AccentNet

Neural Accent Conversion via Disentangled Speech Representations

Group 12:
Arjun Agarwal
Bhushan Mohol
Lakshaya Karthikeyan
Manav Chouhan

Why choose Accent Conversion?

In multilingual countries like Singapore and India, people are from significantly diverse linguistic and cultural backgrounds which gives rise to varied accents.

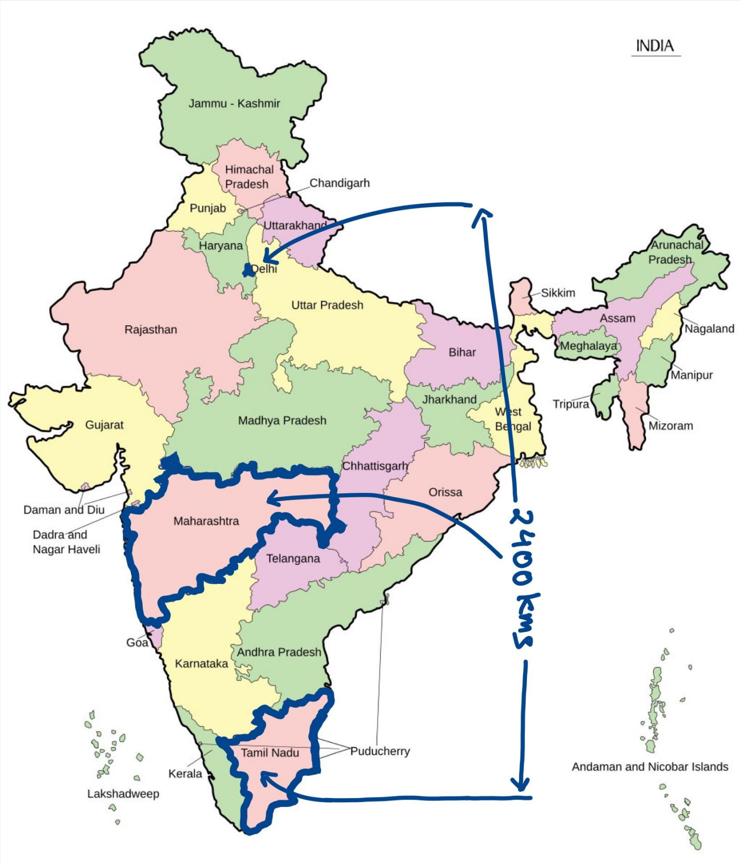
Even interactions in a common language leads to uneven comprehension across local, regional, and international listeners due to these accent differences.

Unfamiliar accents force listeners to expend extra mental effort to decode speech.

This impedes effective communication in various environments:

- ↳ **Education**: Hindered classroom learning and engagement.
- ↳ **Healthcare**: Risk of miscommunication in patient-doctor dialogue.
- ↳ **Business**: Reduced customer satisfaction and slower cross-border collaboration.

Even in our group, we come from 3 states of India with very different languages and hence very different accents.



Why Data Scarcity Limits Accent Conversion

Lack of Parallel Data

- Ideally same sentence spoken by the same speaker in different accents.
- But this “ground truth” is obviously missing

Entangled Features

- Accent and speaker identity are intrinsically intertwined in speech data

Limited Accent Data

- Most of the public speech datasets focus on UK/US english only
- Accented data available has some labelling issues

Accent Datasets

VCTK

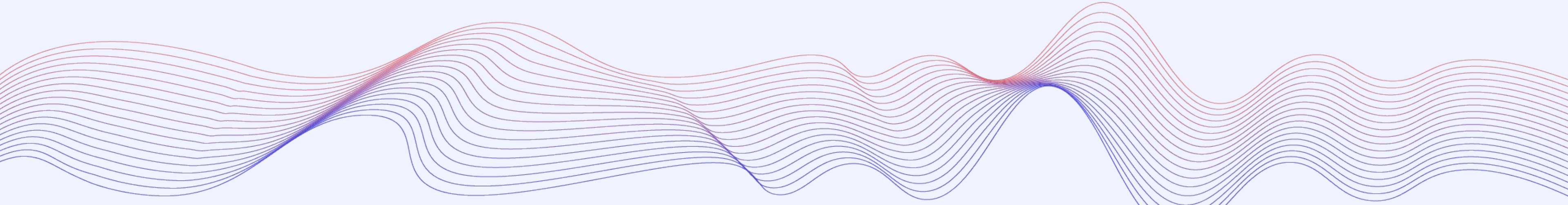
- ↳ 110 speakers, various accents
- ↳ ~44 hours English speech
- ↳ ~400 sentences per speaker

AccentDB

- ↳ ~6.5k audio clips
- ↳ 150+ speakers
- ↳ 4 Indian-English accents

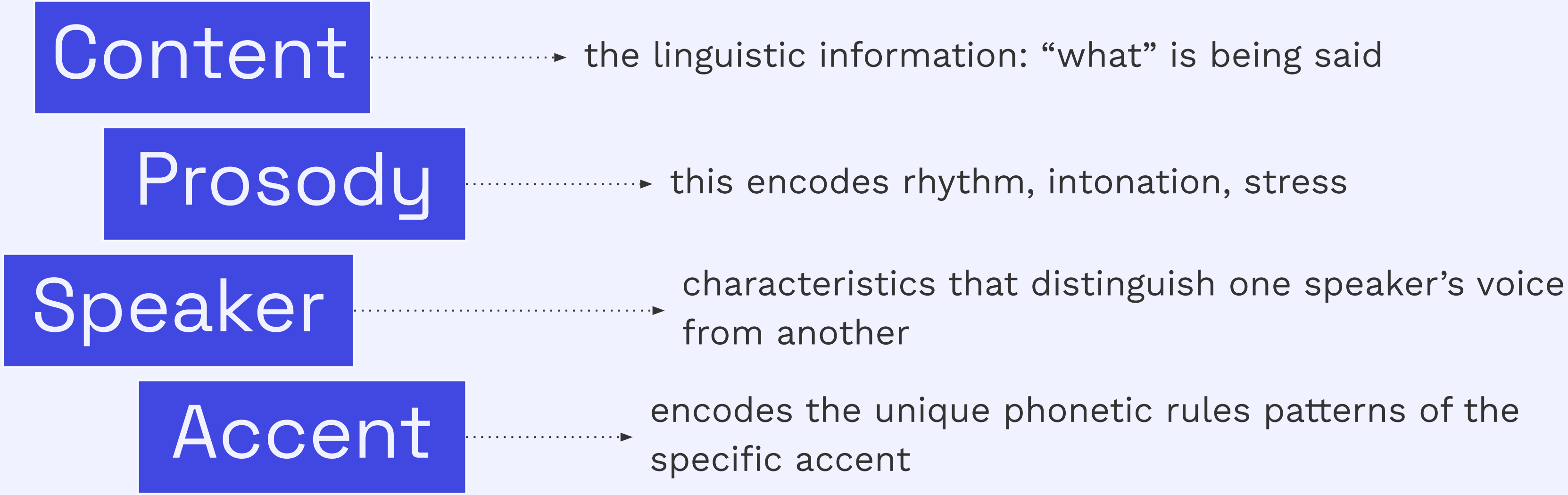
L2-ARCTIC

- ↳ Non-native English speakers
- ↳ 24 different speakers
- ↳ ~1 hour per speaker
- ↳ Includes Indian L1 group



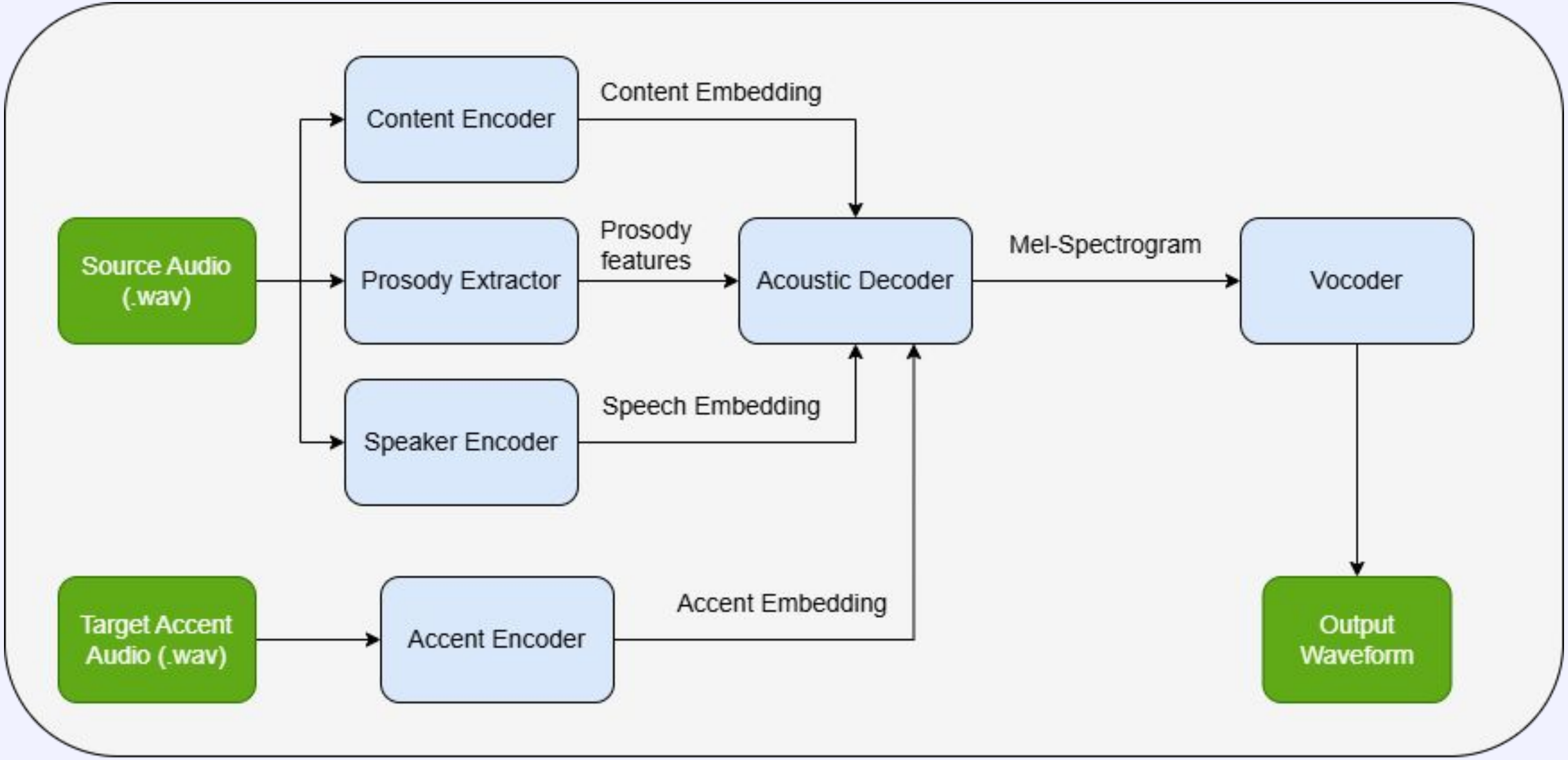
A Disentangled Feature Pipeline

From the accent labelled audios we are extracting independent latent factors as embeddings that represent:



- **Content** → the linguistic information: “what” is being said
- **Prosody** → this encodes rhythm, intonation, stress
- **Speaker** → characteristics that distinguish one speaker’s voice from another
- **Accent** → encodes the unique phonetic rules patterns of the specific accent

Proposed Architecture



Speaker-Specific Characteristics

Why we need it?

- Capturing speaker identity (vocal timbre, style) allows to preserve the speaker’s voice during accent conversion

Why ECAPA-TDNN?

- State-of-the-Art speaker verification model
- Creates robust embeddings which separate unique speaker timbre from the content.

How it extracts?

- It uses Res2Net + TDNN blocks to capture multi-scale temporal features
- Squeeze and Excitation (SE) channel attention highlights speaker-relevant cues
- Attentive statistics pooling aggregates useful frames into a fixed-length vector
- Outputs a compact speaker embedding (192D) that represents vocal identity

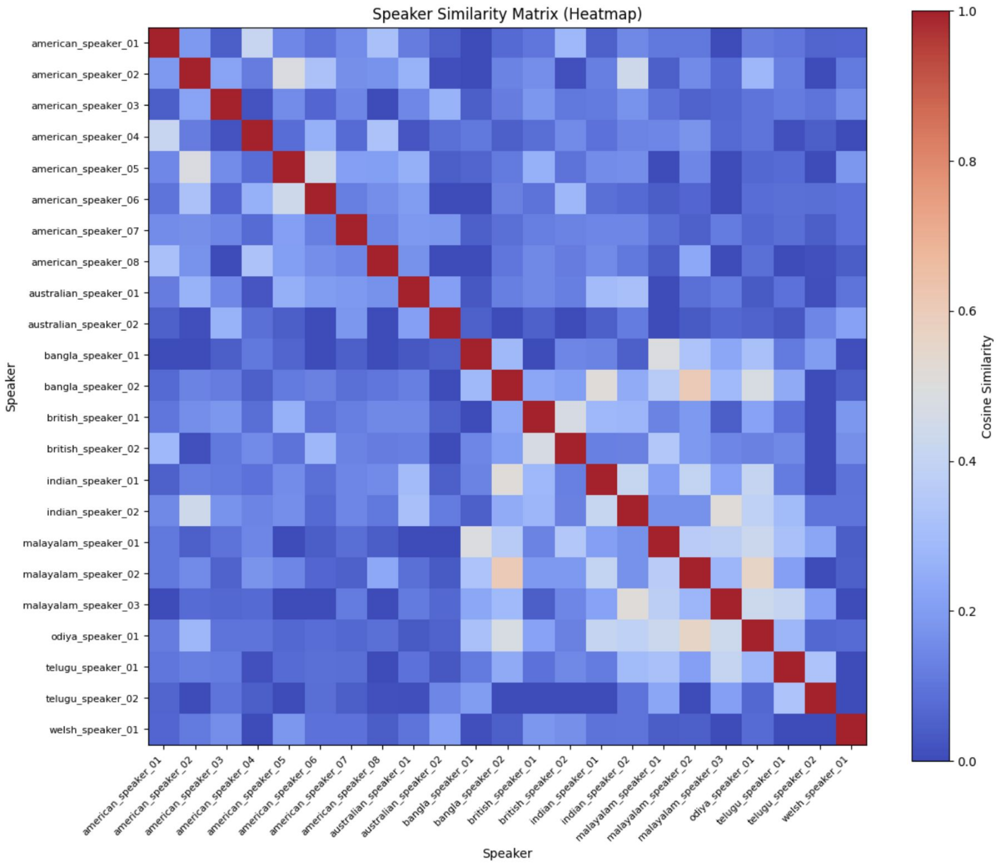
Evaluating Speaker Identity

Similarity matrix using pairwise cosine similarity across speakers for validation:

- Same-speaker pairs has highest similarity ~1.0, ie. ECAPA clusters same speaker utterances closely
- Good inter-speaker separation: Different speakers exhibit much lower similarity (~0.1–0.4).
- Some speakers from similar accents show slightly elevated similarity (eg. Malayalam is ideally a subset of Indian)

Overall, ECAPA embeddings have

- high intra-speaker consistency
- clear inter-speaker separability



Disentangling Content

What is ContentVec?

- SSL (self-supervised) speech representation model
- Based on HuBERT framework

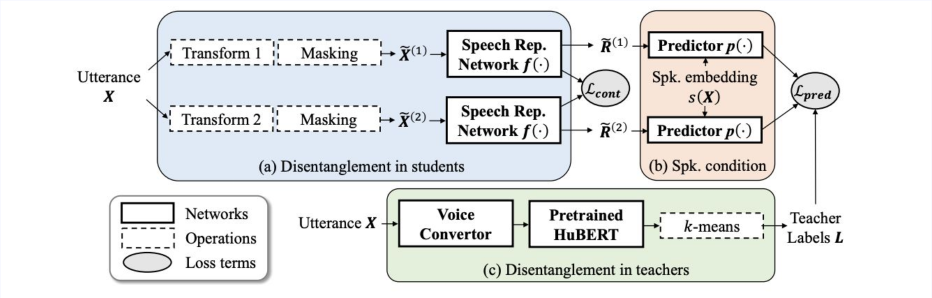
How it works?

It adds 3 disentanglement mechanisms on top of HuBERT :

- in teacher labels
- in student representations
- speaker-conditioning in prediction network

Why ContentVec?

- Need SSL as audio data not always labelled
- But SSL features often retain strong speaker cues eg. HuBERT speaker identification: 81.4% accuracy
- USP: Generates phonetic content while suppressing speaker cues



Disentangling Content

How we used contentvec?

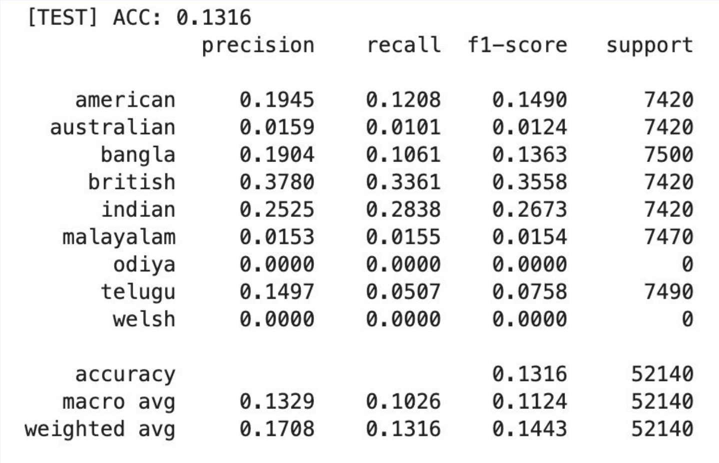
- Used the pre-trained checkpoint \`checkpoint_best_500.pt\`
- Input: Resampled audios 16k + speaker embeddings
- Model produces frame-level content representation

Evaluation

- A simple accent classifier was trained based on the frame-level output
- Classifier poorly predicted accent (~13%) when output for different speakers of same accent where used.

Limitations

- ContentVec reduces speaker cues well but doesn’t remove them completely.
- Different frames from same speaker as training classify at 100%, showing residual speaker cues in the embeddings.



```
[TEST] ACC: 0.1316
              precision    recall  f1-score   support

    american     0.1945    0.1208    0.1490      7420
  australian     0.0159    0.0101    0.0124      7420
      bangla     0.1904    0.1061    0.1363      7500
     british     0.3780    0.3361    0.3558      7420
      indian     0.2525    0.2838    0.2673      7420
   malayalam     0.0153    0.0155    0.0154      7470
       odiya     0.0000    0.0000    0.0000         0
      telugu     0.1497    0.0507    0.0758      7490
       welsh     0.0000    0.0000    0.0000         0

    accuracy                         0.1316     52140
   macro avg     0.1329    0.1026    0.1124     52140
weighted avg     0.1708    0.1316    0.1443     52140
```

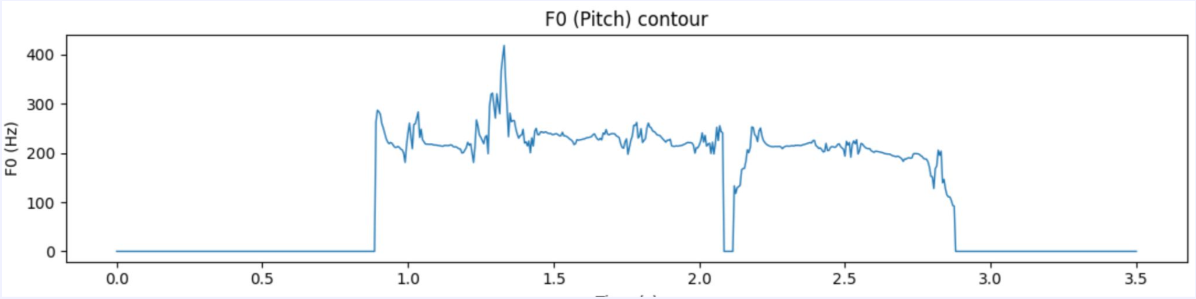
Prosody Extraction

What is Prosody?

- It refers to the rhythm, stress patterns and intonations that shape how speech sounds
- We use F0 to represent speaker's prosody

Why Only F0?

- F0 encodes major prosodic cues like intonation, stress and rhythmic variation.
- Also to keep embeddings lightweight (<1kb)



Prosody Extraction

How did we extract F0?

- ↳ **PyWorld**: a non-neural & deterministic vocoder for extracting F0
- ↳ F0 contour was extracted using the **Harvest + Stonemask** algorithm

- ↳ For perceptual accuracy, we convert F0 to the **Logarithmic Scale** (ln(F0))
- ↳ F0 has 0-value gaps for unvoiced segments, which are filled using interpolation for creating a continuous signal.

Synchronising Prosody to Content Embeddings

- ↳ Extracted with a **5 ms hop size** to obtain **high-resolution pitch contours**.
- ↳ **Downsampled** to match the frame rate of the content embeddings.
- ↳ **Truncated** to ensure temporal alignment by removing any extra tail frames.

Learning Accent Features

What are accent embeddings?

Numerical vectors that capture the unique **pronunciation** and **prosodic patterns** of a speaker's accent for use in speech or TTS models.

How did we use it?

Conformer-based encoder on speech spectrograms to generate accent embeddings for automatic accent classification and analysis using ~17k British/Indian utterances (VCTK + L2-ARCTIC + AccentDB)

Training Approach

- ↳ Dual-objective adversarial training
- ↳ **Maximize** accent classification accuracy
- ↳ **Minimize** speaker identification capability through **gradient reversal.**

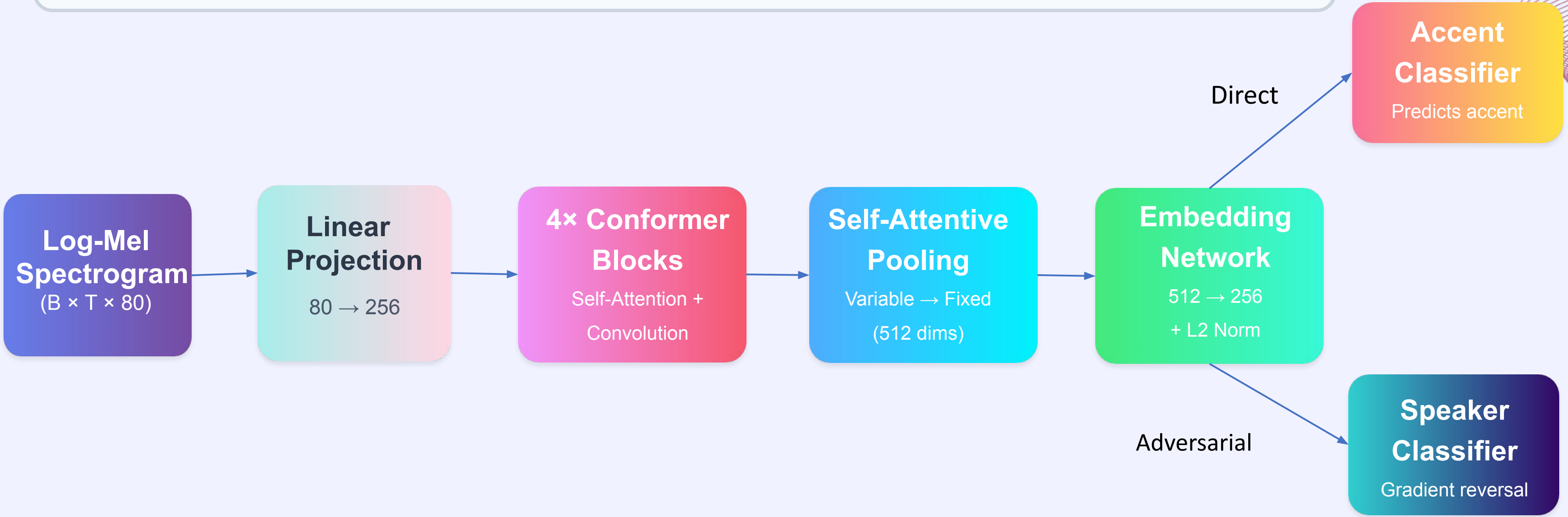
Why speaker-invariance matters?

Without it, the model learns speaker identity instead of accent patterns, failing to generalize across different speakers with the same accent.

Accent Encoder Architecture

Training Objective

Learn speaker-invariant representations that capture accent characteristics through adversarial training



Log-Mel Spectrogram
(B × T × 80)

Linear Projection
80 → 256

4× Conformer Blocks
Self-Attention + Convolution

Self-Attentive Pooling
Variable → Fixed
(512 dims)

Embedding Network
512 → 256
+ L2 Norm

Direct

Accent Classifier
Predicts accent

Adversarial

Speaker Classifier
Gradient reversal

Piecing them all together

What are we doing?

We take the 4 generated embeddings:

↳ content, accent, speaker and prosody features

↳ generate mel-spectrogram

↳ convert to audio using a vocoder

How can we do it?

There are 2 main approaches for the Decoder:

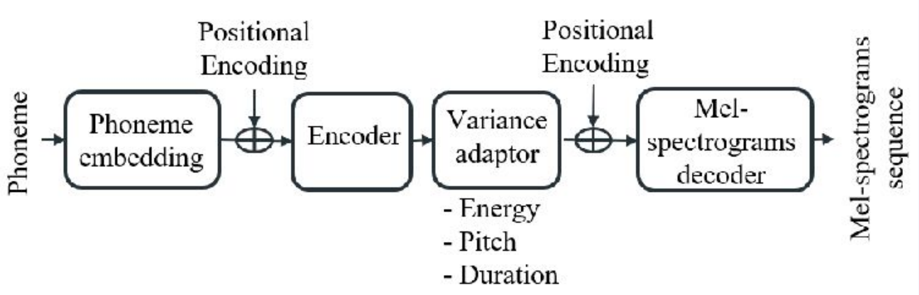
↳ Transformer-based: FastSpeech2 base

↳ Diffusion-based: U-Net Diffusion model base

Using FastSpeech2 as the base

What is FastSpeech2?

- ↳ Non-autoregressive model which synthesizes speech from linguistic data
- ↳ Generates all frames in parallel, making the process faster



Our Proposed approach based on it

- ↳ Use Content embedding attached with Accent and Speaker embeddings instead of Phoneme embeddings
  - ↳ Inject prosody features from source speaker instead of predicting Energy, Pitch and Duration
  - ↳ Finetune pre-trained Mel-Spectrogram Decoder on generated embeddings
  - ↳ Use target mel-spectrogram as groundtruth

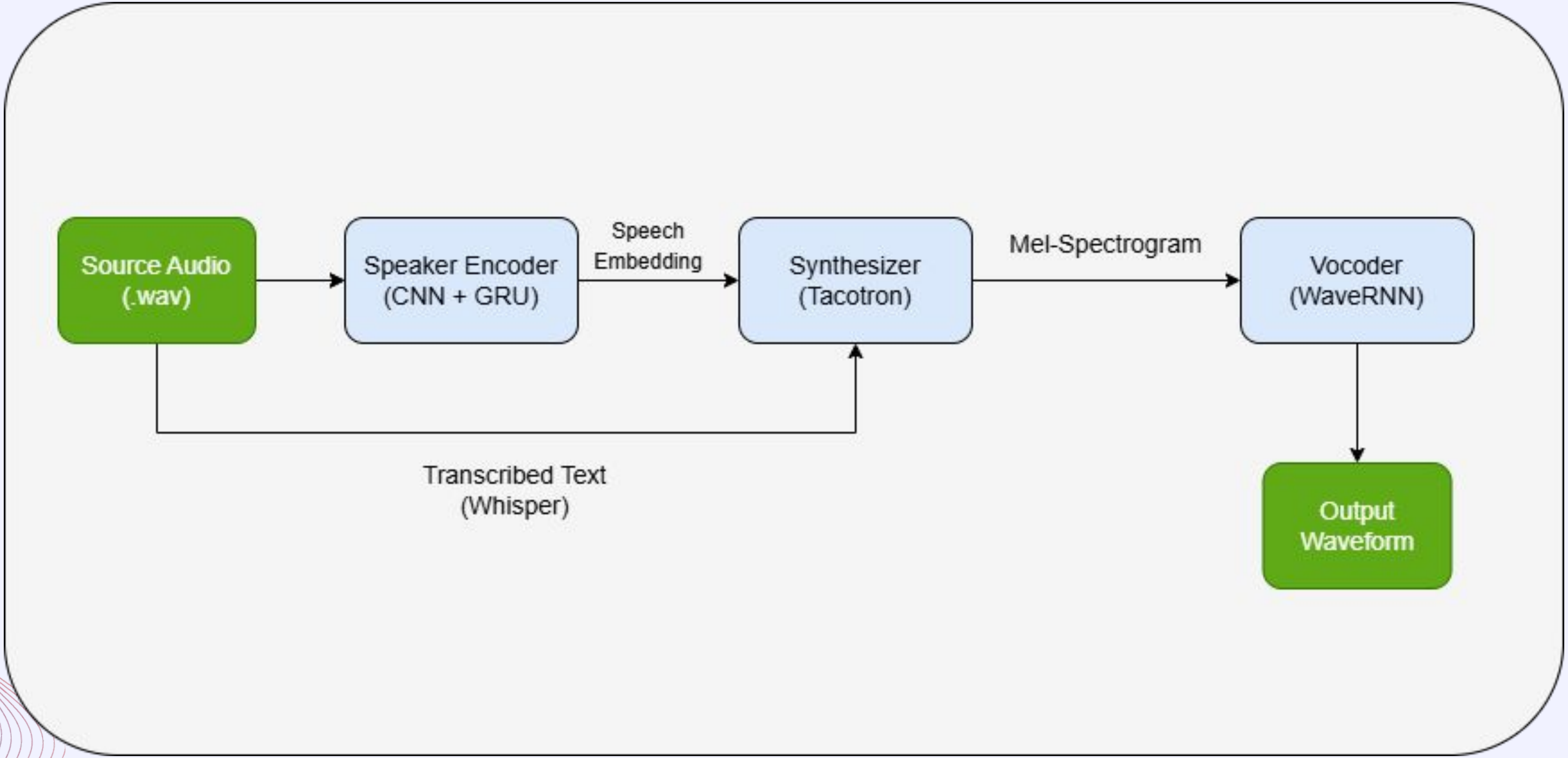
Challenges

- Lack of parallel speech data classified according to accents → No mel-spectrogram for ground truth
- Mel Decoder trained for phoneme embeddings → Expects 256D embeddings → Need to project concatenated embeddings (Content + Accent + Speaker) to expected format
- Training a large model end to end requires a large amount of compute

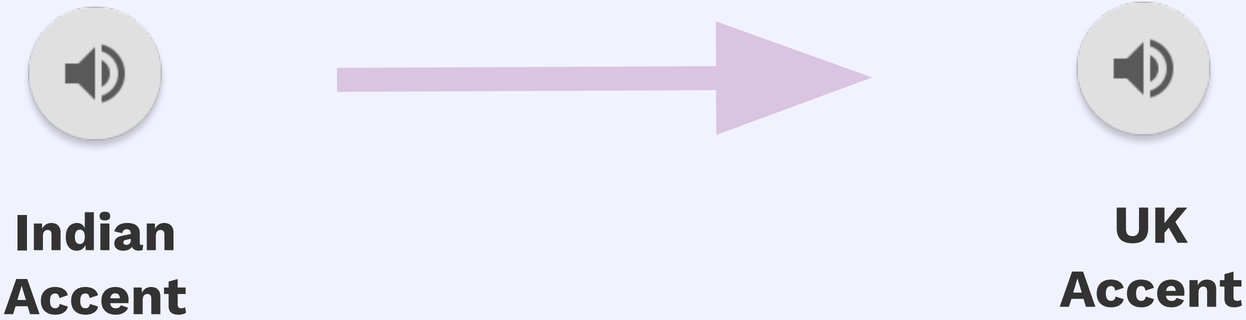
How to solve them in the future?

- Data Synthesis
- Updating the core structure of the mel decoder and training from scratch
- Using Diffusion based architecture for Decoder

Alternate Architecture



Alternate Architecture - Outputs



Indian Accent

UK Accent

Evaluation

Thank You!