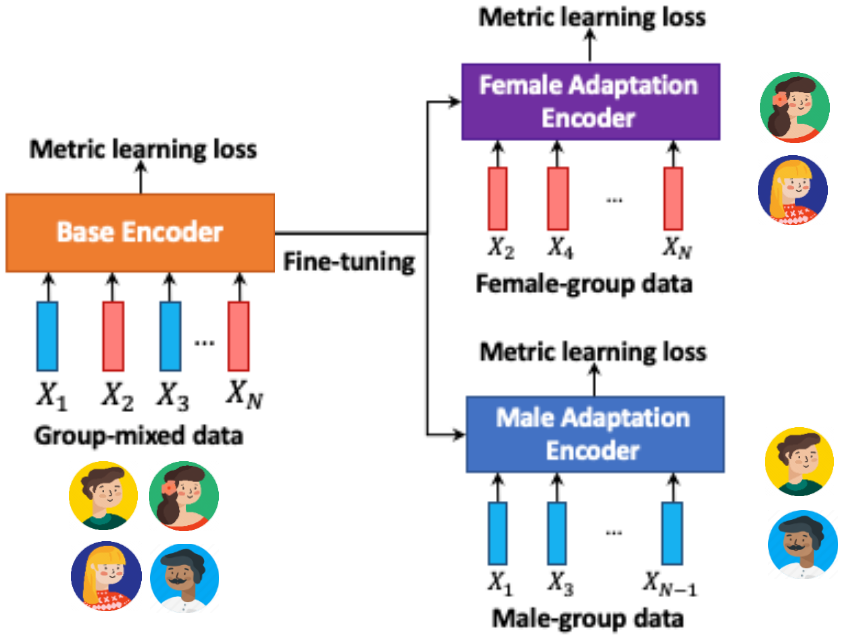
Score Fusion Strategy

Use **score fusion strategy** to aggregate scores from all group-specific embeddings

Group-adapted Fusion Network (GFN)

Group-adapted Fusion Network (GFN)

Front-end



Group Embedding Adaptation

$$\mathbf{E}_i^B = \text{BaseEncoder}(\mathbf{X}_i),\ i = 1, 2$$

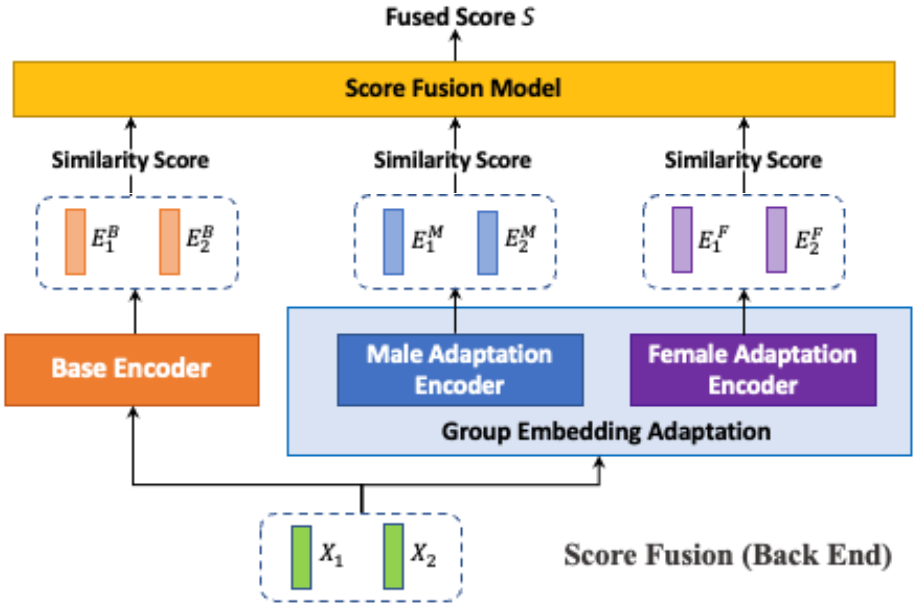
$$\mathbf{E}_i^F = \text{FemaleAdaptationEncoder}(\mathbf{X}_i),\ i = 1, 2$$

$$\mathbf{E}_i^M = \text{MaleAdaptationEncoder}(\mathbf{X}_i),\ i = 1, 2$$

The front-end encoders extract base (general) and group-adapted embeddings.

Group-adapted Fusion Network (GFN)

Back-End



Score fusion model

$$S^B = \text{CosineSimilarity}(E_1^B, E_2^B),$$
$$S^F = \text{CosineSimilarity}(E_1^F, E_2^F),$$
$$S^M = \text{CosineSimilarity}(E_1^M, E_2^M)$$
$$S = \text{Sigmoid}(f([S^B, S^F, S^M]; W)).$$

Neural Network

The back-end score fusion model combines all scores for speaker verification.

Training objective

Binary cross-entropy loss
with positive and negative training pairs

$$L = -\frac{1}{M}\left(\sum_{n\in\mathcal{P}} y_n \log S_n + \sum_{n\in\mathcal{N}} (1-y_n)\log(1-S_n)\right)$$

Crafted Datasets and Metrics for Fairness

Training sets

- Voxceleb2-GRC (Gender Ratio Controlled) Dataset

Front-End

Gender Ratio (Female:Male)	Female Speakers	Male Speakers	Female Utterances	Male Utterances
9:1	2250	250	387,322	45,181
4:1	2000	500	341,500	95,157
1:1	1250	1250	214,919	228,823
1:4	500	2000	86,616	372,133
1:9	250	2250	43,482	419,853
-	**Total Speakers: 2500**		-	

unbalanced

balanced

unbalanced

Back-End

Sample positive (same speaker) and negative (different speakers) training pairs from VoxCeleb2-GRC for metric learning.

Test sets

- Voxceleb1-F (Fairness) Dataset

Gender Trials	Trial Count	VoxCeleb1-F		
		[F]	[M]	[All]
Positive F-F	150,000	✓		✓
Negative F-F	150,000	✓		✓
Negative M-F	150,000	✓	✓	✓
Positive M-M	150,000		✓	✓
Negative M-M	150,000		✓	✓

Crafted Datasets and Metrics for Fairness

Evaluation metrics

Equal error rate (EER) is one of the most common metrics to evaluate speaker verification models, denoting the rate where *False accept rate (FAR) = False rejection rate (FRR).*

Model fairness evaluation via three metrics:

(1) **Group-wise EERs:** monitor group-specific performance

- **Female**-group: $EER[F]$
- **Male**-group: $EER[M]$

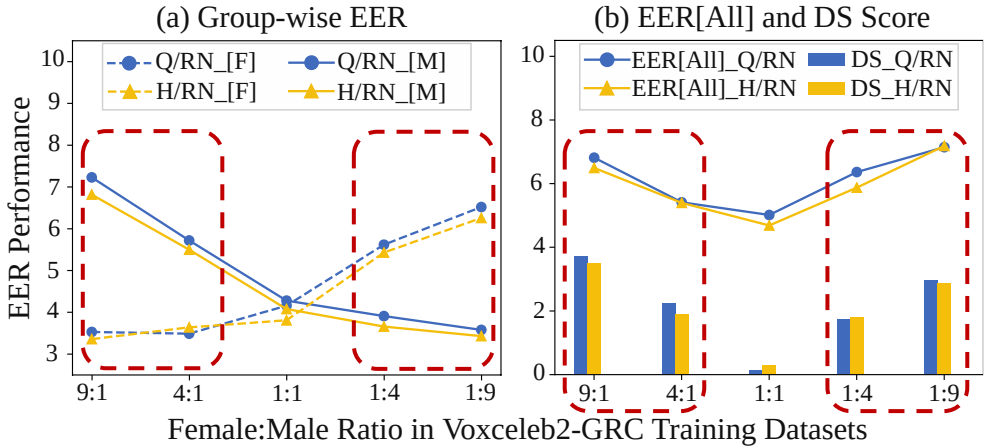
(2) **Overall EERs:** monitor performance across all groups

- Overall EER: $EER[\text{All}]$

(3) **Disparity Score (DS):** model performance gap between groups

- $\text{DiparityScore (DS)} = |EER[F] - EER[M]|$

Does *imbalanced group size* in training *cause* model *unfairness*?



Baselines:

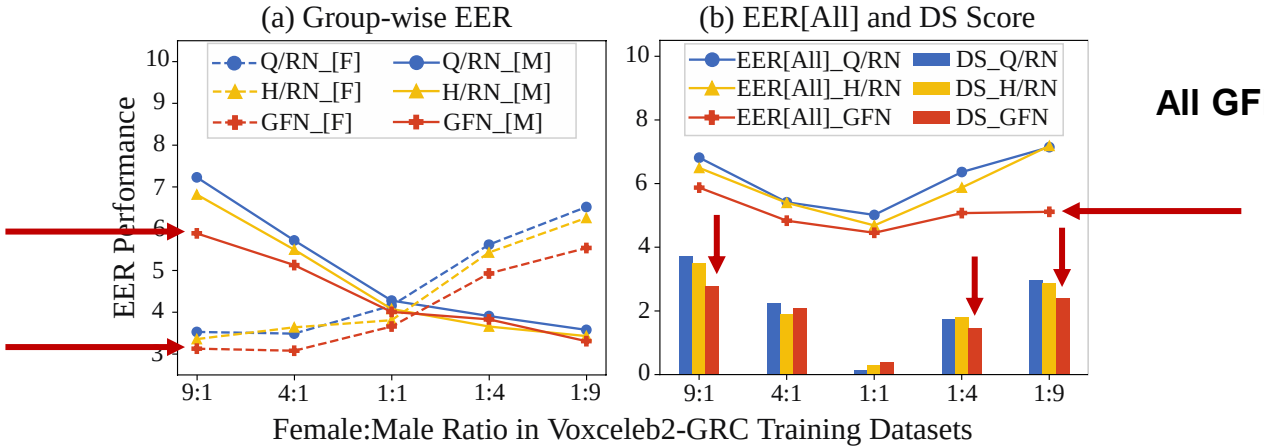
- **Q/RN: Q**uarter-channel **R**es**N**et-34
- **H/RN: H**alf-channel **R**es**N**et-34;

Findings:

- Training with same total speaker numbers (i.e., 2500), the dominant group achieves better group-wise EER than the underrepresented group.
- Increasing dominance of one gender group (e.g., 4:1 → 9:1) leads to increasing performance gap (DS score) and overall EER, indicating increasing model unfairness and worse overall performance, respectively.

Imbalanced group ratios in training sets can lead to model unfairness towards underrepresented groups.

Does GFN mitigate model unfairness?



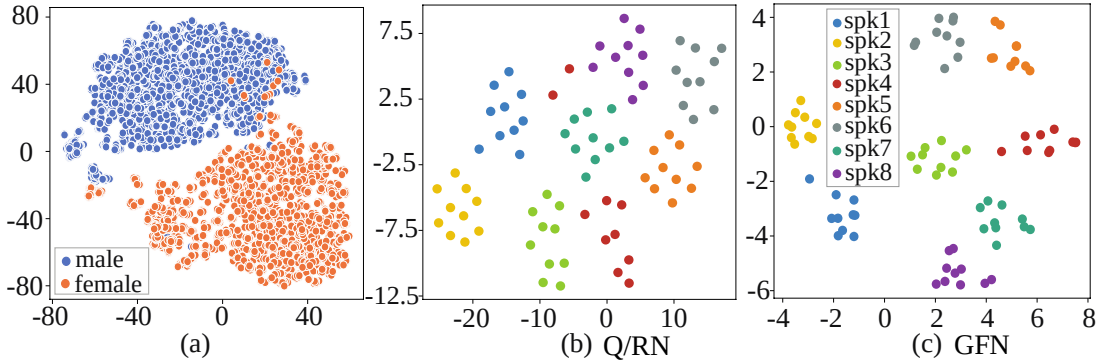
All GFN's encoders: Q/RN

Findings:

- GFN model achieves better group-wise and overall EERs than baselines, regardless of gender group imbalances.
- The GFN also reduces the performance gap (DS Score) in 9:1, 1:4 and 1:9 gender ratio settings.

GFN model can improve gender-specific EER over baselines, and further reduces the performance gap in most imbalanced group ratio settings.

Embedding visualization

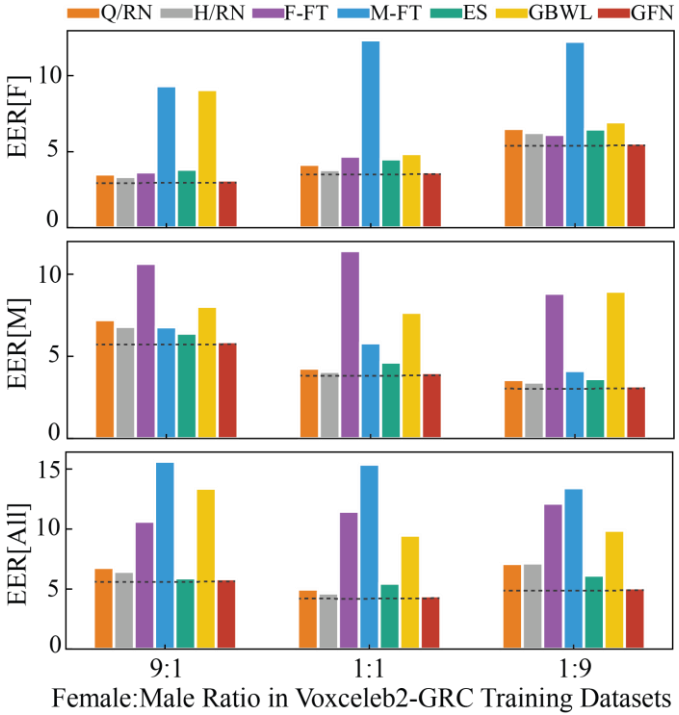


t-SNE projection

Genders tend to aggregate in different regions of the embedding space.

GFN encoder tends to generate higher quality embeddings compared with Q/RN baseline (more compact for the same speakers and separate for different speakers)

Ablation Study



Listing Methods:

- Gender Batching with Weighted Loss (GBWL);
- Equal Score (ES);
- Female-FineTuned (F-FT);
- Male-FineTuned (M-FT);
- Q/RN Baseline;
- H/RN Baseline.

GFN achieves the best performance among all methods.

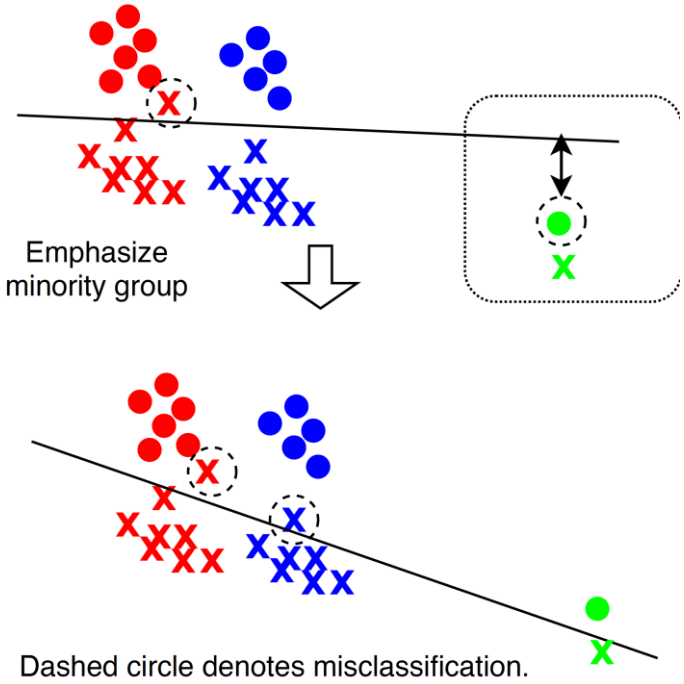
Summary

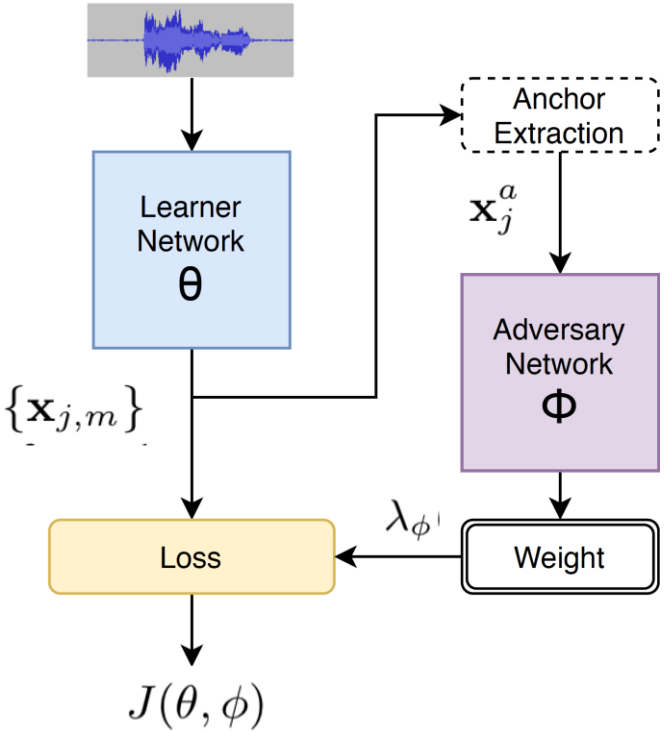
- We use **evaluation metrics** and **datasets with defined group (male/female) ratios** to analyze model fairness performance.
- We provide the direct evidence that **imbalanced group presence can lead to model unfairness** to different subgroups, specialized in gender-group settings.
- We **propose Group-adapted Fusion Network (GFN),** based on group embedding adaptation and score fusion, to counteract model unfairness.
- We demonstrate that **GFN reduces group-disparity** for imbalanced training scenarios, while **reducing overall speaker verification EER.**

Adversarial Reweighting for Speaker Verification Fairness

Minho Jin, C. J.-T. Ju, Z. Chen, Y.-C. Liu, J. Droppo, A. Stolcke, Adversarial Reweighting for Speaker Verification Fairness, *Proc. Interspeech*, 2022

Problem Formulation





Problem Formulation

- The algorithm aims to computationally **identify and boost underperforming groups** in the optimization
- Emphasizing minority groups can lead to **better fairness** and better accuracy.
- To computationally identify minority groups, without additional information, we employ an **adversarial reweighting**:
  - The adversary network outputs weights for each training sample, and is optimized to maximize the loss giving more weight to underperforming samples.
  - The learner is optimized to minimize the weighted loss.
  - This leads to a mini-max optimization given by

$$J(\theta, \phi) = \min_{\theta} \max_{\phi} \sum_{(\mathbf{x}_i, y_i) \in D \times L} \lambda_{\phi}(\mathbf{x}_i, y_i) l(h_{\theta}(\mathbf{x}_i), y_i)$$

Speaker Verification Loss

- The loss $l(h_\theta(\mathbf{x}_i), y_i)$ used for SV is **prototypical loss**
- Given a batch of size *M*, the **pairwise similarity** $\mathbf{S}_{j,k} = w\cos(\mathbf{x}_j^a, \mathbf{x}_k^q) + b$ is computed using a query $\mathbf{x}_j^q = \mathbf{x}_{j,M-1}$ and **anchor** (= speaker enrollment data)

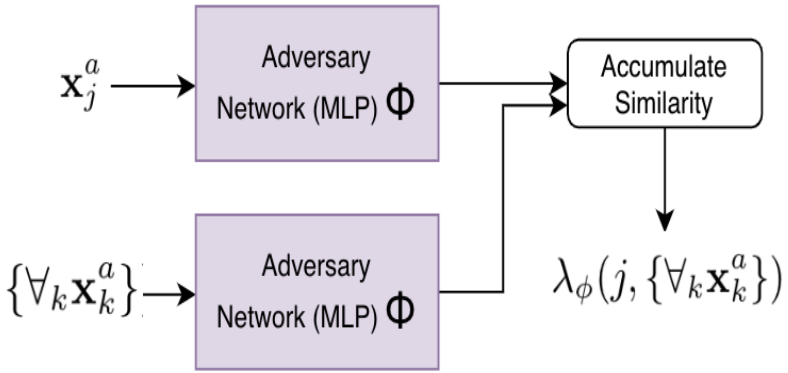
$$\mathbf{x}_j^a = \frac{1}{M-1}\sum_{m=0}^{M-2}\mathbf{x}_{j,m}$$

- Then, the training loss is computed as follows:

$$L_p = \frac{1}{N}\sum_{j=0}^{N-1} L_{p,j} \quad \text{, where} \quad L_{p,j} = -\log\frac{e^{\mathbf{S}_{j,j}}}{\sum_{k=0}^{N-1} e^{\mathbf{S}_{j,k}}}$$

Adversarial Reweighting Approaches (1)

- **APS (accumulated pairwise similarity):** transform each anchor to a space where their pairwise similarity predicts recognition difficulty, with inner product (10) or cosine (11)



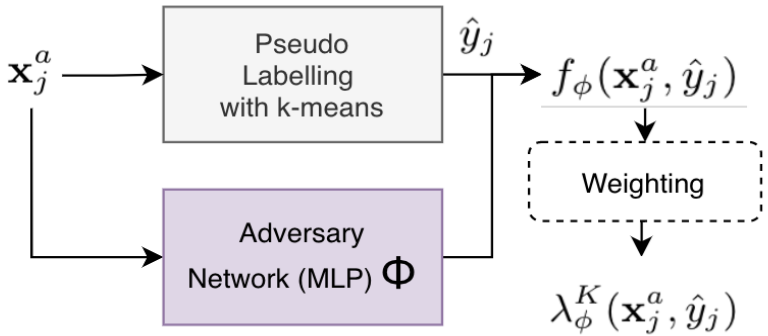
$$J(\theta, \phi) = \min_\theta \max_\phi \sum_{\mathbf{x}_{j,k} \in [D]} \left(\sum_{j=0}^{N-1} \lambda_\phi(j, \{\forall_k \mathbf{x}_k^a\}) L_{p,j} \right)$$

$$\lambda_\phi(j, \{\forall_k \mathbf{x}_k^a\}) = 1 + \frac{\sum_k f_\phi(\mathbf{x}_j^a) \cdot f_\phi(\mathbf{x}_k^a)}{\frac{1}{N} \sum_j \sum_k f_\phi(\mathbf{x}_j^a) \cdot f_\phi(\mathbf{x}_k^a)}, \quad (10)$$

$$\lambda_\phi(j, \{\forall_k \mathbf{x}_k^a\}) = 1 + \frac{\sum_k e^{\cos(f_\phi(\mathbf{x}_j^a), f_\phi(\mathbf{x}_k^a))}}{\frac{1}{N} \sum_j \sum_k e^{\cos(f_\phi(\mathbf{x}_j^a), f_\phi(\mathbf{x}_k^a))}}. \quad (11)$$

Adversarial Reweighting Approaches (2)

- **PL (pseudo labeling):** the entire training data is clustered with k-means, and each training speaker is mapped to a cluster; the adversarial weight is a function of the anchor's cluster ID, serving as a pseudo label



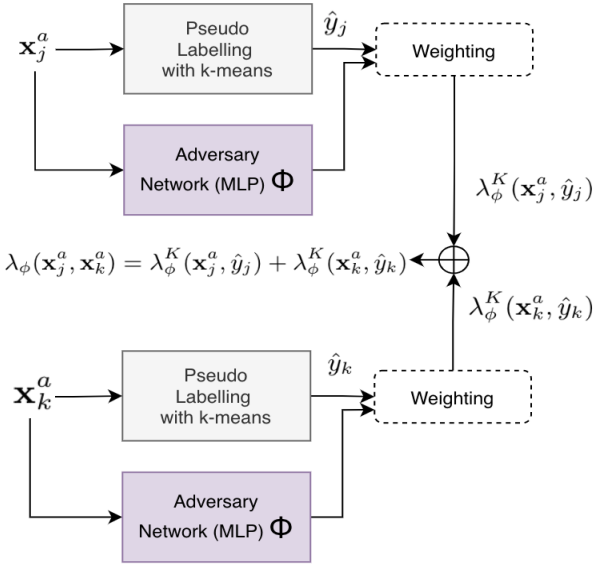
$$\hat{y}_j = \underset{k \in \{0,\ldots,K-1\}}{\arg\max} \cos(\mathbf{c}_k, \mathbf{x}_j^a)$$

$$J(\theta, \phi) = \min_\theta \max_\phi \sum_{\mathbf{x}_{j,k} \in [D]} \left(\sum_{j=0}^{N-1} \lambda_\phi^K(\mathbf{x}_j^a, \hat{y}_j) L_{p,j} \right)$$

$$\lambda_\phi^K(\mathbf{x}_j^a, \hat{y}_j) = 1 + \frac{f_\phi(\mathbf{x}_j^a, \hat{y}_j)}{\frac{1}{N} \sum_{k=0}^{N-1} f_\phi(\mathbf{x}_k^a, \hat{y}_k)}$$

Adversarial Reweighting Approaches (3)

- **PW (pairwise weighting):** the weight is formulated using the pseudo labels (see above) of both anchor and query. Adversarial weights are applied to the similarities either in the exponent (15) or linearly (16)



$$J(\theta, \phi) = \min_\theta \max_\phi \sum_{\mathbf{x}_{j,k} \in [D]} \left(-\frac{1}{N} \sum_{j=0}^{N-1} \log \frac{e^{\lambda_\phi(\mathbf{x}_j^a, \mathbf{x}_j^a)\mathbf{S}_{j,j}}}{\sum_{k=1}^{N} e^{\lambda_\phi(\mathbf{x}_j^a, \mathbf{x}_k^a)\mathbf{S}_{j,k}}} \right), \quad (15)$$

$$J(\theta, \phi) = \min_\theta \max_\phi \sum_{\mathbf{x}_{j,k} \in [D]} \left(-\frac{1}{N} \sum_{j=0}^{N-1} \log \frac{\lambda_\phi(\mathbf{x}_j^a, \mathbf{x}_j^a) e^{\mathbf{S}_{j,j}}}{\sum_{k=1}^{N} \lambda_\phi(\mathbf{x}_j^a, \mathbf{x}_k^a) e^{\mathbf{S}_{j,k}}} \right). \quad (16)$$

Results by Speaker Gender

- The model was trained with 5,994 training speakers in VoxCeleb2, and its equal-error rate (EER) was evaluated with VoxCeleb 1 test data

Method	ALL	female (45%)	male (55%)	gap
Baseline	1.17	0.69	1.39	0.70
APS (10)	1.09	0.67	1.29	0.62
APS (11)	1.12	0.65	1.26	0.61
PL	1.08	0.63	1.27	0.64
PW (15)	1.09	0.65	1.27	0.62
PW(16)	1.08	0.67	1.25	0.58

Results by Speaker Region

Method	US (64%)	UK (17%)	Others (19%)	std.
Baseline	1.09	0.72	1.22	0.21
APS (10)	1.05	0.72	1.24	0.21
APS (11)	1.12	0.80	1.26	0.19
PL	1.07	0.69	1.16	0.20
PW (15)	1.09	0.80	1.26	0.19
PW (16)	1.04	0.76	1.22	0.19

Summary

- Proposed a novel approach to speaker verification fairness based on adversarial reweighting
  - ARW previously used only for classification depending on single input
- This reduces the EER gap between speaker groups based on
  - Gender
  - Locale
- Overall EER is also improved

Synthetic data for ASR robustness to stuttered speech

Xin Zhang, I. Vallés-Pérez, A. Stolcke, C. Yu, J. Droppo, O. Shonibare, R. Barra-Chicote, V. Ravichandran, Stutter-TTS: Controlled Synthesis and Improved Recognition of Stuttered Speech, *Proc. NeurIPS Workshop on Synthetic Data for Machine Learning*, 2022

Background and Motivation

Stuttering is a speech disorder where the natural flow of speech is interrupted by blocks, repetitions or prolongations of syllables, words and phrases.

- The majority of existing automatic speech recognition (ASR) interfaces perform poorly on utterances with stutter, mainly due to lack of matched training data.
- Synthesis of speech with stutter thus presents an opportunity to improve ASR for this type of speech.

Different Stutter Types

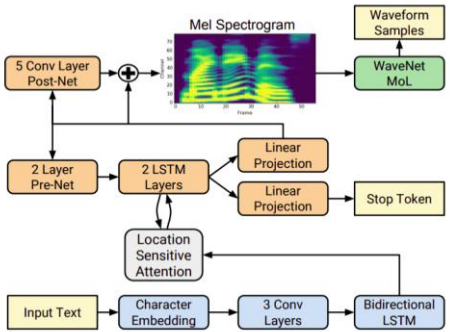
- According to the National Institute on Deafness and Other Communication Disorders, nearly **three million** Americans suffer from lifelong stuttering.

- The following types of disfluencies happen when someone stutters:
  - Part-word repetitions – "I **w-w-w-**want a drink."
  - One-syllable word repetitions – "**Go-go-go** away."
  - Prolonged sounds – "**Sssssssss**am is nice."
  - Blocks or stops – "I want a **(pause)** cookie."

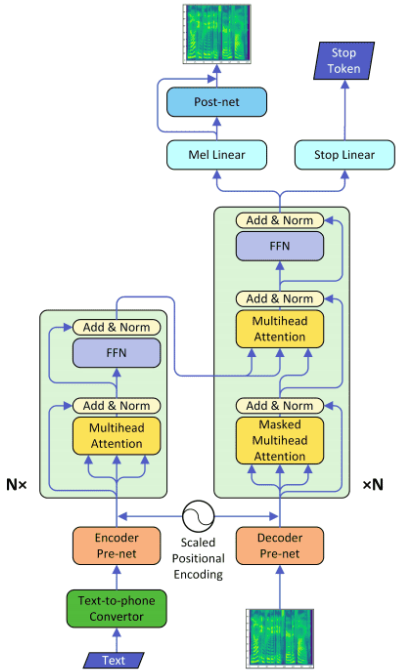
https://www.asha.org/public/speech/disorders/stuttering

Text-To-Speech (TTS)

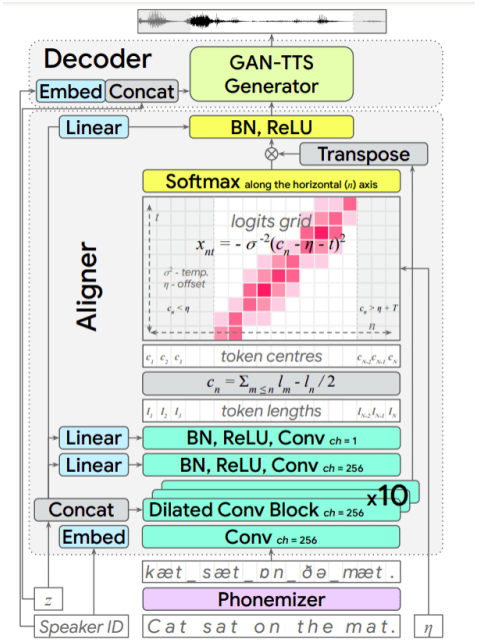
- TTS technology has been widely utilized to produce artificial voices that closely emulate natural human speech.



Tacotron 2
https://arxiv.org/abs/1712.05884



Transformer-based TTS
https://arxiv.org/abs/1809.08895

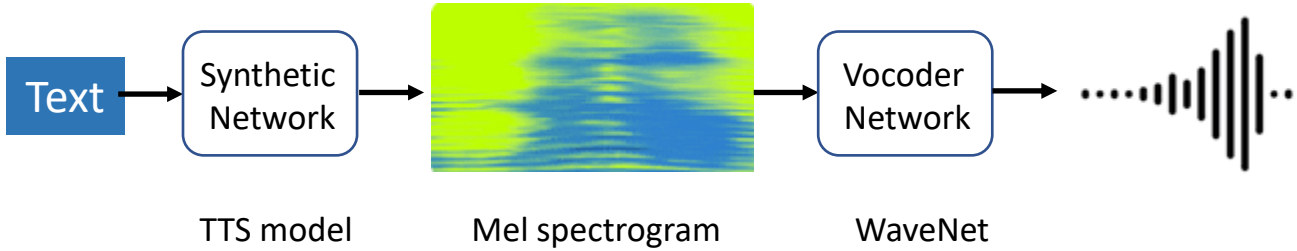


GAN-based TTS
https://arxiv.org/pdf/2006.03575.pdf

Project Objectives & Outcomes

Objectives

We propose ***Stutter-TTS***, an end-to-end neural text-to-speech model capable of synthesizing diverse types of stuttering utterances with controlled prosody.



Outcomes

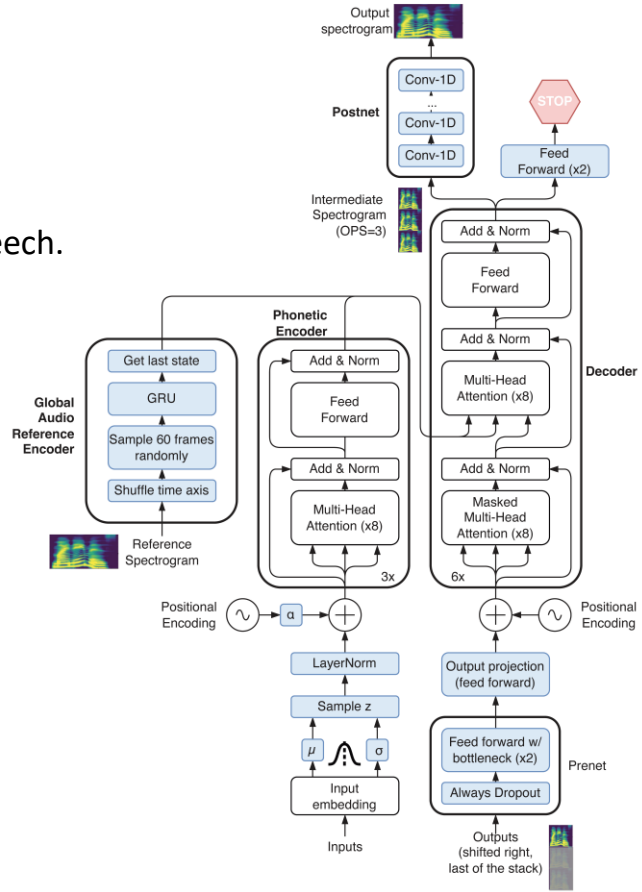
1. Develop a simple yet effective prosody-control strategy to produce specific stuttering characteristics
2. Synthesize stutter events with high accuracy (F1-scores between 0.63 to 0.84, depending on stutter type)
3. Reduce word error rate by 5.7% relative by fine-tuning an ASR model on synthetic stuttered speech

Stutter-TTS Architecture

A multi-speaker transformer-based TTS network is used to model stuttering speech.

(similar to the transformer-based TTS)

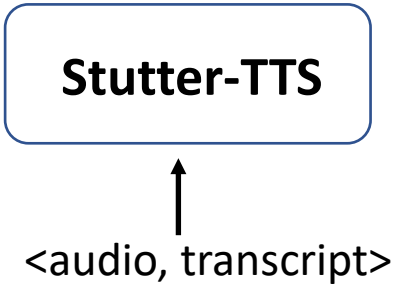
- A phonetic encoder
- An acoustic autoregressive decoder
- An audio reference encoder (speaker identity and prosody)
- A prenet with a strong regularization



Stutter-TTS Training Data

It is critical to train the Stutter-TTS model using a combination of two datasets.

1. Fluent speech (without stutter) captured on close-talking microphones
2. Stuttered speech with human-produced annotation on stutter type



Dataset	# of speakers	# of utterances	Total hours
Stutter	1,000	130,000	600
Fluent	146	18,000	40

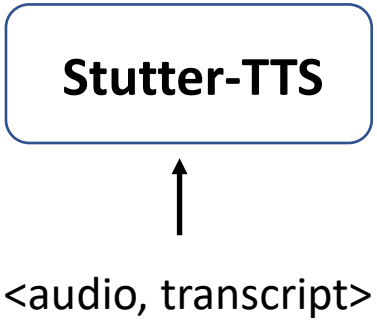
- Utterances in both datasets are 6 to 12 seconds long.
- We employ the universal neural vocoder to synthesize audio samples from spectrograms generated by Stutter-TTS

Training Data Preprocessing--Stutter Tokens

we use a list of special tokens to denote different stuttering patterns and their location.

Stutter type	Stutter token	Rel. frequency (%)
Phoneme repetition	s_repetition	40.11
Dysrhythmic phonation	s_phonation	21.40
Block	s_block	15.59

The mapping rule from different types of stutter
to corresponding tokens inserted in the source sentence.



How stutter tokens work?

I want some coffee please

↓

I want some stutter-token coffee please

- We insert stutter tokens immediately in front of the word where stuttering occurs in the corresponding audio.
- We mainly focus on three common stutter types as detailed in the above table.

Stutter-TTS Training

[stuttered audio, transcript w/ stutter token]

I want some s_repetition coffee please

[fluent audio, transcript w/o stutter token]

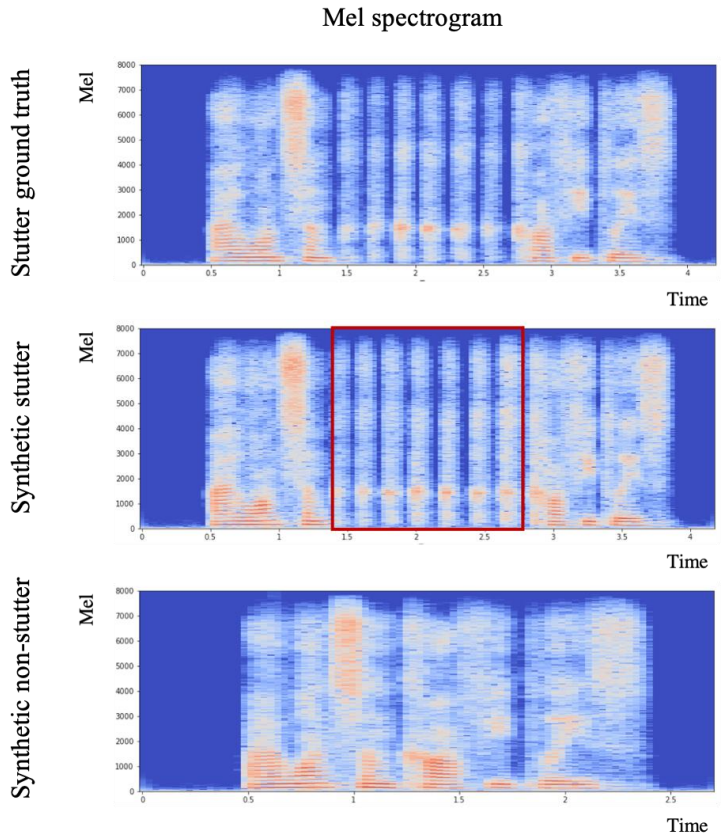
I want some coffee please

Stutter-TTS

During training, each stutter token is mapped to a corresponding special phoneme.

These stutter phonemes are added to the regular phoneme set.

The TTS model thus learns embedding vectors associated with each of stutter type.



Results & Discussions

- We compare the Mel spectrogram generated from Stutter-TTS with the associated recording, collected from speakers with stutter.

- By inserting the stutter tokens in the source text, Stutter-TTS can reproduce the original stutter pattern.

- When eliminating the stutter token from the source text, the resulting synthetic utterance contains no stutter.

F1 scores for correct synthesis of different types of stutter, while varying the ratio of fluent to stuttered utterances in training

Training ratio	Phoneme Repetition	Dysrhythmic Phonation	Block	Non-Stutter
95:5	0.692	0.503	0.720	0.647
90:10	**0.786**	**0.633**	0.837	**0.733**
85:15	0.773	0.615	**0.853**	0.575

Results & Discussions

- We randomly sampled 500 utterances containing phoneme repetition, dysrhythmic phonation, block and non-stutter to quantify the generation performance.

- By varying the ratio between two types of utterances when sampling data for training, a fluent-to-stuttered ratio of 90:10 gives a good compromise between over- and inder-generating stutter events.

Relative change in ASR WER when fine-tuning an RNN-T ASR model using different ratios of fluent to synthetic stuttered speech

Training ratio	Test set with stutter	Test set without stutter
99:1	-2.78	2.13
97:3	**-5.74**	**0.18**
95:5	-3.89	0.35
90:10	-4.17	1.24

Results & Discussions

- We fine-tuned an RNN-T based ASR model for additional five epochs using the 100 hours of synthetic utterances produced by Stutter-TTS and 66k hours of fluent speech (the same data used for training the baseline model).

- A good tradeoff is obtained by sampling 3% of the training data from synthetic speech with stutter, achieving 5.74% relative WER reduction for human utterances with stutter.

Summary

1. Develop a simple yet effective prosody-control strategy to produce specific stuttering characteristics
2. Synthesize stutter events with high accuracy (F1-scores between 0.63 to 0.84, depending on stutter type)
3. Reduce word error rate by 5.7% relative by fine-tuning an ASR model on synthetic stuttered speech

Wrapping Up

Takeaways

- Both speaker recognition (ID, verification) and speech recognition systems suffer from unequal performance for different groups
- Underperformance is typically associated with underrepresentation in the training data
- Mitigation by increasing representation of the targeted group in the overall training loss
- This can be done with a variety of techniques:
    - Oversampling based on observable features (geolocation, demographic proxies)
    - Acoustic feature-based clustering of cohorts
    - Group-specific model training and combination (implicitly reweighted)
    - Adversarial reweighting of training samples (no labels required)
    - Fabrication (synthesis) of group-representative training data

Open issues

- More systematic exploration of the fairness method space (as mapped out at the beginning)
- How to predict which methods work for what task (depending on model type/size, metrics, amount of data, etc.)
- How to compare “fairness” for different notions of groups and attributes
  - Can we define a “GINI coefficient” for model fairness, similar to how economists characterize wealth/income distributions?
  - How to optimize for it?

Thank You!

More information at https://www.amazon.science/author/andreas-stolcke

amazon | science