



Speech Model Personalization: From Research to Production

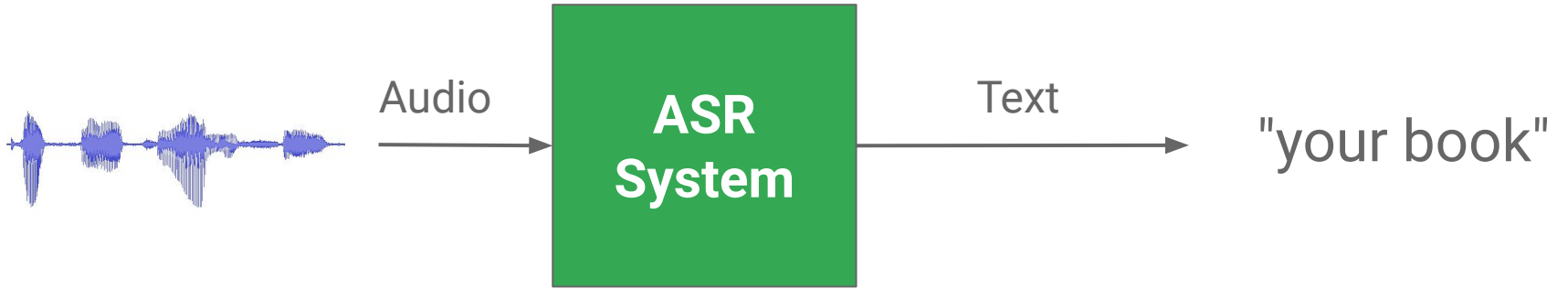
CS6998 class
Columbia University
Mar, 28 2023

Fadi Biadsy
Senior Staff Research Scientist, Google Inc.

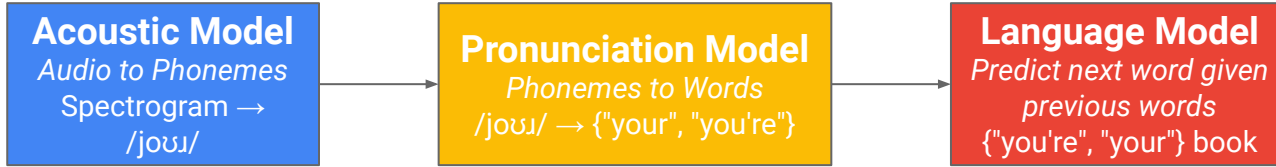
Goal: build a scalable approach for speech model personalization

- Background: Automatic Speech Recognition (ASR) models
- Will walk you through the challenges and proposed solutions to make it possible to personalize models for a large number of speakers
- Introduce **Project Relate**, which enables speakers with atypical speech to personalize their speech models: ASR and conversion
- We discuss research and engineering aspects

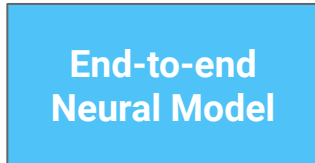
Automatic Speech Recognition (ASR)



"Classic":



End-to-End:

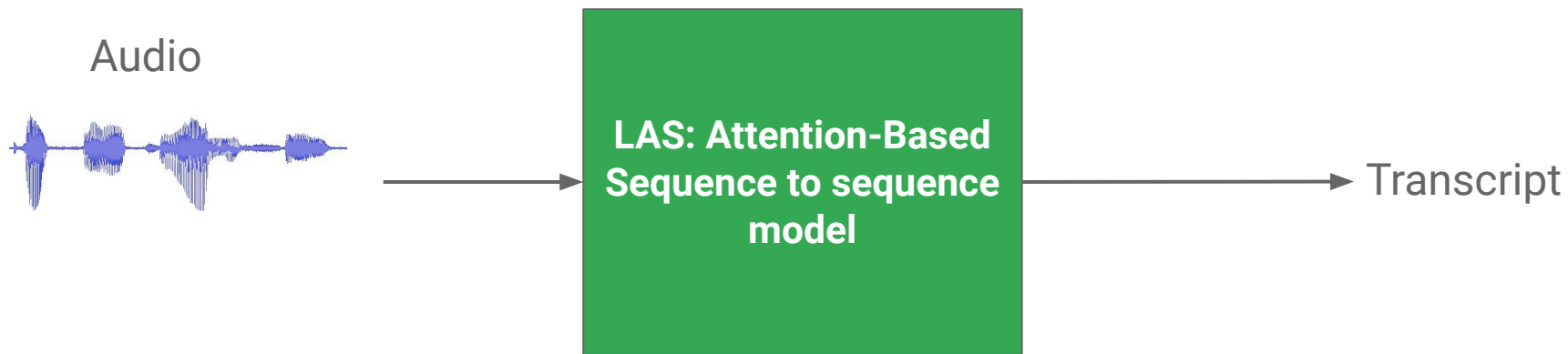


Google

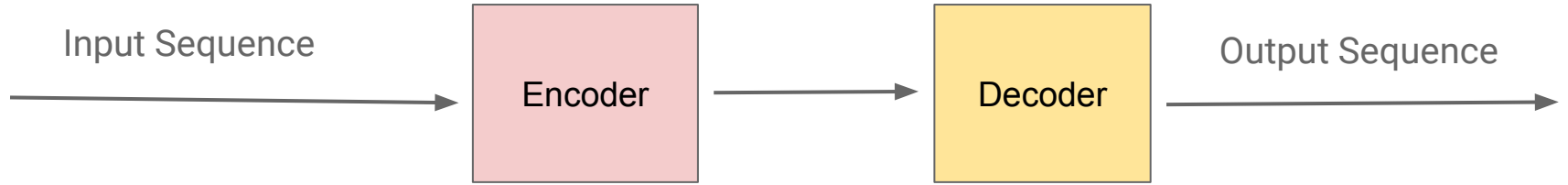
Sequence-to-Sequence Models for ASR

- End-to-end sequence-to-sequence models
 - **CTC**: Connectionist Temporal Classification (Graves+, 06) (Hannun+, 14) (Soltau+, 16)
 - **RNN-T**: Recurrent Neural Network Transducer (Graves, 12) (Rao+, 17)
 - **LAS: Listen, Attend and Spell** (Chorowski+, 15) (Chan+, 15) (Prabhavalkar+, 17) (Kim+, 17)
 - Online Attention-based models (Jaitly+, 16) (Chiu & Raffel, 18)
- Directly map input acoustics into a sequence of graphemes/words
 - Do not require a separate lexicon, decoder graph, separate language models
- Graphemes have been explored as the linguistic units in the past with limited success (Kanthak & Ney, 02) (Killer+, 03)
- Typical lexical units are word pieces (*th- is you- r bo- o- k*)

End-to-End (E2E) ASR model

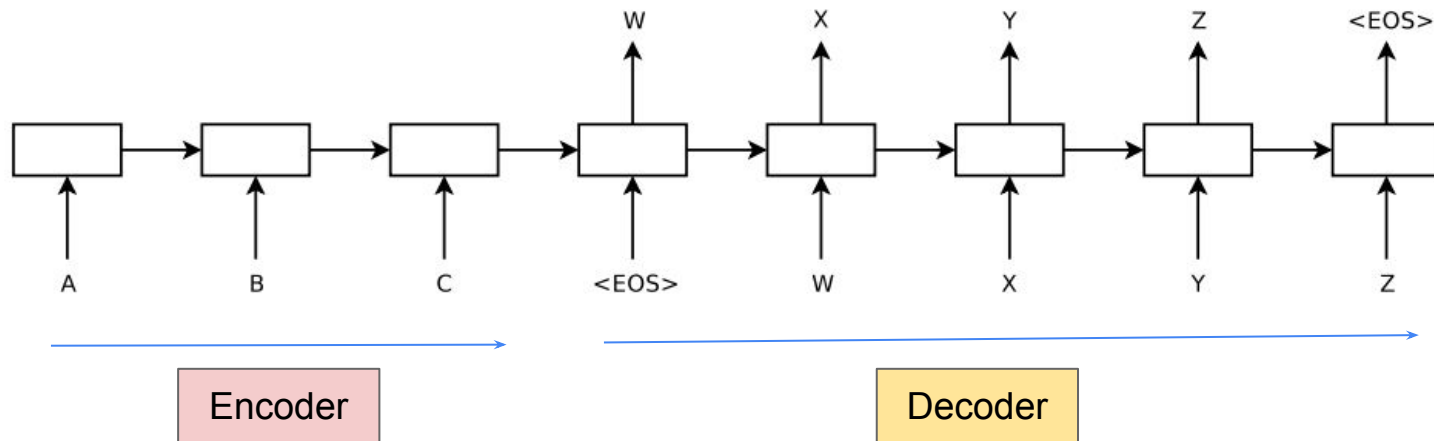


Neural Sequence to Sequence models



Sequence-to-sequence model

Introduced for machine translation (Sutskever, 2014)



On the WMT'14 English to French translation task: BLEU score of 34.81 using an ensemble of 5 deep LSTMs

Best result achieved by direct translation with large neural networks (2014)

BLEU score of SMT baseline on this dataset is 33.30

Neural Sequence to Sequence models with Attention



Attention-based Encoder-Decoder Models

- Emerged first in the context of neural machine translation
- Were first applied to ASR by [Chan et al., 2015] [Chorowski et al., 2015]

Listen, Attend and Spell

William Chan
Carnegie Mellon University
williamchan@cmu.edu

Navdeep Jaitly, Quoc V. Le, Oriol Vinyals
Google Brain
{ndjaitly, qvl, vinyals}@google.com

[Chan et al., 2015]

Attention-Based Models for Speech Recognition

Jan Chorowski
University of Wrocław, Poland
jan.chorowski@ii.uni.wroc.pl

Dzmitry Bahdanau
Jacobs University Bremen, Germany

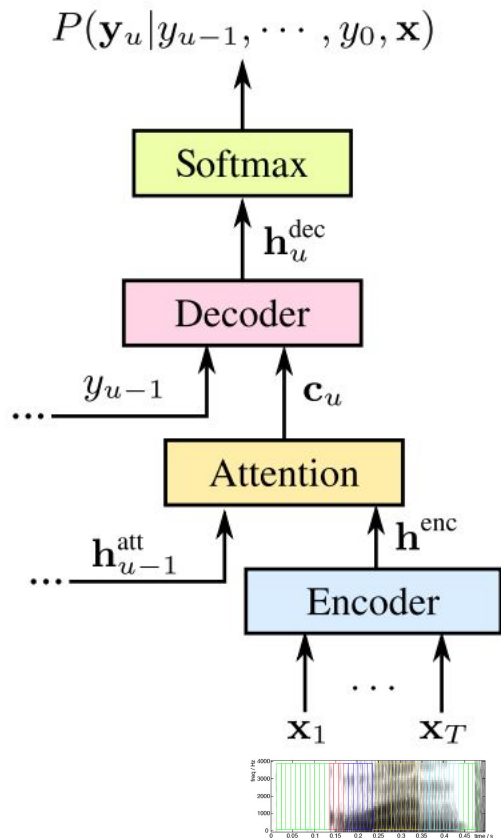
Dmitriy Serdyuk
Université de Montréal

Kyunghyun Cho
Université de Montréal

Yoshua Bengio
Université de Montréal
CIFAR Senior Fellow

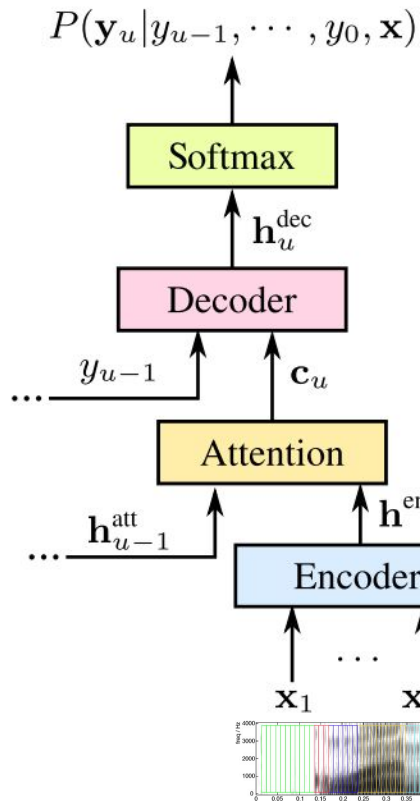
[Chorowski et al., 2015]

Attention-based Models

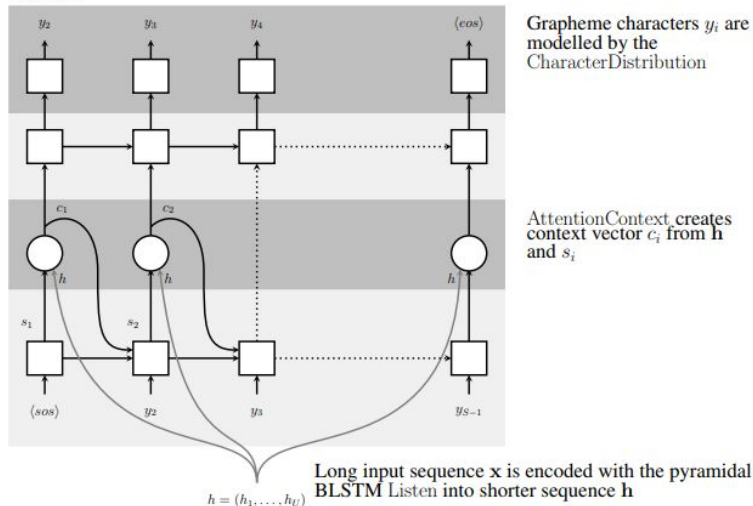


- **Encoder (analogous to AM):**
 - Transforms input speech into higher-level representation
- **Attention (alignment model):**
 - Identifies encoded frames that are relevant to producing next output
- **Decoder (analogous to PM, LM):**
 - Operates autoregressively by predicting each output token as a function of the previous predictions

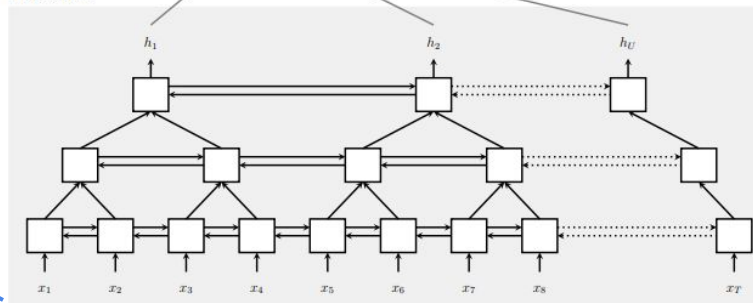
Attention-Based Models



Speller

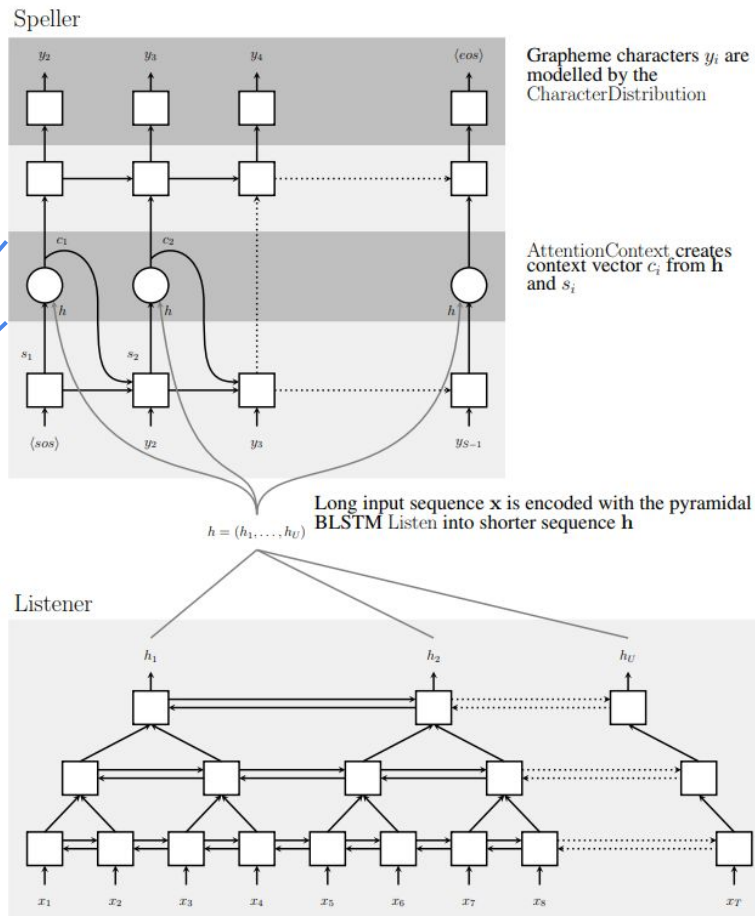
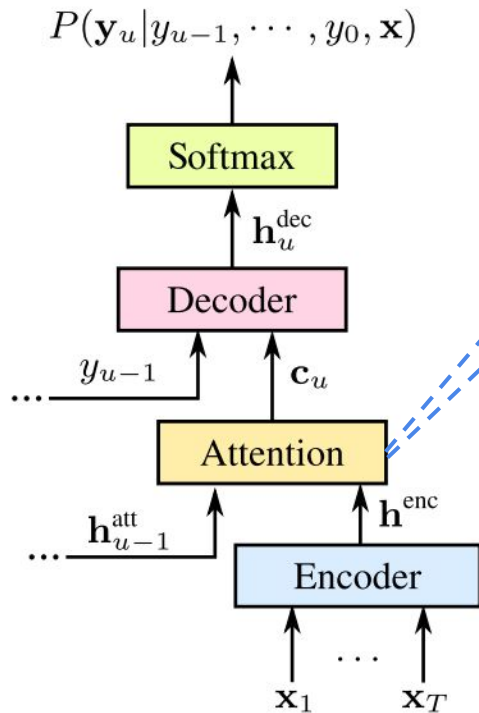


Listener

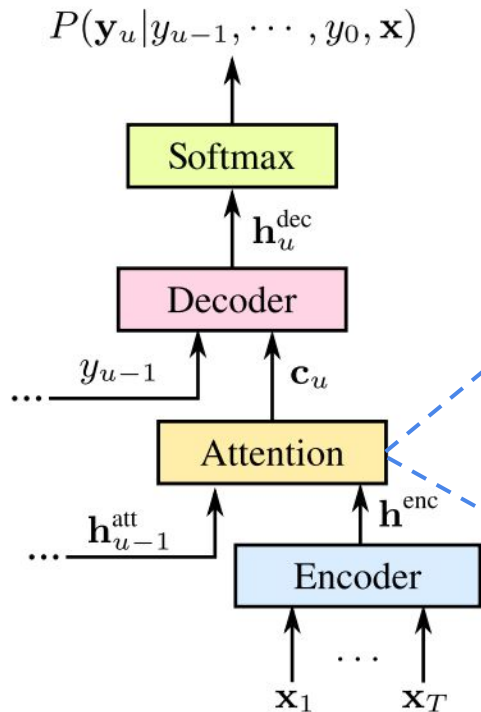


Reproduced from [Chan et al., 2015]

Attention-Based Models



Attention-Based Models



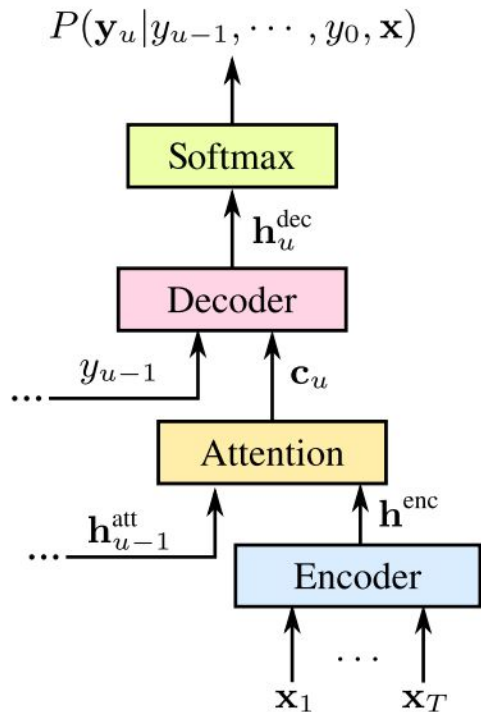
Attention module 'picks' what frames it needs to pay attention to

$$e_{u,t} = \text{score}(\mathbf{h}_{u-1}^{\text{att}}, \mathbf{h}_t^{\text{enc}})$$

$$\alpha_{u,t} = \frac{\exp(e_{u,t})}{\sum_{t'=1}^T \exp(e_{u,t'})}$$

$$\mathbf{c}_u = \sum_{t=1}^T \alpha_{u,t} \mathbf{h}_t^{\text{enc}}$$

Attention-Based Models



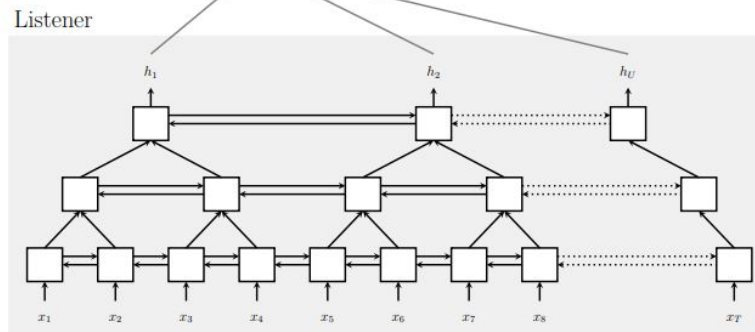
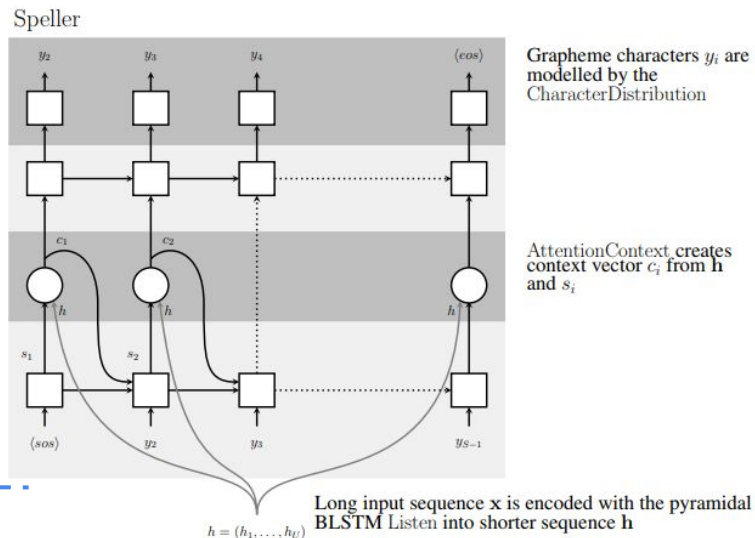
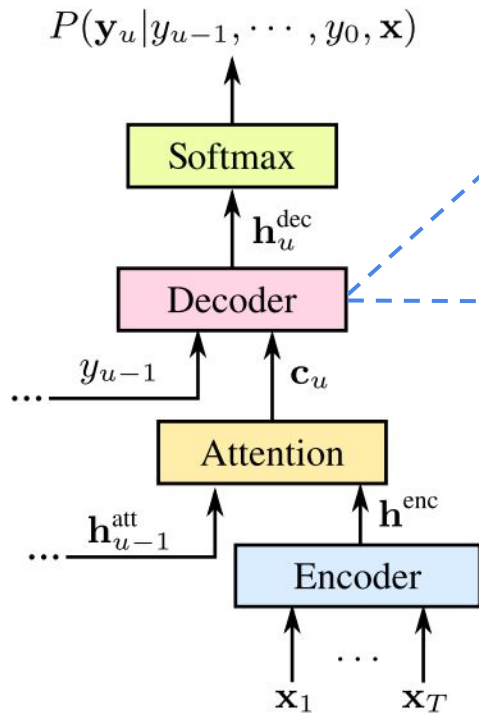
Dot-Product Attention [Chan et al., 2015]

$$e_{u,t} = \left\langle \phi(W\mathbf{h}_{u-1}^{\text{att}}), \psi(V\mathbf{h}_t^{\text{enc}}) \right\rangle$$

Additive Attention [Chorowski et al., 2015]

$$e_{u,t} = w^T \tanh(W\mathbf{h}_{u-1}^{\text{att}} + V\mathbf{h}_t^{\text{enc}} + b)$$

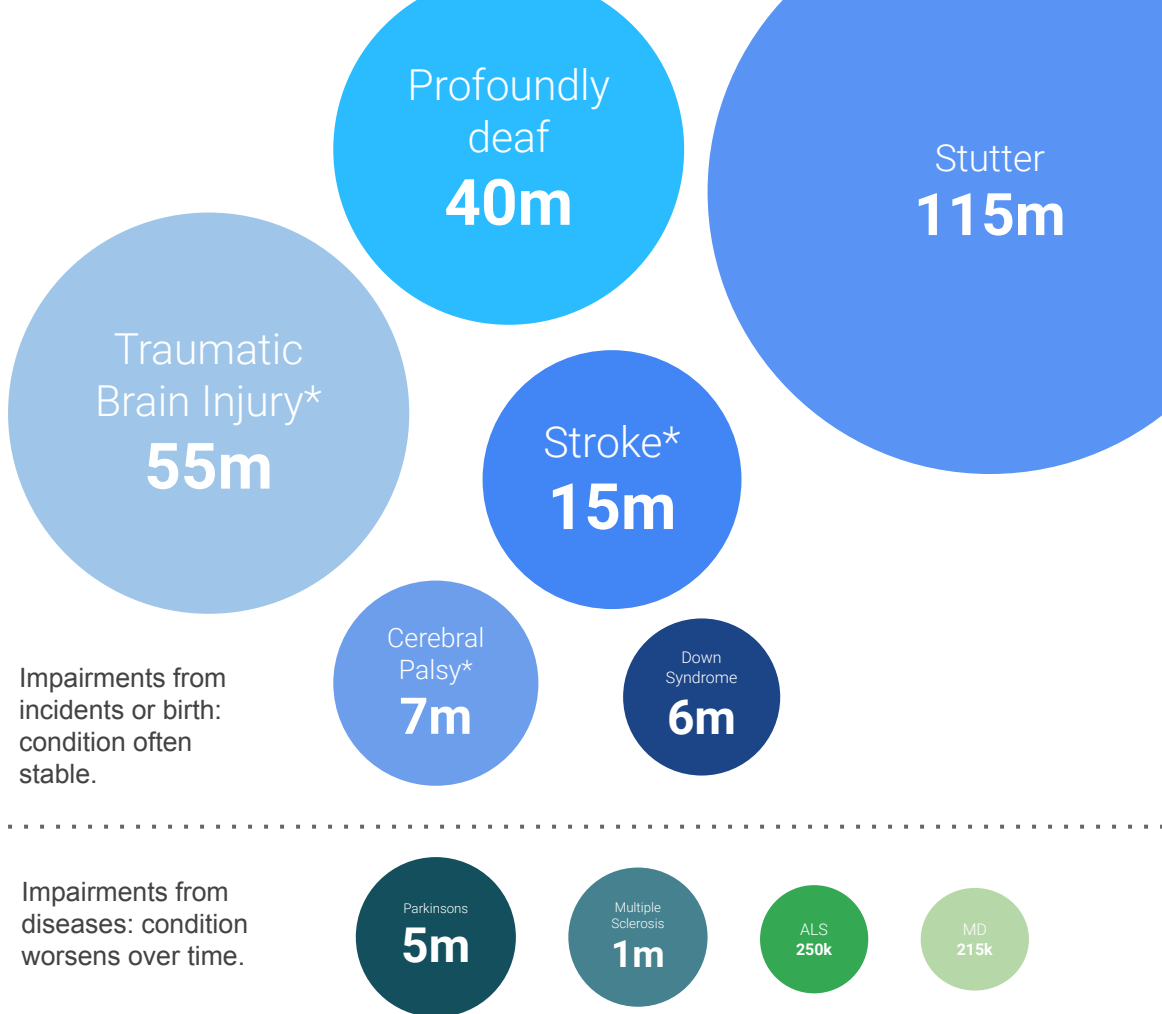
Attention-Based Models



Speech Impairments

~250M+ people worldwide have difficulty speaking and can't rely on speech technology today.

Speech Technology is life changing



**Prevalence of people with neurologic conditions who have speech impairments (estimated, globally)*



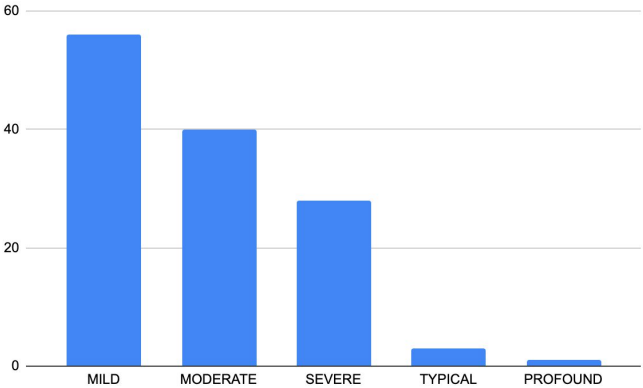
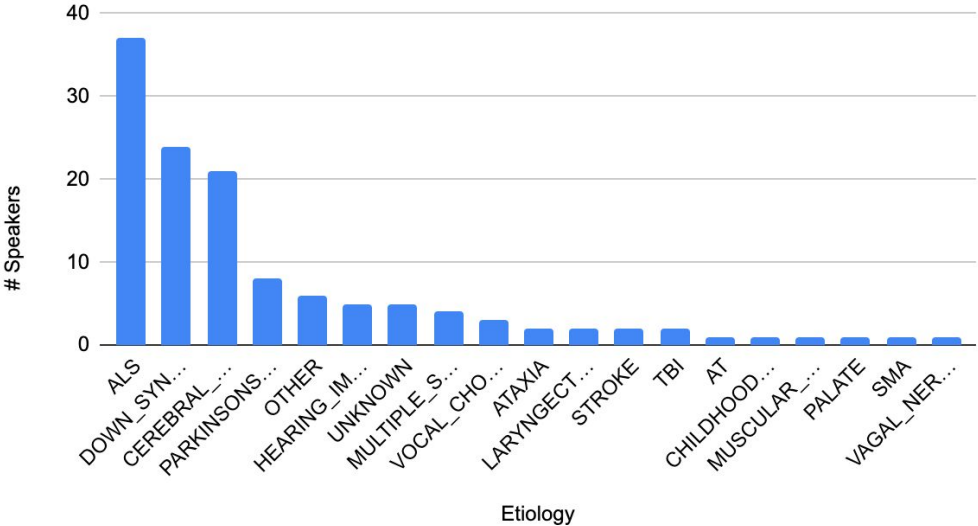
Project Relate

sites.research.google/relate/



Data: A Sample of 128 dysarthric speakers

Speakers vs. Etiology



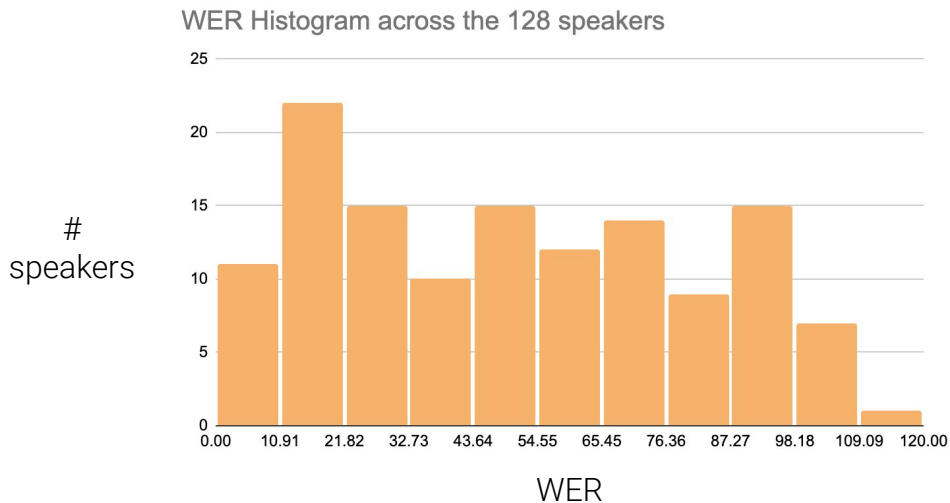
Mean # utterances, per speaker: 2154
Median: 1769
Min: 250; Max: 4250

WER of Google's state-of-the-art ASR engine on these 128 dysarthric speakers

Mean WER: **50.51**

Median WER: **49.5**

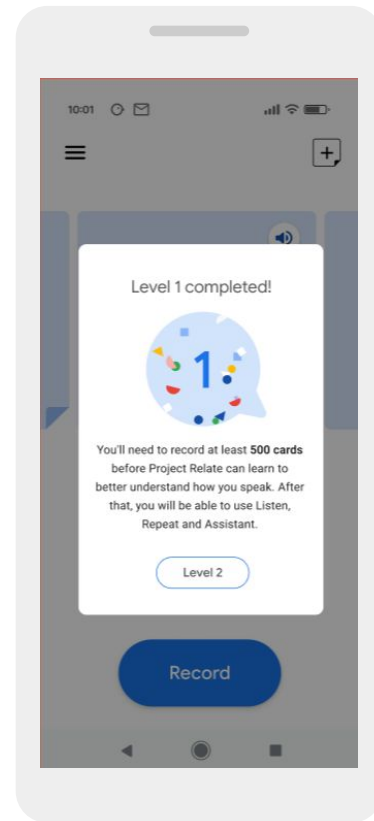
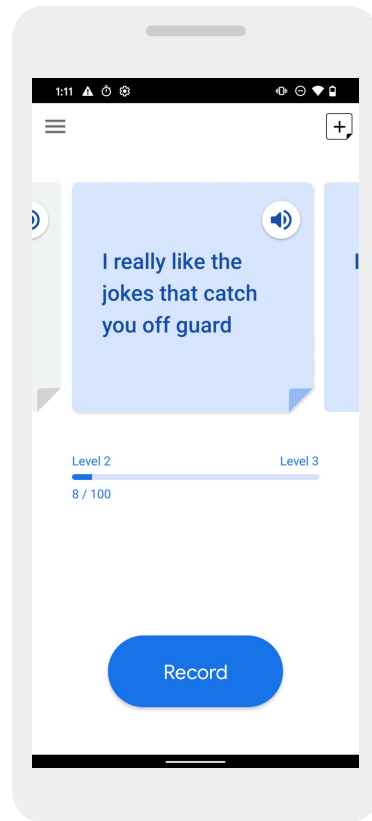
SD: **30.5**





Project Relate

- 1 Users with atypical speech record at least 250 prompted speech samples.
- 2 Obtain two personalized models: ASR model and a speech-to-speech conversion to convert their atypical speech to typical clear speech





Typical vs. Atypical Speech

1. Large variance across etiologies/conditions
2. Large variance across severities
3. Significantly higher variance within etiology and speech idiosyncrasy
4. Some etiologies are degenerative
5. Lack of large and diverse transcribed corpora

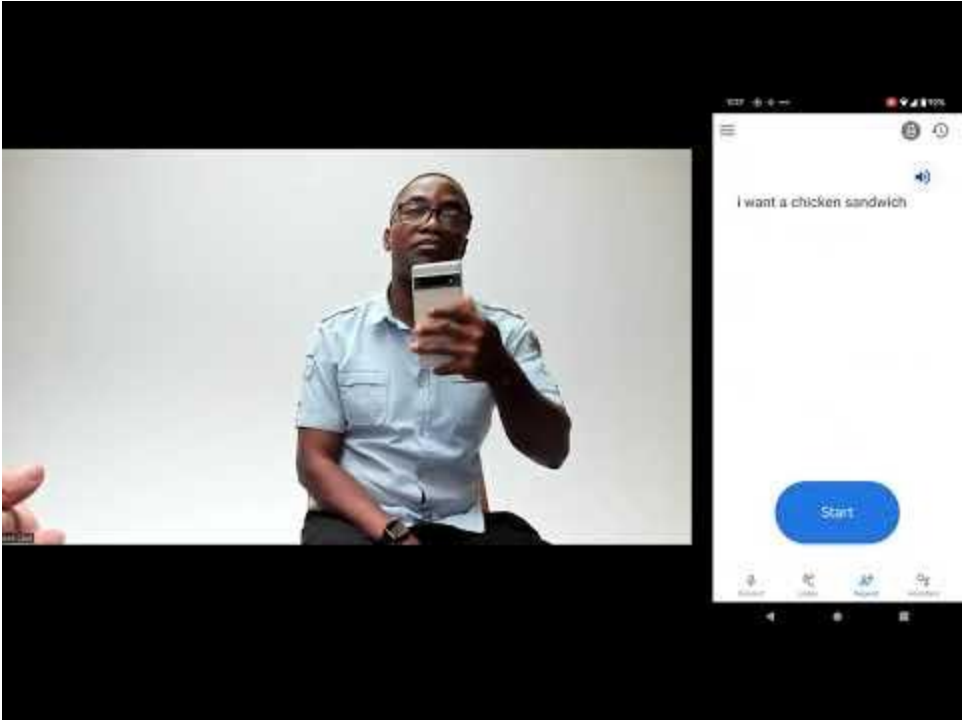
⇒ **Our Conclusion:** Model Personalization seem to be the answer for now

Parrotron Model

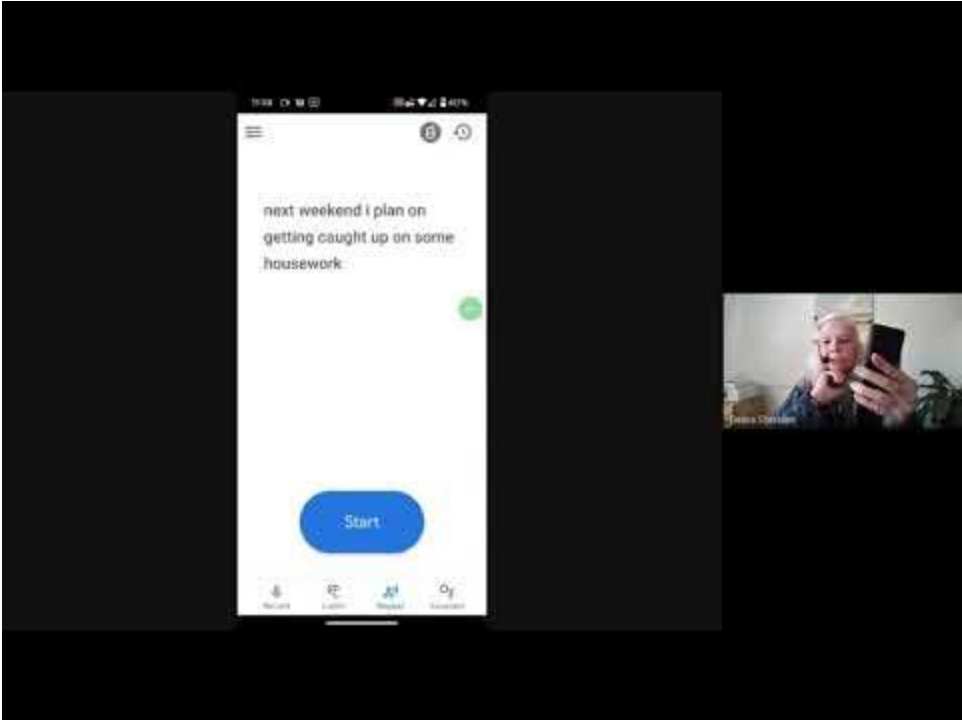
(biadsy et al., 2018)



Stuttering

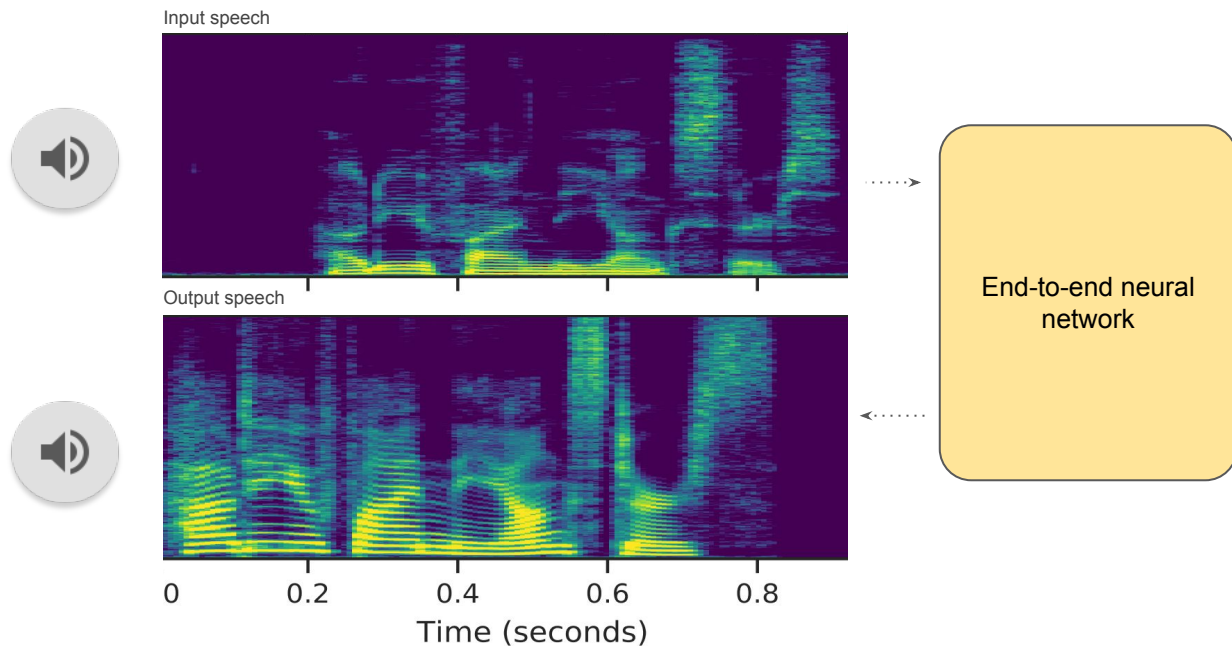


A speaker with laryngectomy using Relate to communicate with Google Home and people



Parrotron - Speech Conversion:

Directly map speech to speech using a single model



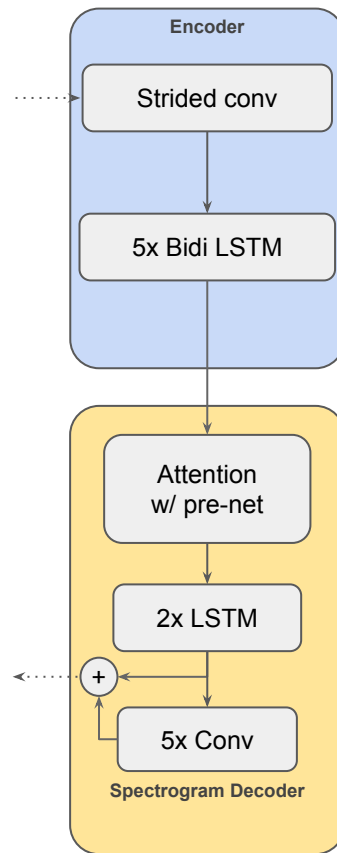
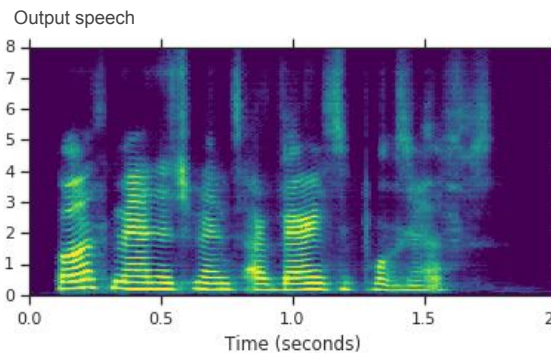
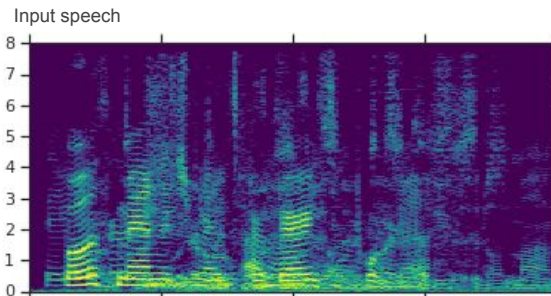
Single E2E neural model for speech conversion

Encoder

Input: log-mel spectrogram

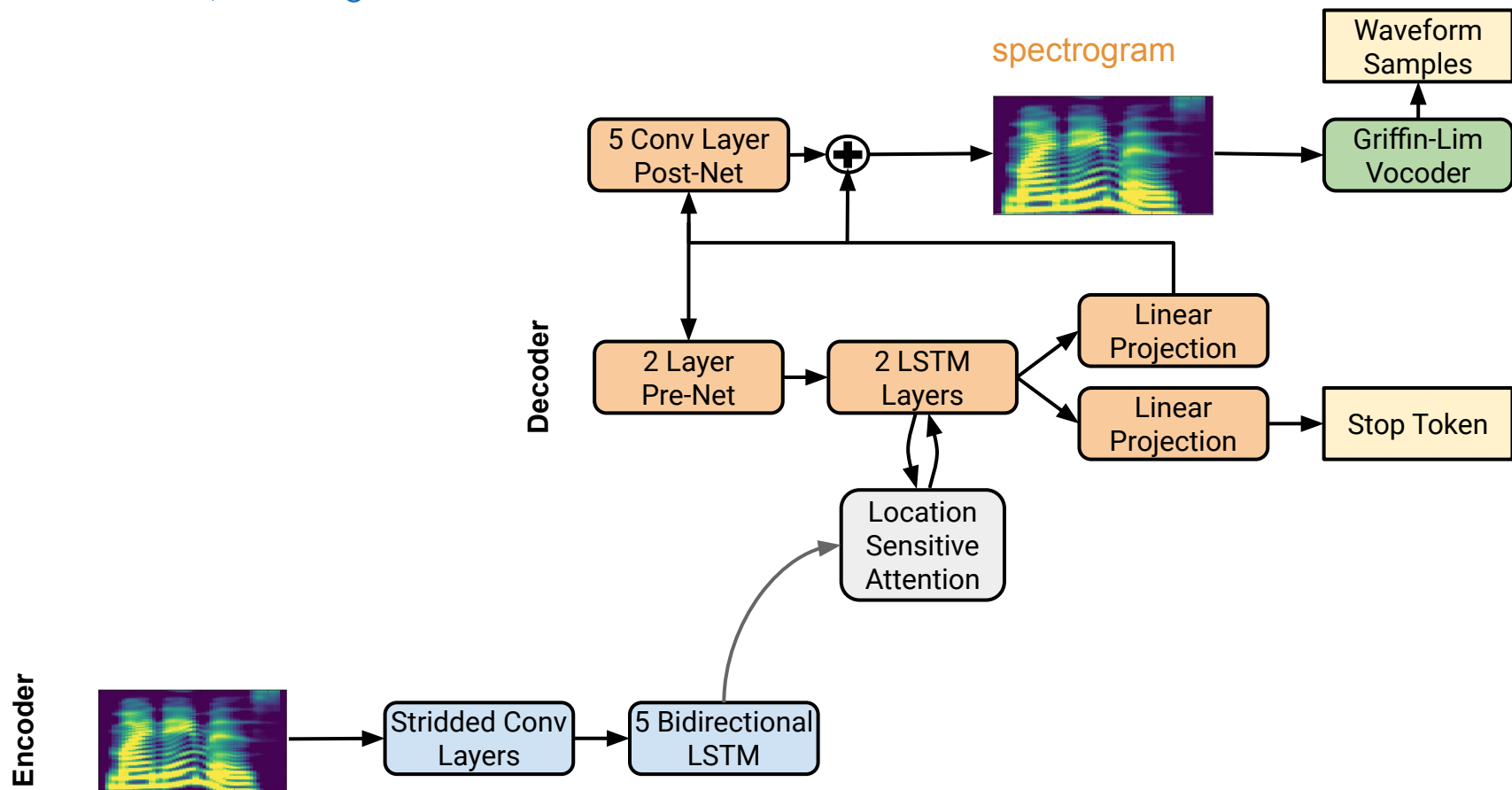
Spectrogram Decoder

LSTM to predict the output spectrogram *one frame at a time* while paying attention to the encoder's RNN states.



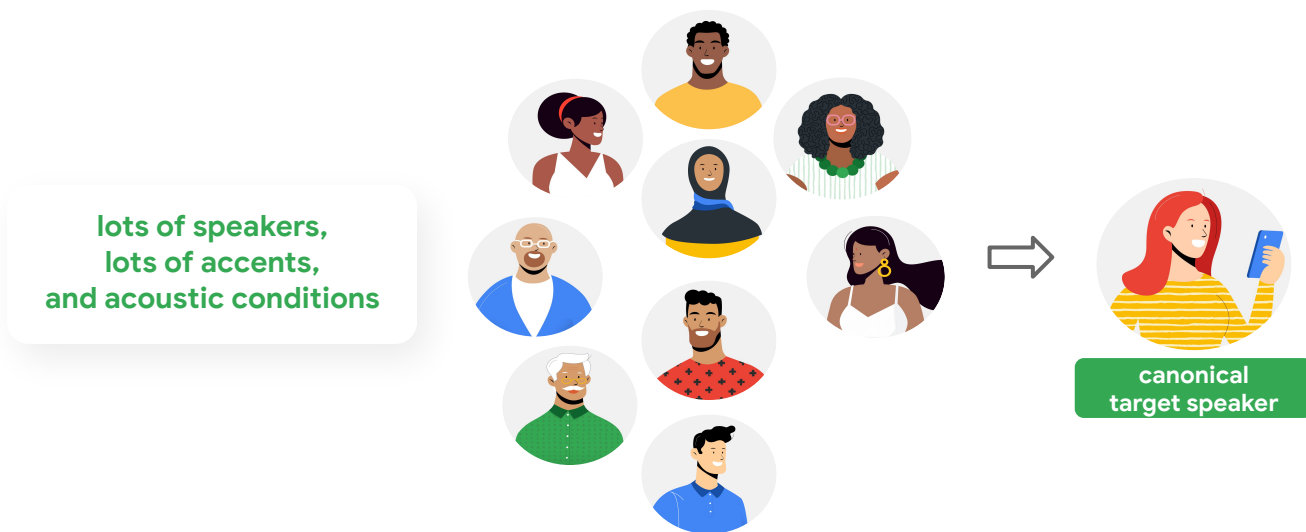
Decoder, follow Tacotron decoder

Encoder, leverage ideas from ASR Encoder



1. Basemodel: Many-to-one conversion

- First build a model that converts/normalizes **anyone's speech to synthesized speech**, preserving only the linguistic content

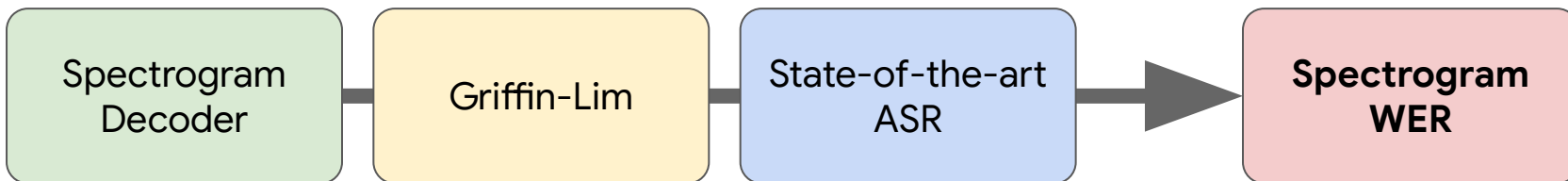


Generate Training data (Parallel Corpus)

1. Use pre-existing manually transcribed speech recognition data (300K hours of speech)
 2. Run Google's TTS on reference transcripts
- ▶ Pairs of utterances: **<Source audio, Target TTS>**

Model Evaluation

Evaluating decoder outputs



Voice normalization eval - Many-to-many conversion (v1)

ASR decoder target	#CLSTM	#LSTM	Attention	WER
None	1	3	Additive	27.1

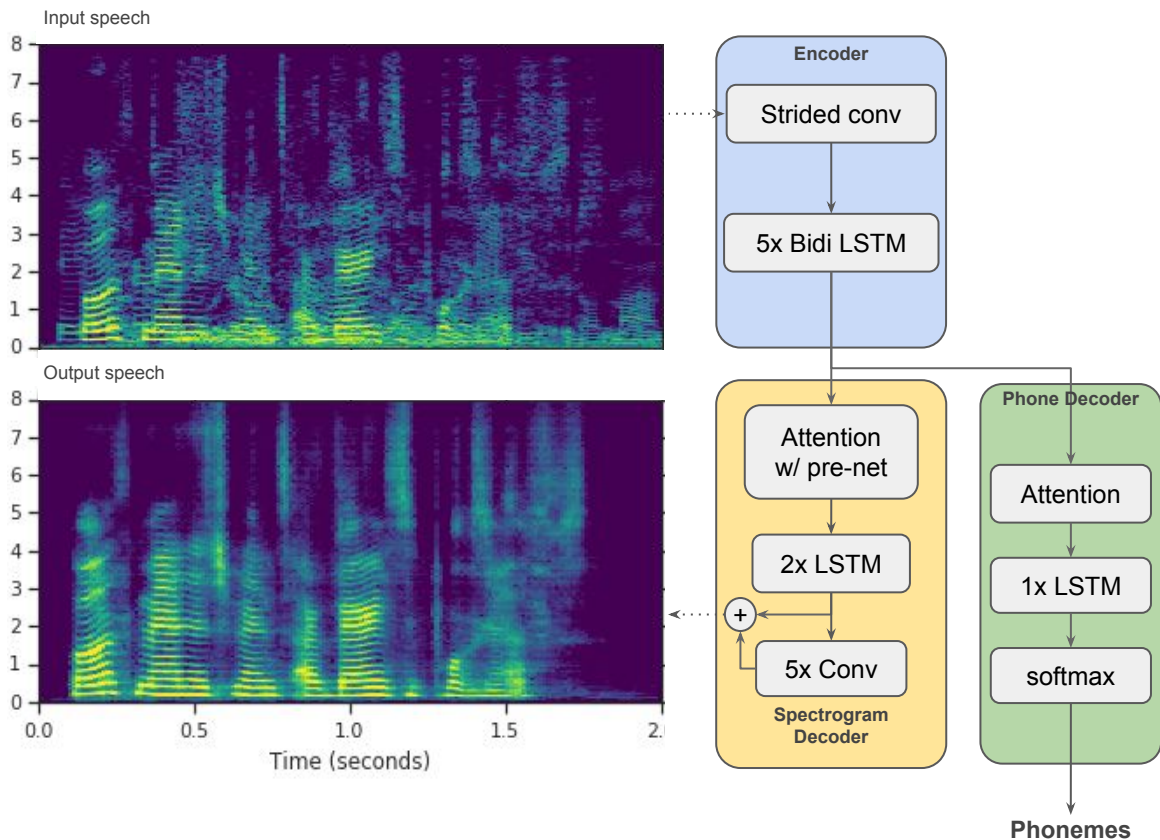


Multi-task training

- Add a phoneme decoder to stabilize the trainer
- Encourage the encoder to capture linguistic information

Loss

$Frame L2 + CE(phone) + Logit(EOU)$



Speech normalization results

ASR decoder target	#CLSTM	#LSTM	Attention	WER
None	1	3	Additive	27.1
Grapheme	1	3	Location	19.2
Phoneme	1	3	Location	18.5
Phoneme	0	5	Location	18.3
Phoneme w/slow decay	0	5	Location	17.6

- Multi-task training important
 - phoneme targets perform better than graphemes
- Location attention is better than additive



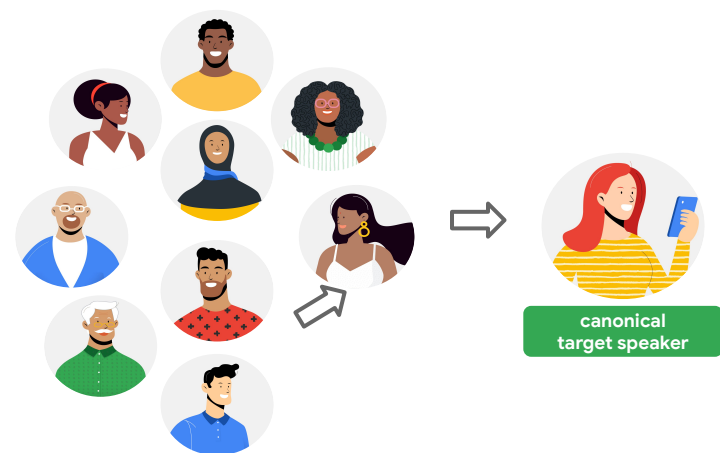
2. Model Adaptation

Adaptation on Dysarthric Speech \Rightarrow *One-to-One Conversion*

Given a few hundreds of manually transcribed adaptation data from a dysarthric speaker

1. Generate a parallel corpus
2. Adapt **all** BaseModel params on this data




► **Highly personalized model for this speaker**






Atypical speech conversion: Results (v1)

- **Adapt** basemodel using 13 hours of transcribed speech



Input  Output  After adaptation 

Input  Output  After adaptation 

Source: <https://google.github.io/tacotron/publications/parrottron>

Model	MOS	WER
Real speech	2.08 ± 0.22	89.2
Parrottron (male)	2.58 ± 0.20	109.3
Parrottron (male) finetuned	3.52 ± 0.14	32.7

36



Substantially improves subjective **naturalness** and **intelligibility**

Conversion and ASR In a single Model

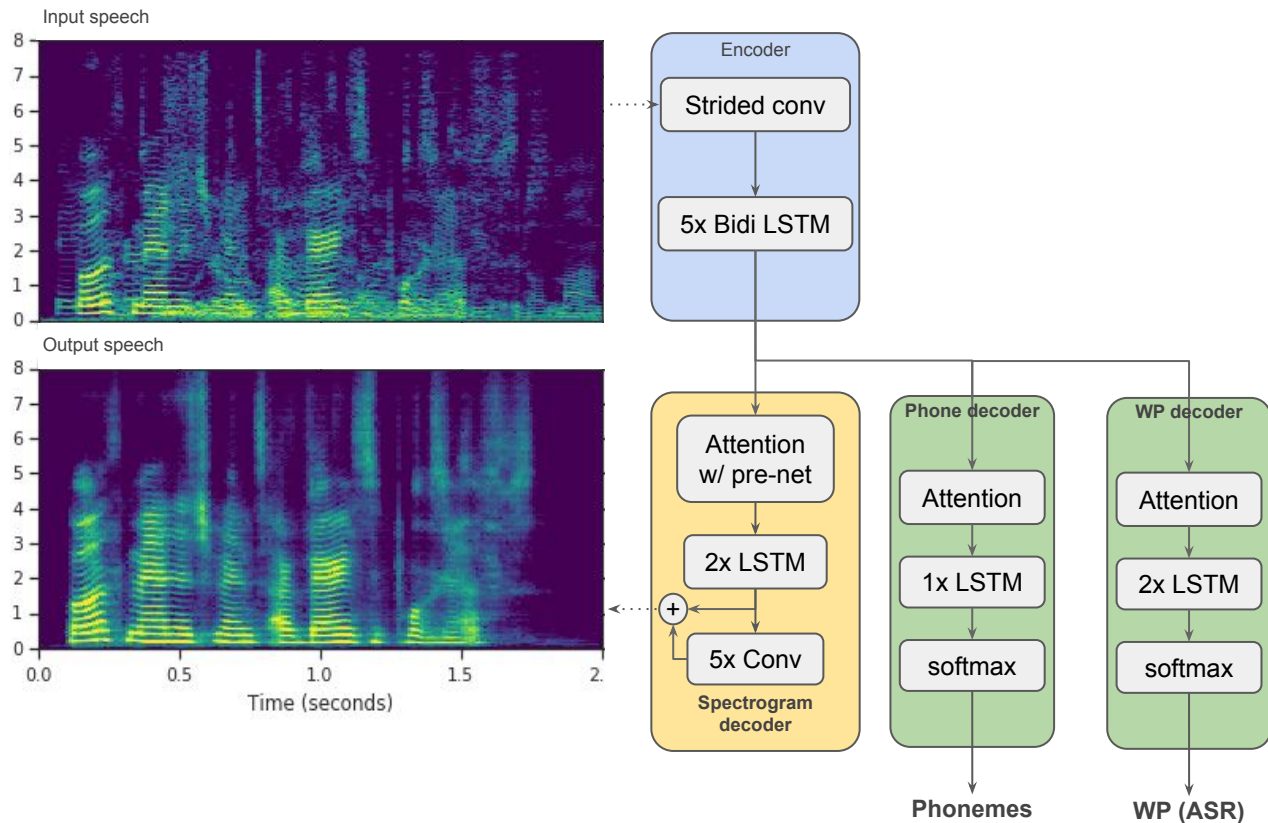


Conversion + ASR

Add additional word-piece decoder to produce words.

Loss

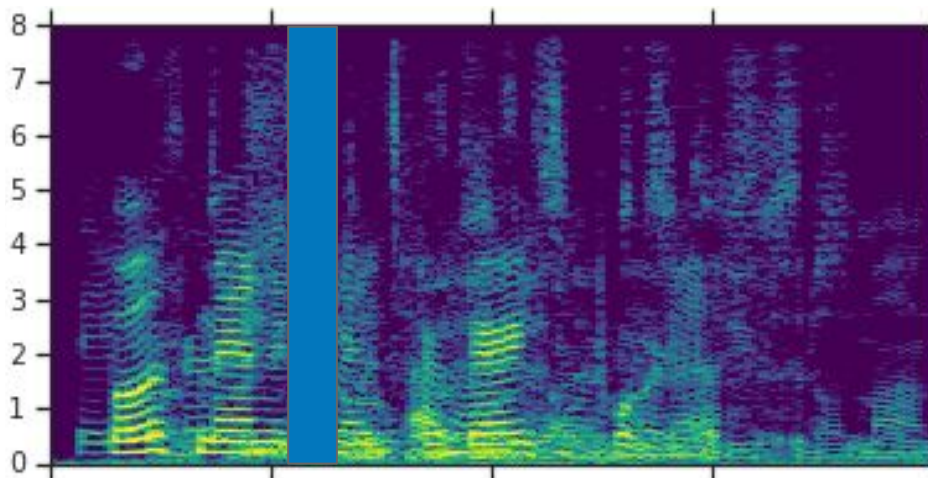
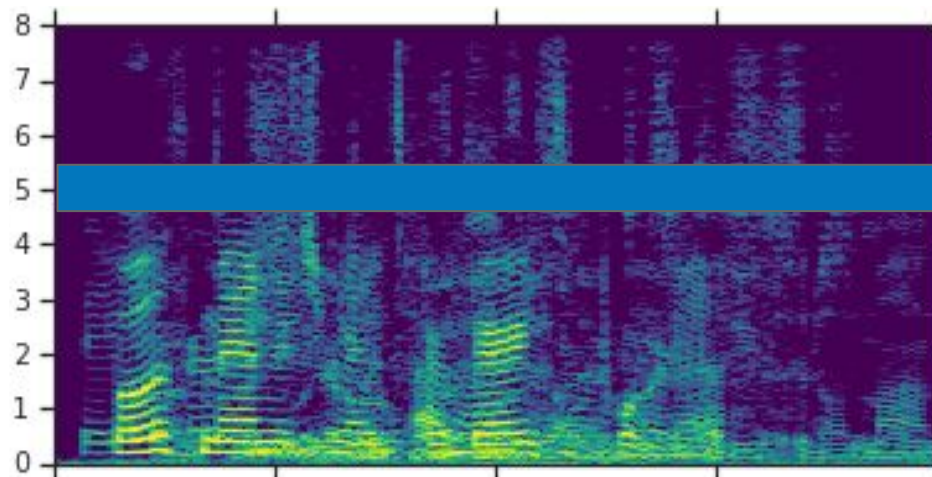
$Frame L2 + CE(phone) + CE(WP) + Logit(EOU)$



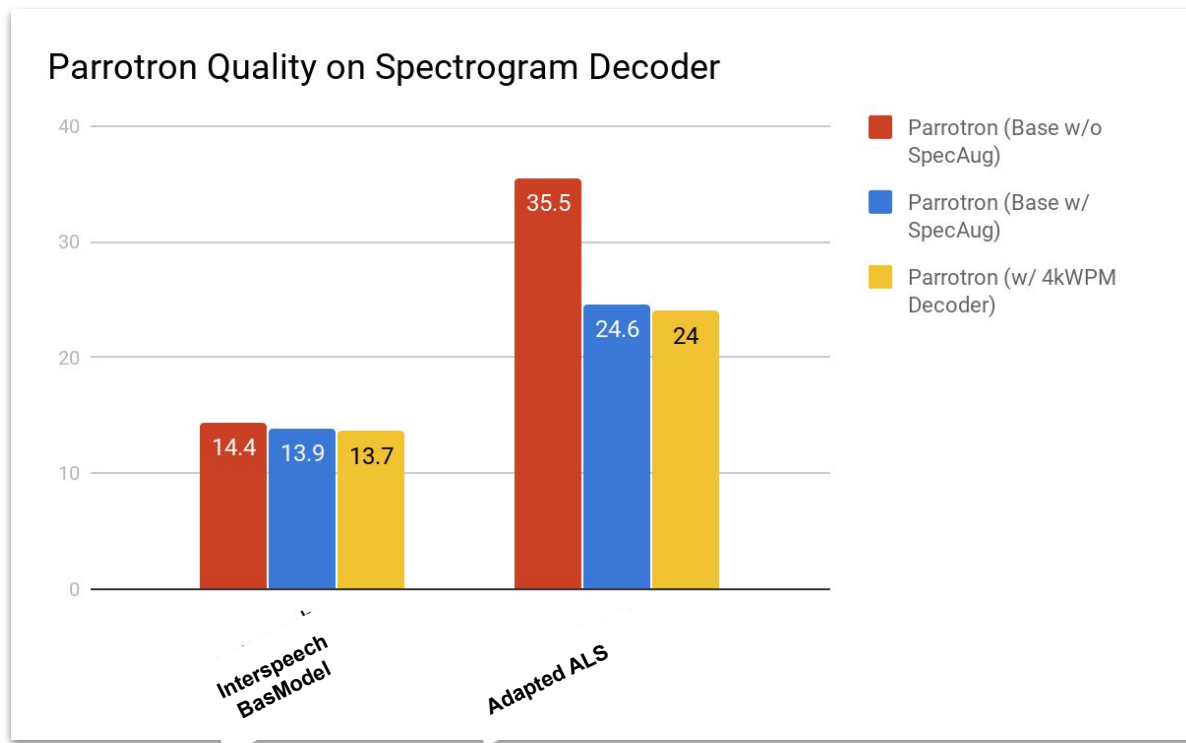
Spec Augmentation

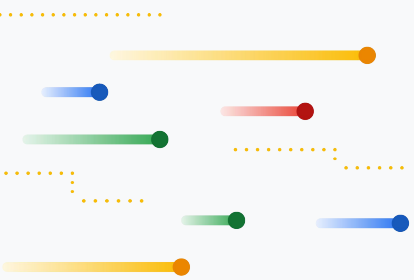
(Park et al, 2019)

- Random up to 27 frequency bands
- 10 random time masks, up to 5% of the utterance length



Word-piece Decoder - “ASR for free”



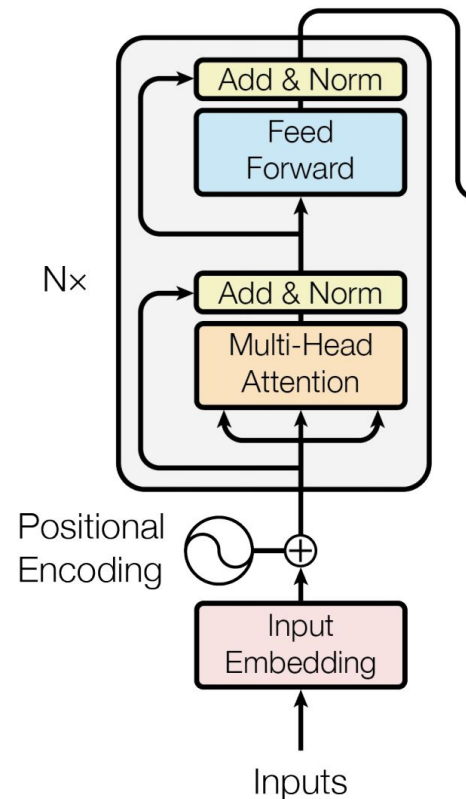


Better Encoder

Transformer:

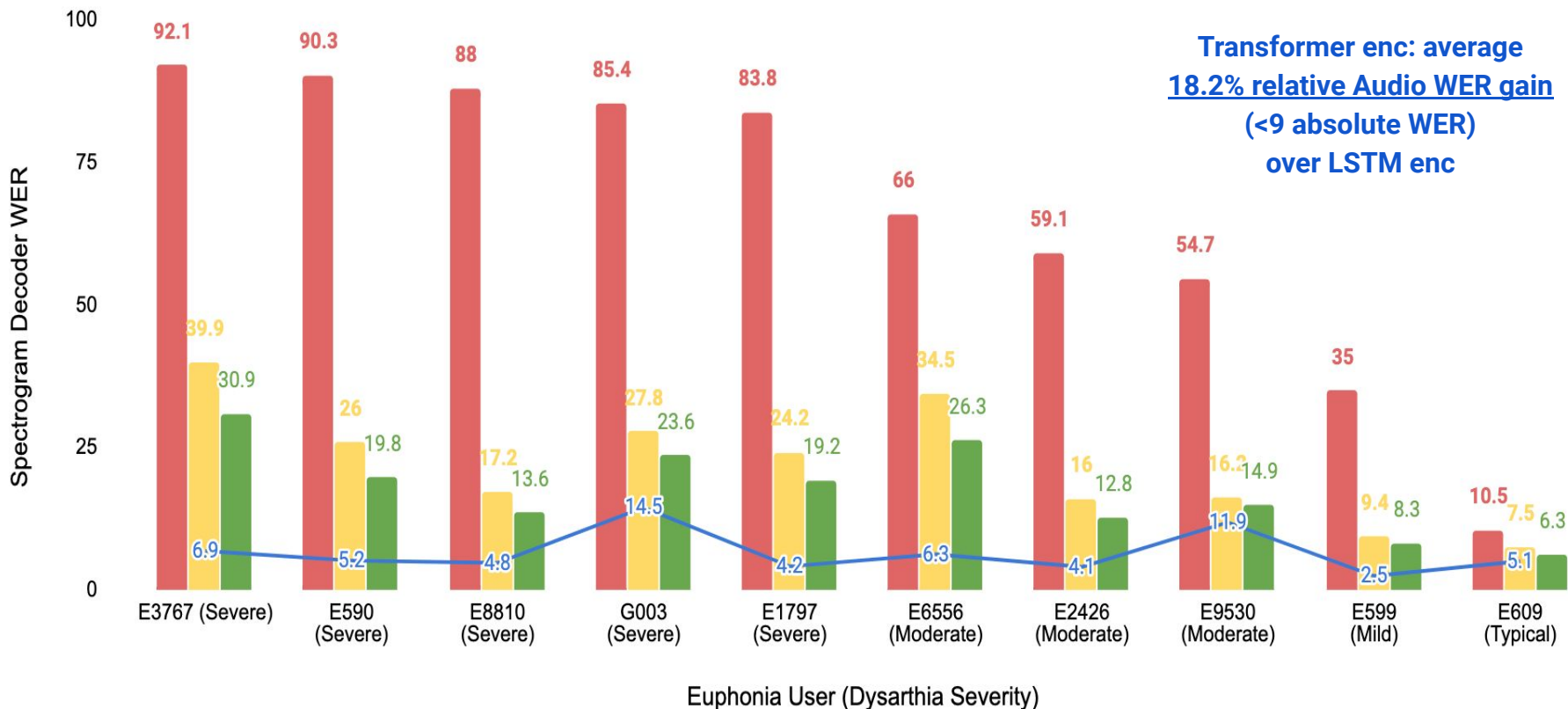
Attention is all you need (Vaswani et al 2017)

- Replace the LSTM encoder by a large transform erencoder
- Why Transformers?
ASR success, speed, long-range dependencies,
GPU/TPU-friendly



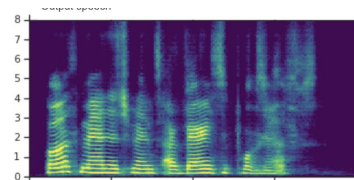
Transformer vs LSTM Encoder (Spectrogram Decoder WER)

Unadapt ASR Adapted LSTM Parrottron Adapted Transformer Parrottron Lower Bound (Reference TTS => ASR)

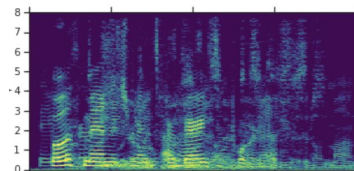
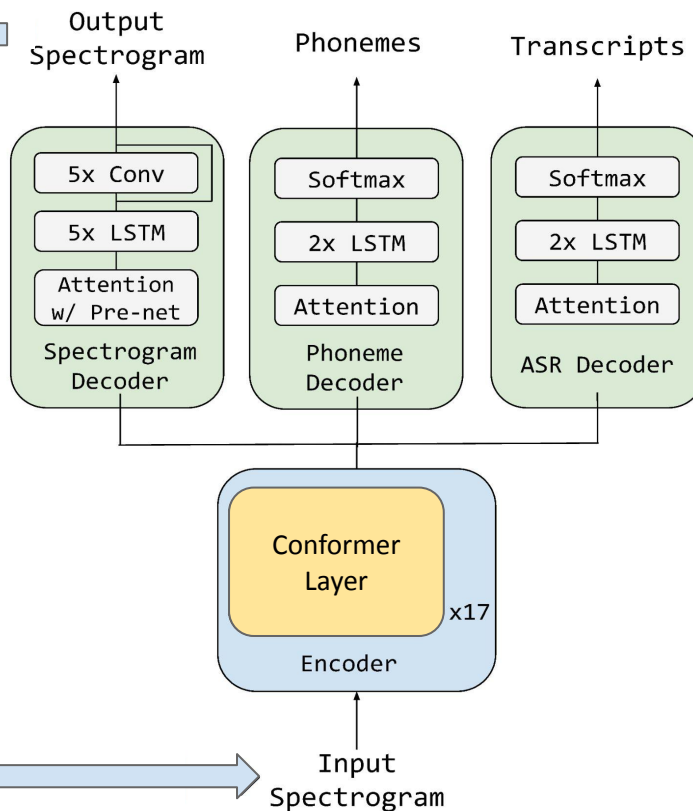


Parrotron Model

(biadisy et al., 2019)

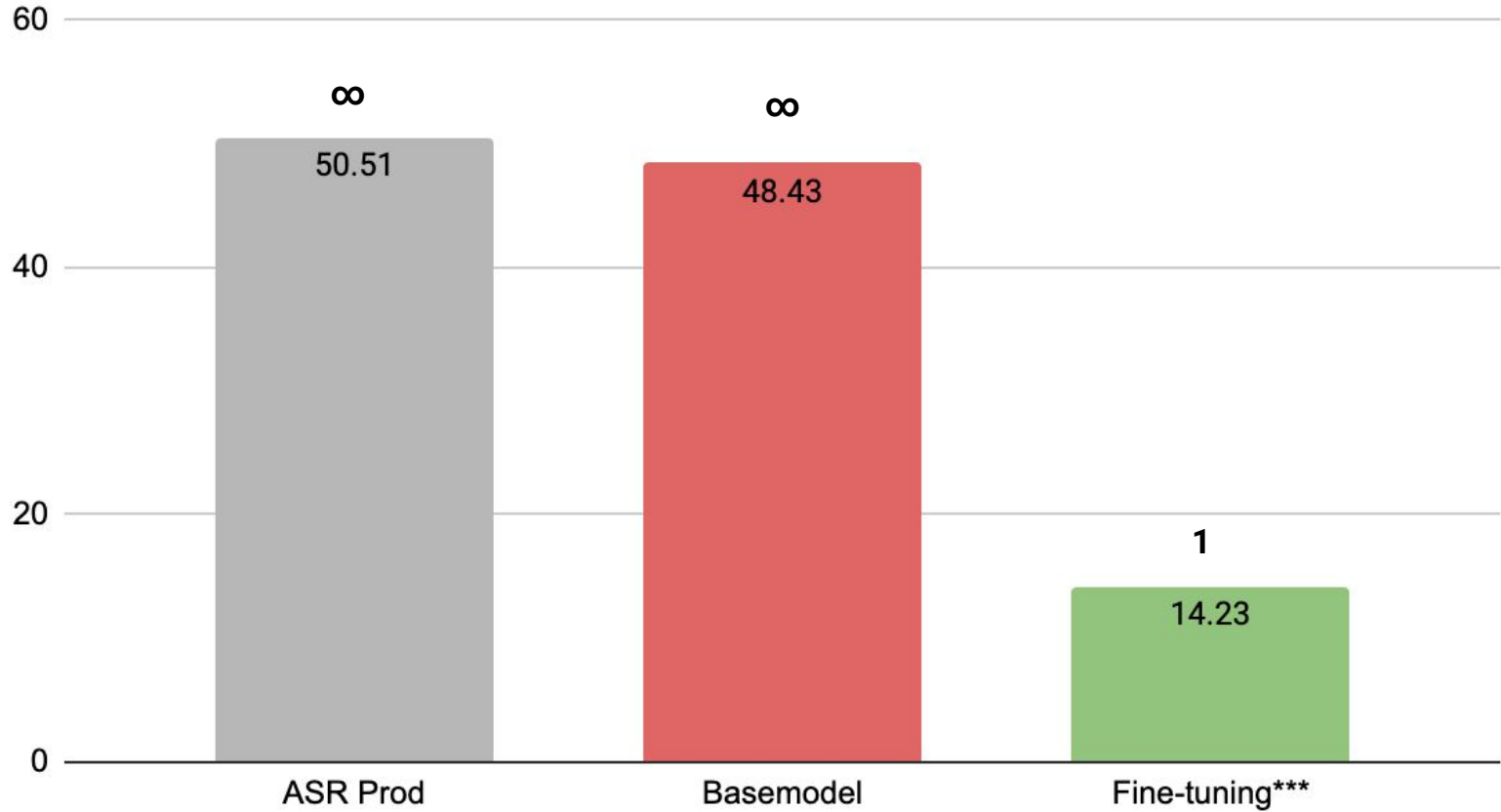


Converted
Typical Speech



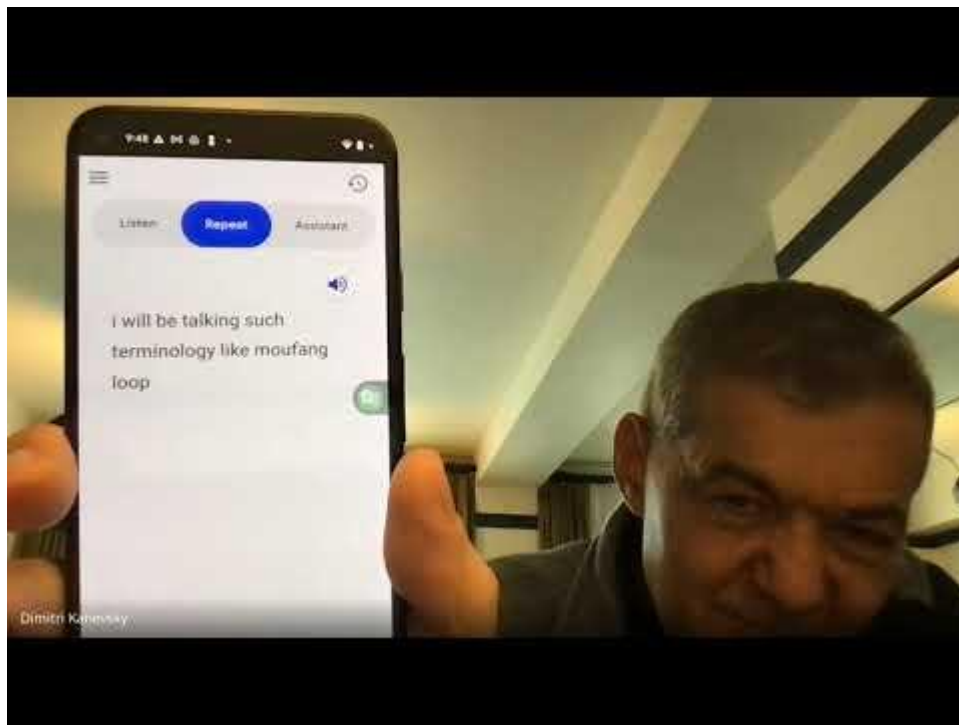
Atypical Speech

Approaches - Mean WER across 128 Speakers



Personalized model for a profoundly deaf speaker (Demo)

+ Semi-Streaming model + Streaming vocoder to speed up inference*



* **Real time spectrogram inversion on mobile phone**, Rybakov et al., 2022

* **Streaming Parrottron for on-device speech-to-speech conversion**, Rybakov and Biadsy 2022

Production Challenges of model Fine-tuning

1. **Storage and memory footprint:** Maintain a large model for each user is expensive and doesn't scale (**668 MB** - 165M params)
2. **Serving (on server):**
 - **Loading Time:** Loading this model per request takes ~15s from remote SSD
 - **RAM:** Impractical to cache all user models and wouldn't scale
3. **Training:**
 - Training a full model per user is infeasible

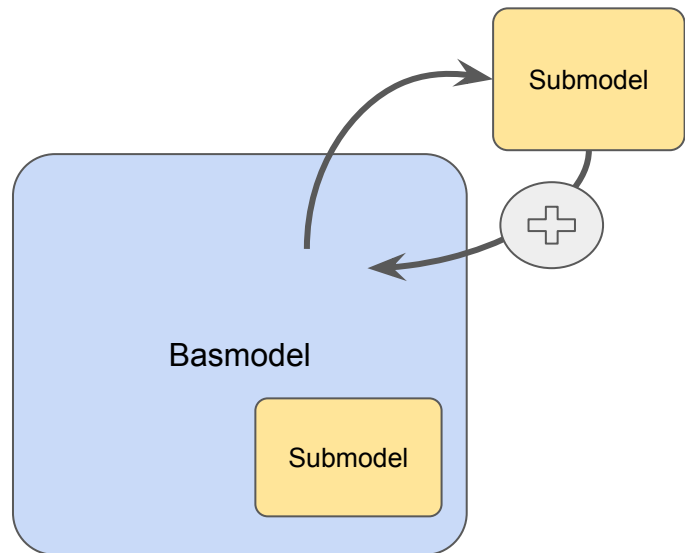


Submodeling

(Biadysy et al., 2022)

A subset of parameters of the *Basemodel*, or extra new parameters added to the *Basemodel* that can be specialized or adapted for a given use-case while freezing the remaining *Basemodel* parameters. To qualify as a Submodel:

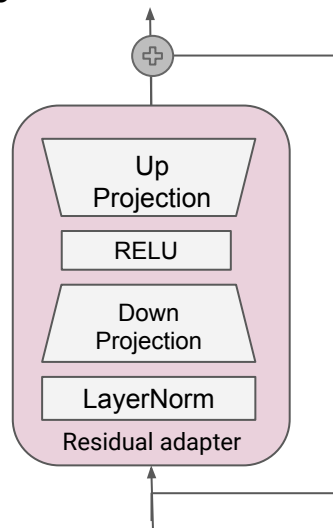
1. **Activation:** Submodels must have an ability to be activated and deactivated dynamically during inference without the need of reloading the Basemodel and re-optimizing the graph.
2. **Size:** A Submodel must be sufficiently small that can be quickly loaded on-demand from storage, and enabled in the Basemodel during inference.
3. **Data Independence:** Submodels ensure that data relevant for a specific use-case can be used independently.



* A scalable model specialization framework for training and inference using submodels and its application to speech model personalization,

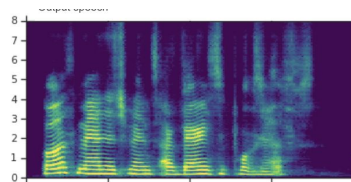
Residual adapters are an efficient choice of Submodel

(for every user) Hounsby et al., 2019

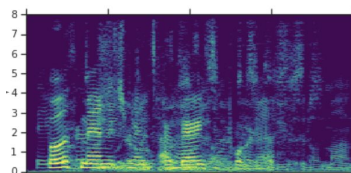
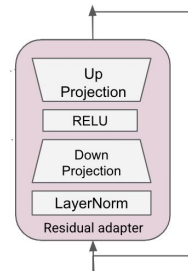
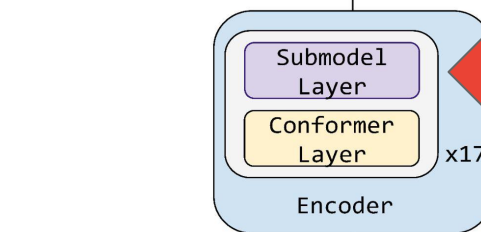
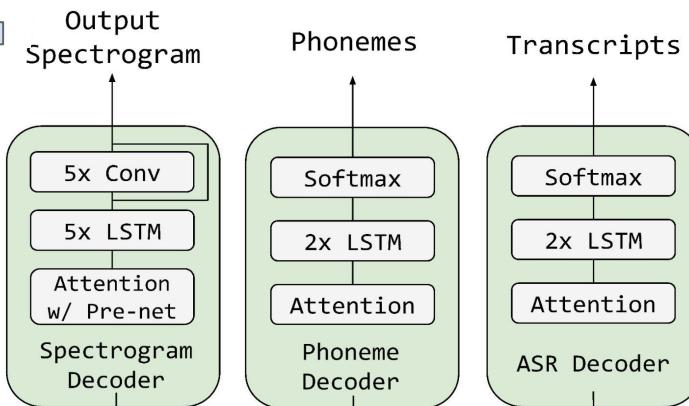


1. **Activation:** Easily set residual factor to 1/0 to activate/deactivate
2. **Size:** Controlled by the down-projection bottleneck dimension
3. **Data Independence:** Each set of adapters (Submodel) can be trained on an independent data

Parrotron Model with Submodel



Converted Typical Speech



Atypical Speech

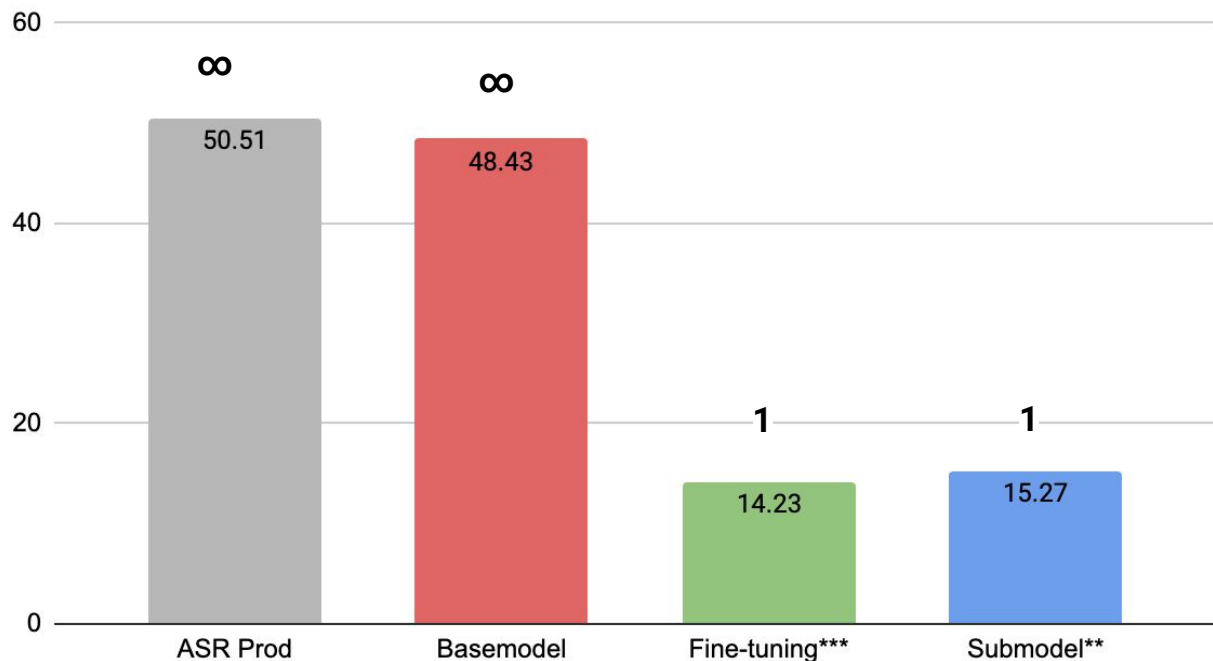
Input Spectrogram

Adapting only the residual adapter Submodel

- *Bottleneck of 64 dim*
- *1.2 Million parameters*
- *Size on disk **4.6 MB***

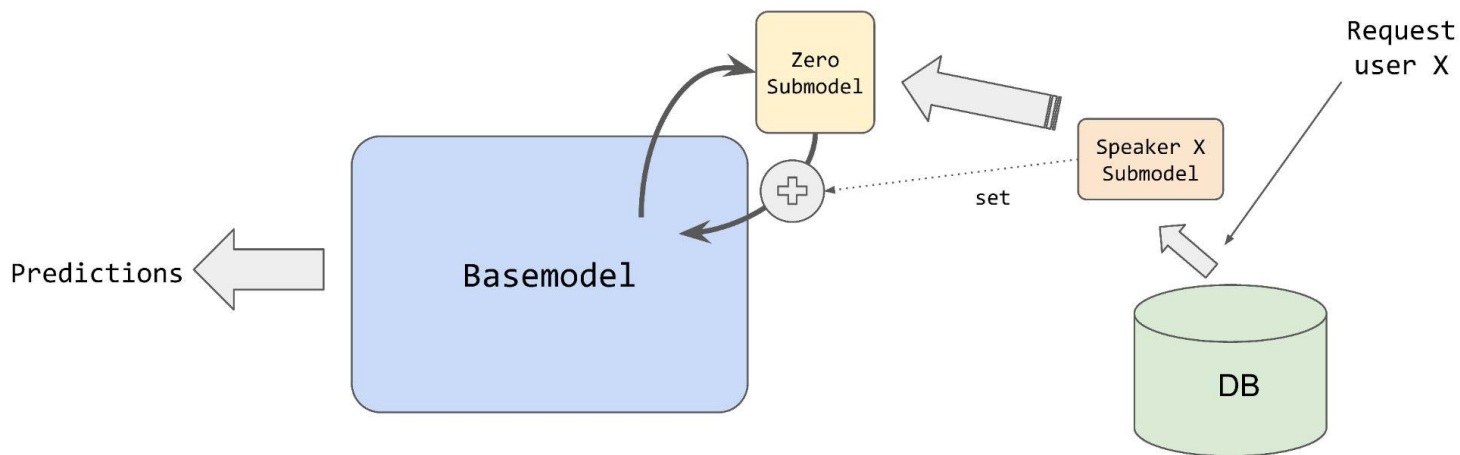
*Solves first problem:
Storage and RAM*

Approaches - Mean WER across 128 Speakers



Scaling inference: Dynamic and on-the-fly loading

- Dynamically load the Submodel from SSD and activate on-the-fly per speaker
- Using remote SSD Submodel takes 18 ± 5.3 ms on average per request (vs. 18.3 seconds \pm 15 for fully fine-tuned model)
- Address second problem: Loading time



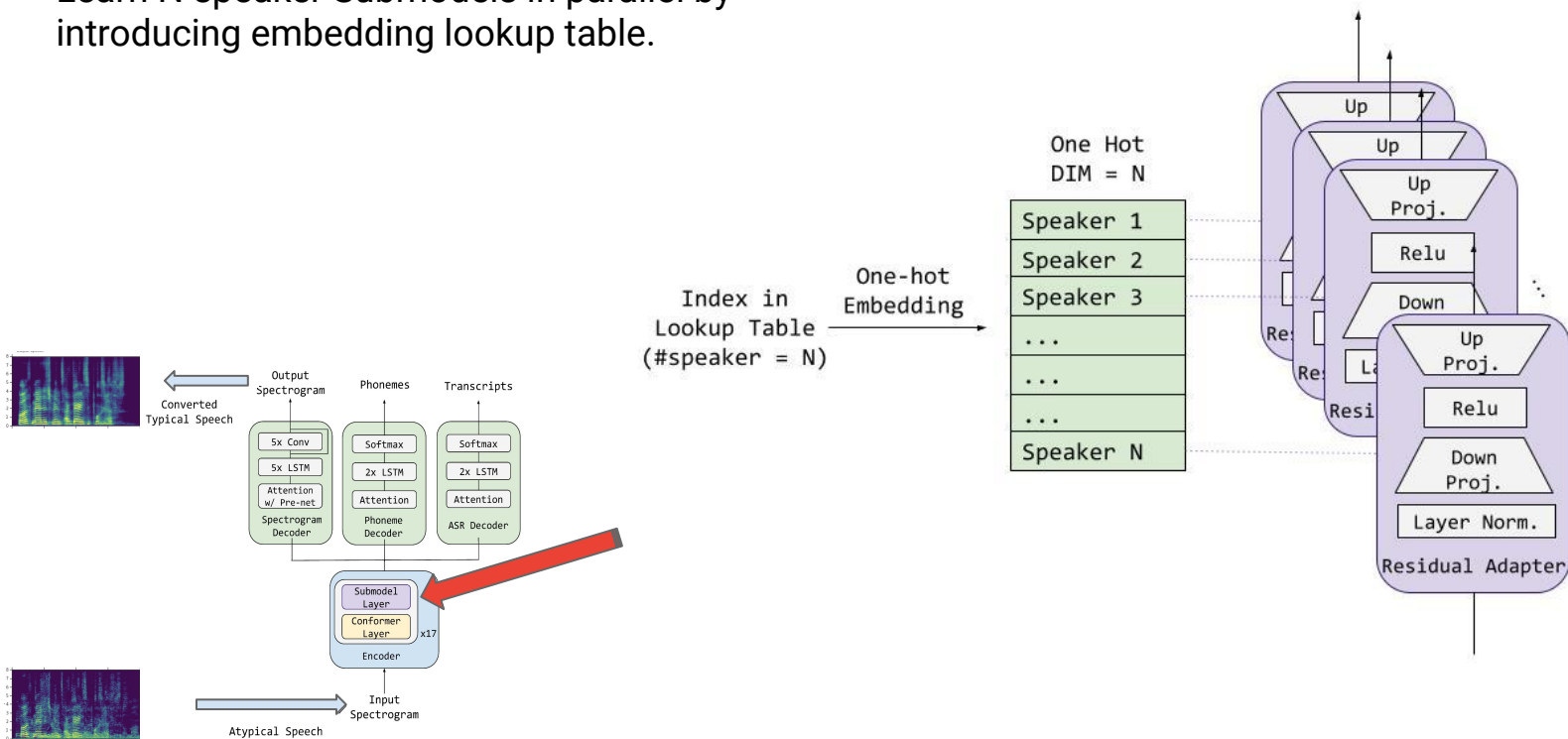
Last problem:

Scale Training:

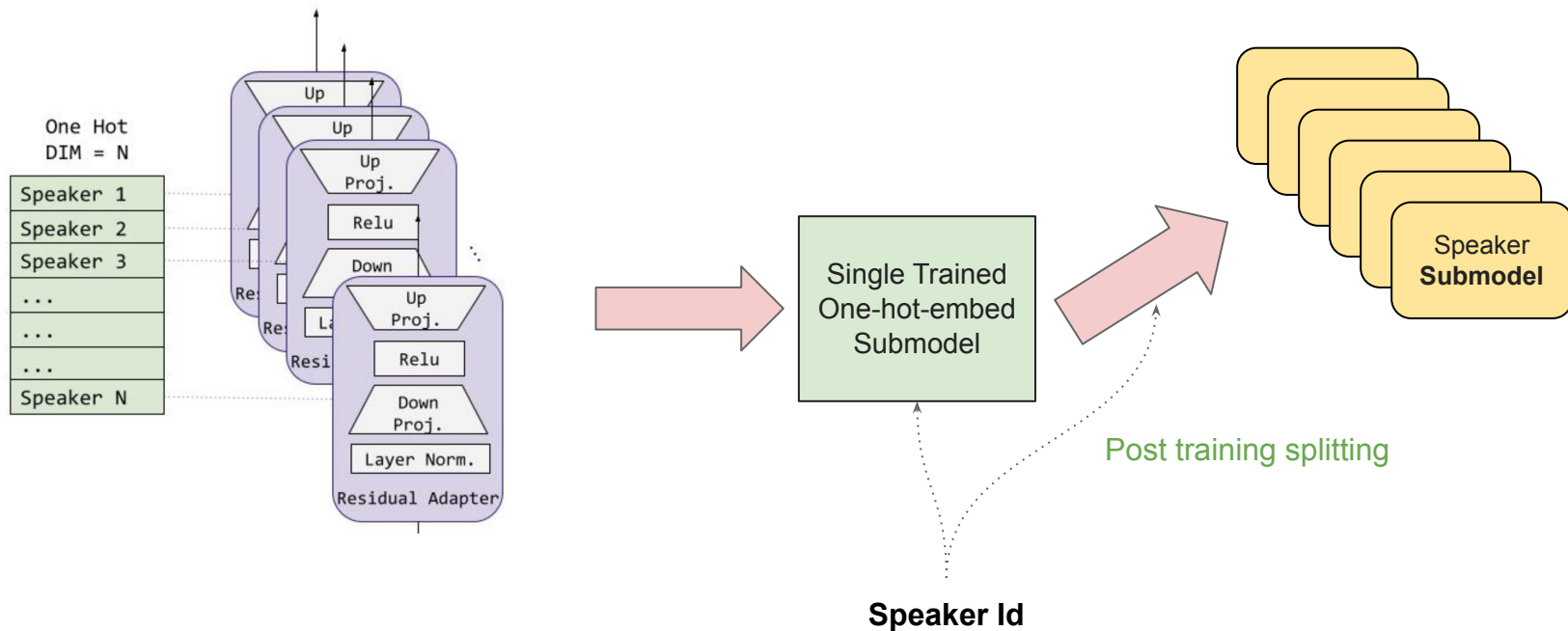
We need to train many Submodels,
one for each user

Scale Training: Introduce One-hot-embedding over adapters

Learn N speaker Submodels in parallel by introducing embedding lookup table.



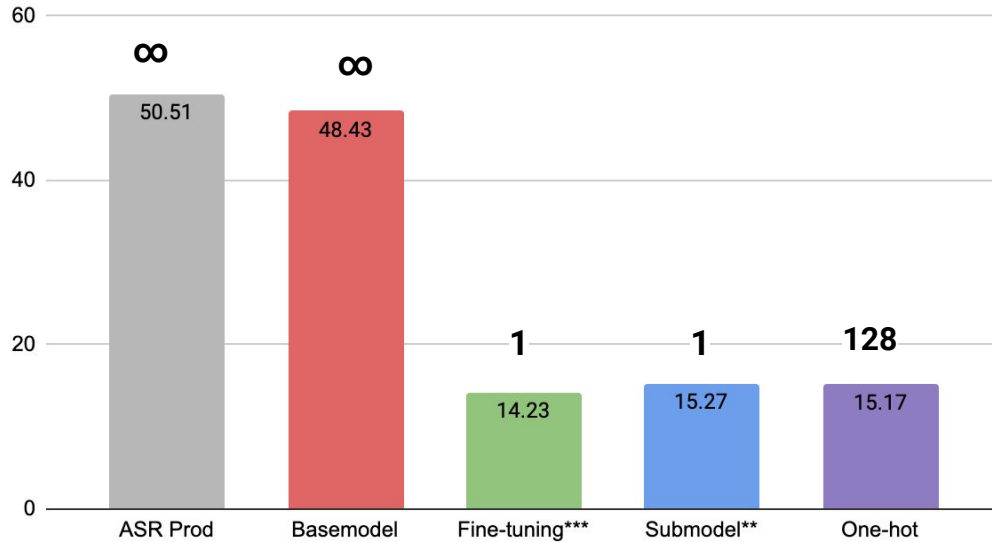
Post Training Splitting of Submodels



- A small Submodel is now in its own file
- Discard the embedding lookup table

Results of 128 Submodels in Parallel (same training budget)

Approaches - Mean WER across 128 Speakers

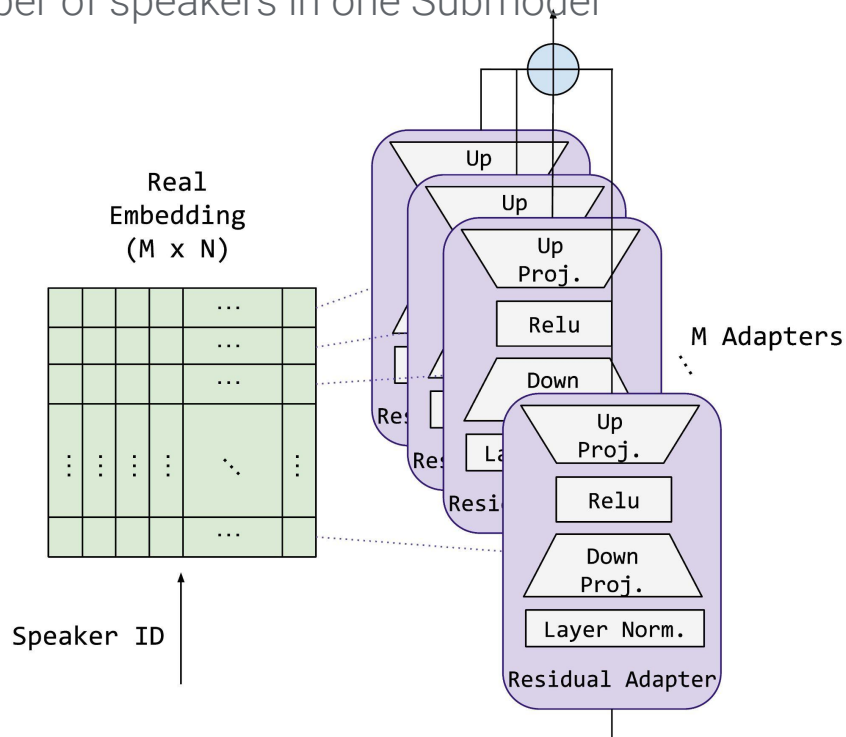


- We tested up to 512 user models in a single training job (4h) without loss in accuracy
- Free 512X throughput Parallelism
- With only 20 TPUS, we can scale/enroll up to 1.8 million users a month

Scale Further: Real-valued embedding Submodel

- Share a set of M adapters across speakers
- Learn an embedding of N speakers to weight those adapters
- **Submodel:** embedding + adapters across layers
- $M \ll N \Rightarrow$ Model a very large number of speakers in one Submodel

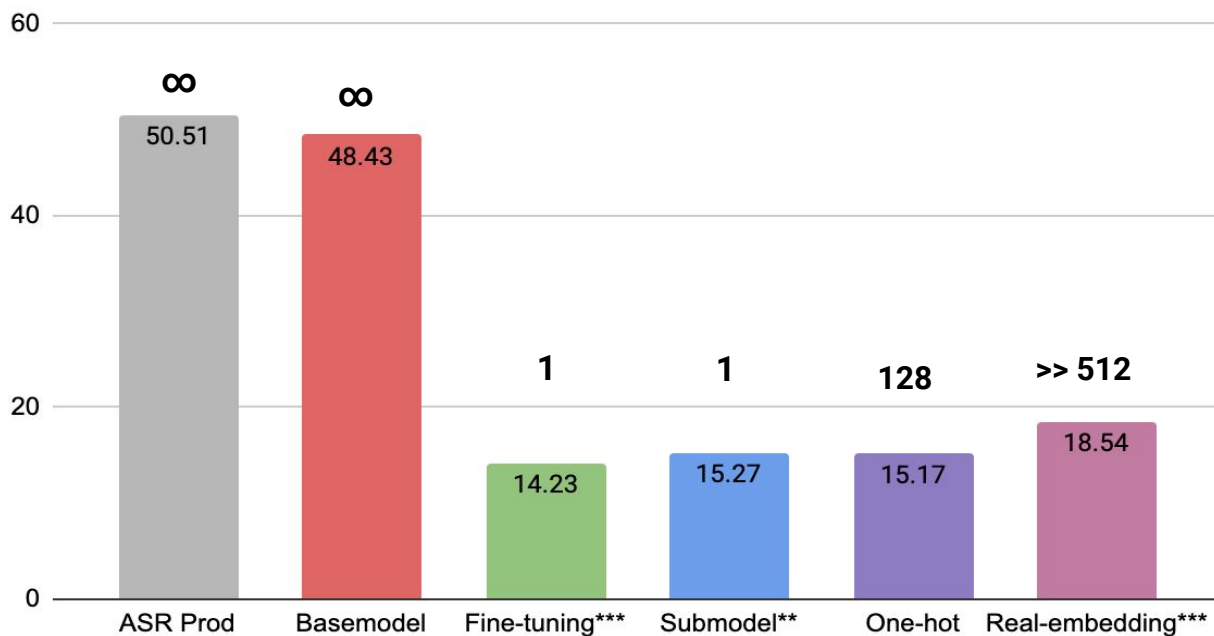
- Load and cache this Submodel on-the-fly per request, thus M can't be too large
- Optimization: Regroup active users together to increase cache hits



Real-Valued Embedding Submodel

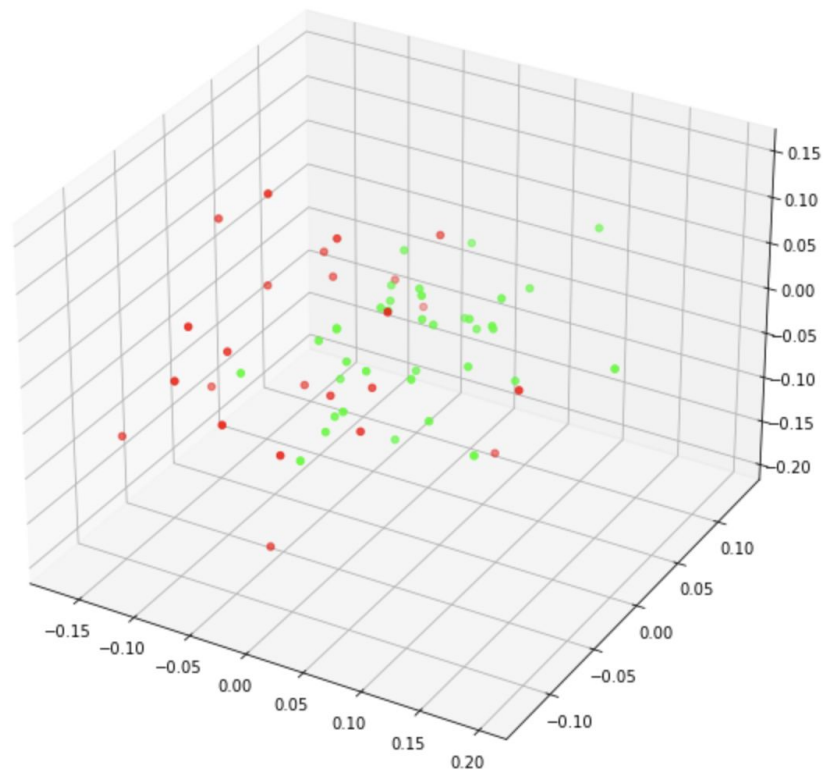
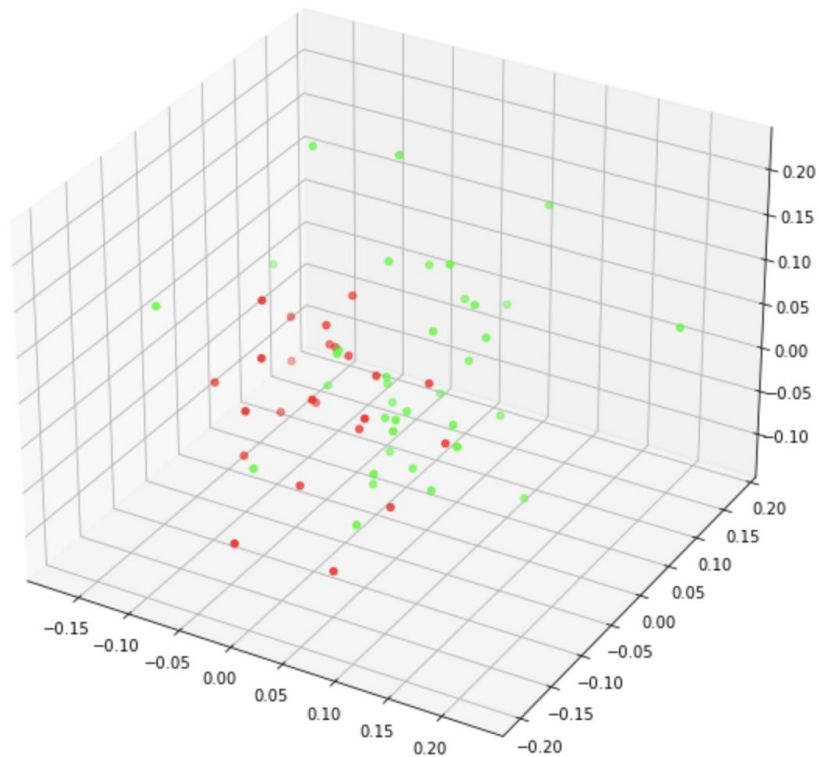
- Loss in WER but can scale to thousands of speakers in a single Submodel.
- Share data across speakers but structurally via embedding (e.g., etiology)

Approaches - Mean WER across 128 Speakers



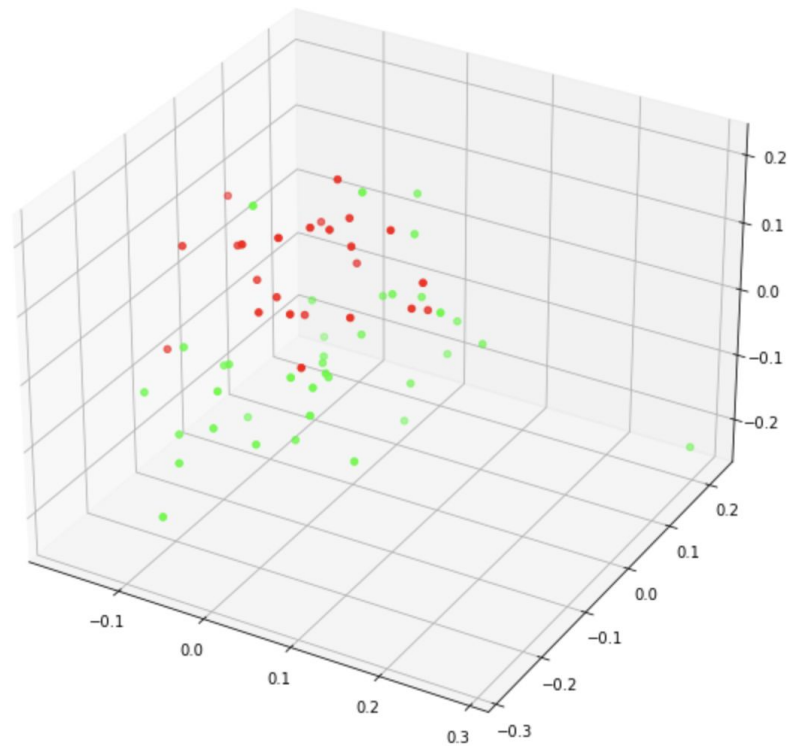
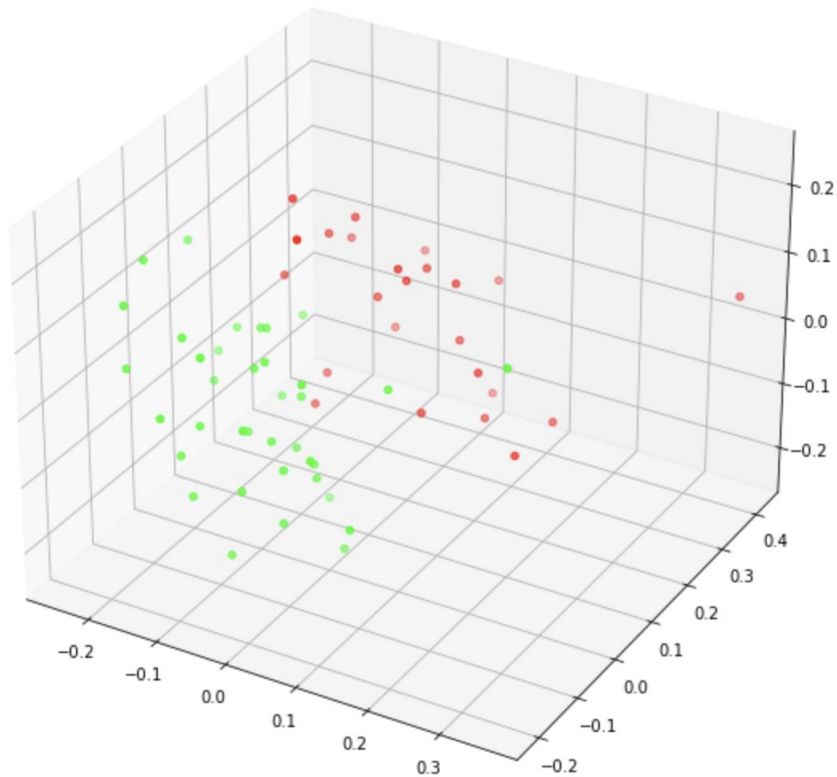
Embedding space between ALS and Down syndrome (Layer 0, 2)

- Model has never been given the etiology just speaker id
- 8D → Multidimensional scaling to 3D to plot



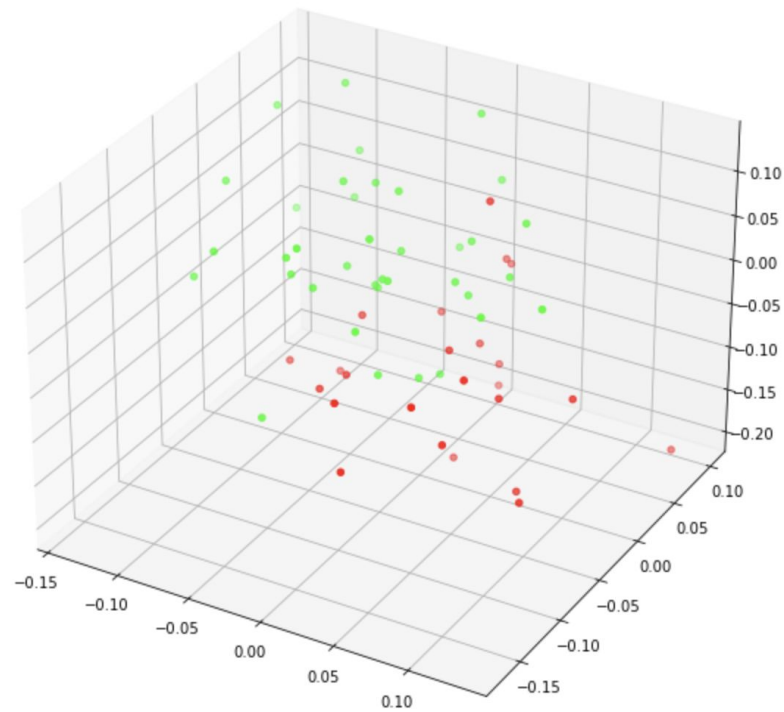
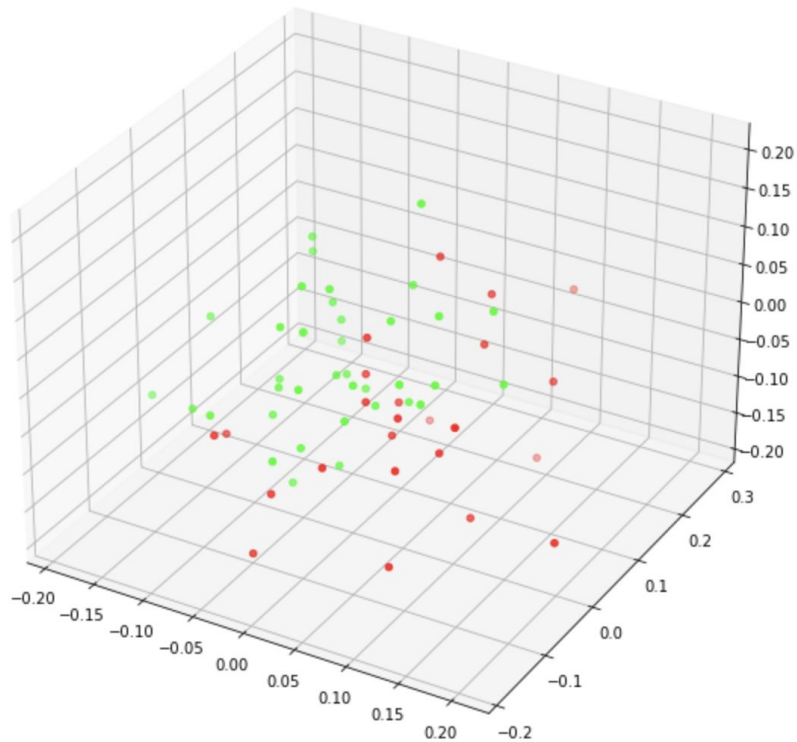
Embedding space between ALS and Down syndrome

Layer 9, 10



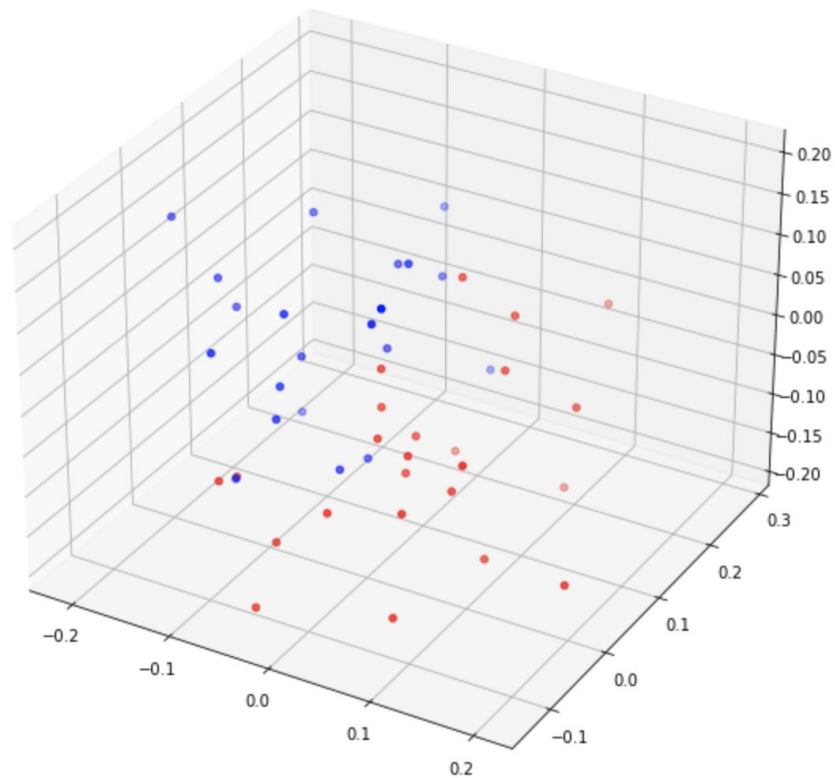
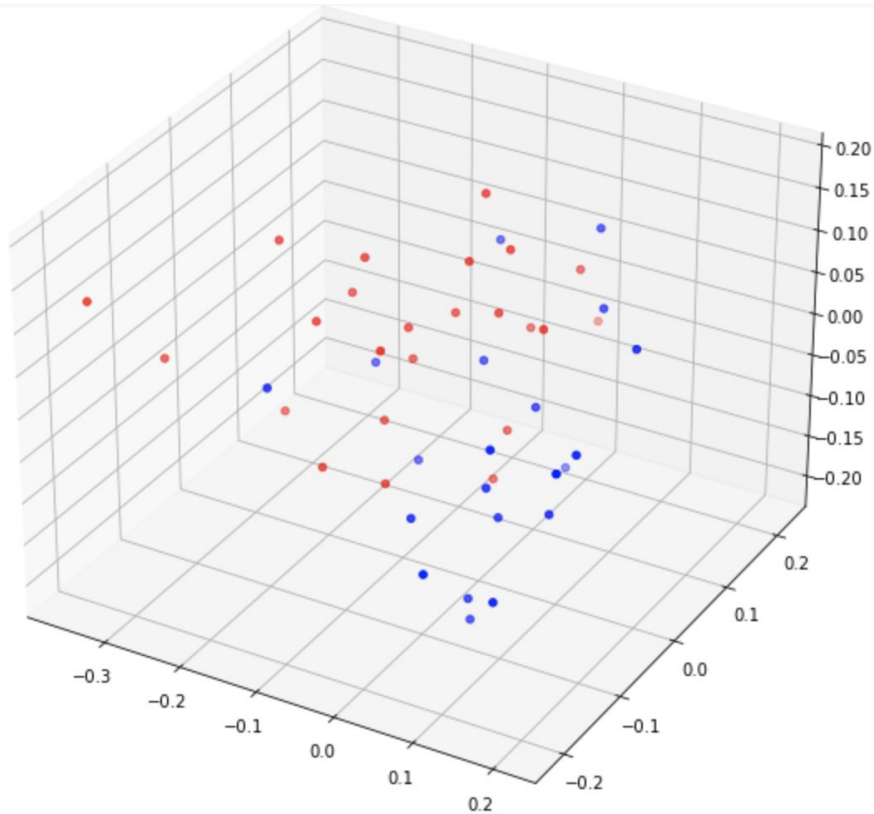
Embedding space between ALS and Down syndrome

Layer 15, 16



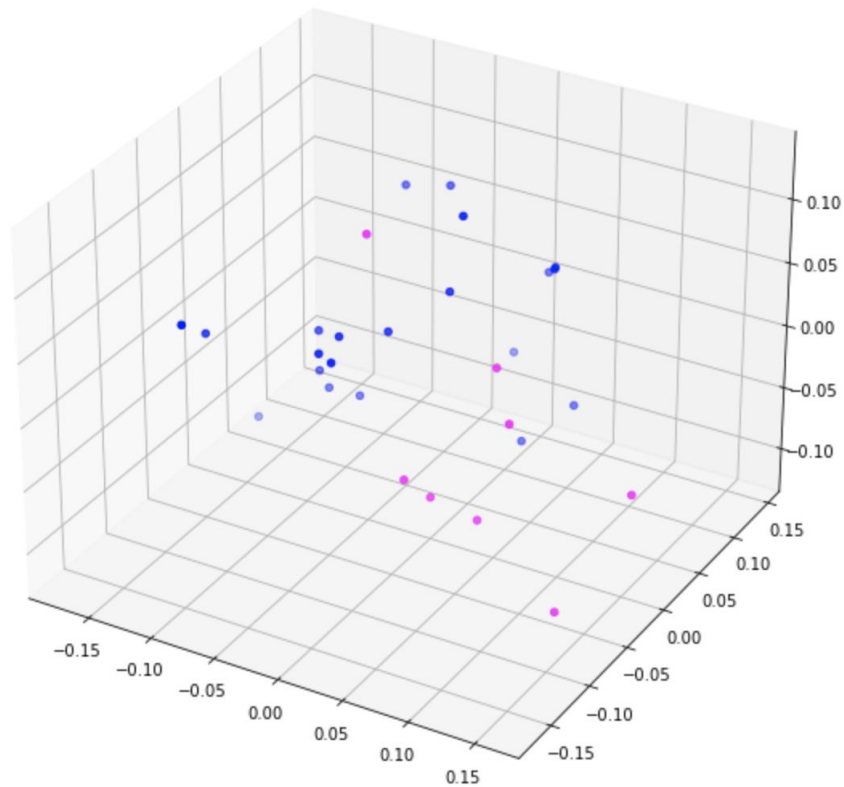
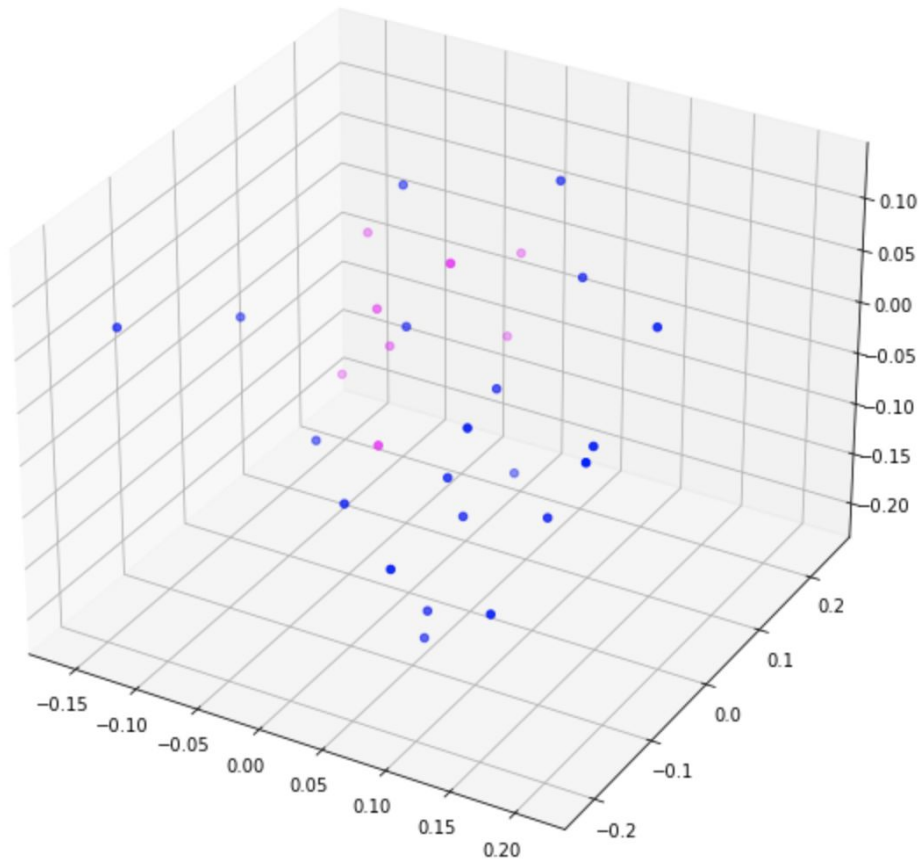
Embedding space between Down Syndrome and Cerebral Palsy

Layer 1, 15



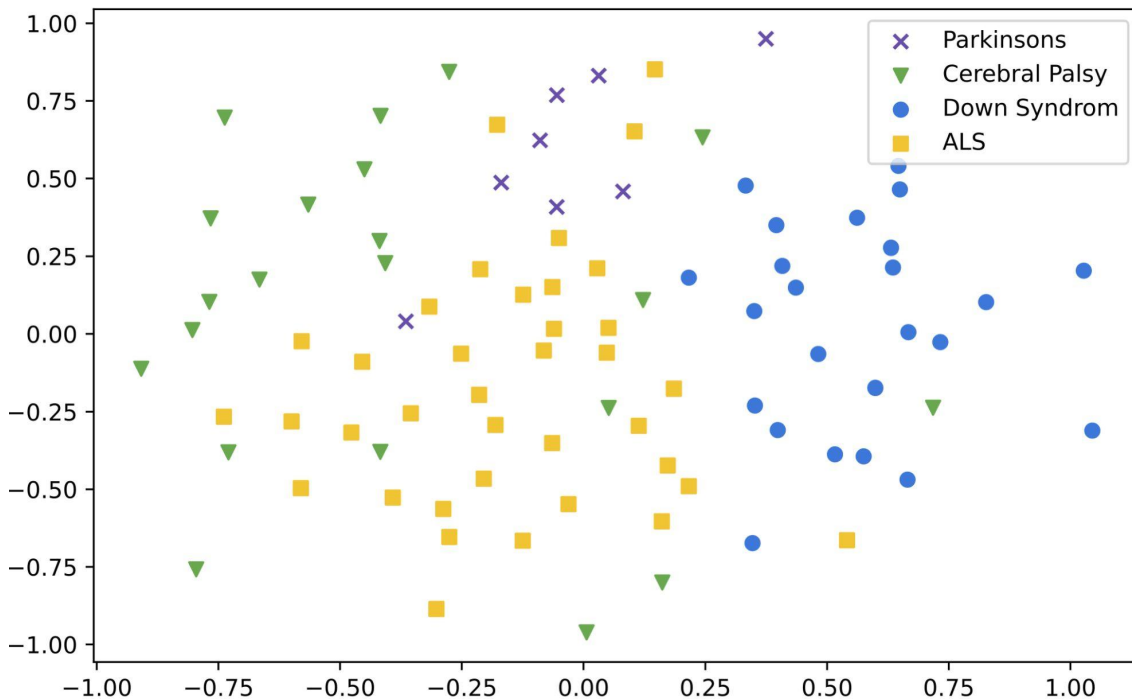
Embedding space between Cerebral Palsy and Parkinson's

Layer 1, 2



Embedding Submodel learns structural 'correction' of the Basemodel

- Learned embedding can easily separate speakers from different etiologies
- Logistic classifier using embedding vectors: average pairwise accuracy of 92.8%.
- The Submodel is, not memorizing speakers but rather learning a structural correction of the Basemodel



Multi speaker training using speaker embedding (closed set)

Pros:

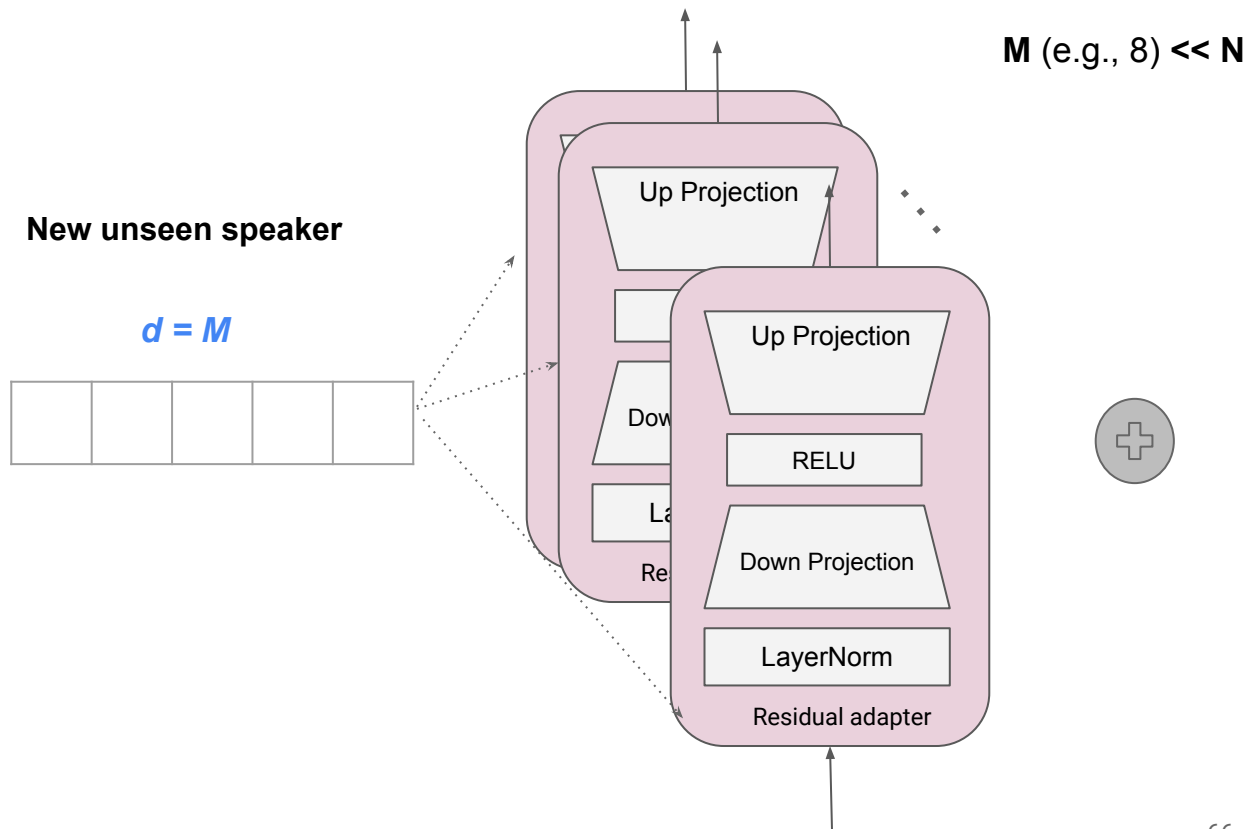
- Train MANY speakers in a single job!
(Scales to thousands in one job - todo!)
- Parameter sharing - learn from others
- Small submodel

Cons:

- Speaker models are not independent.

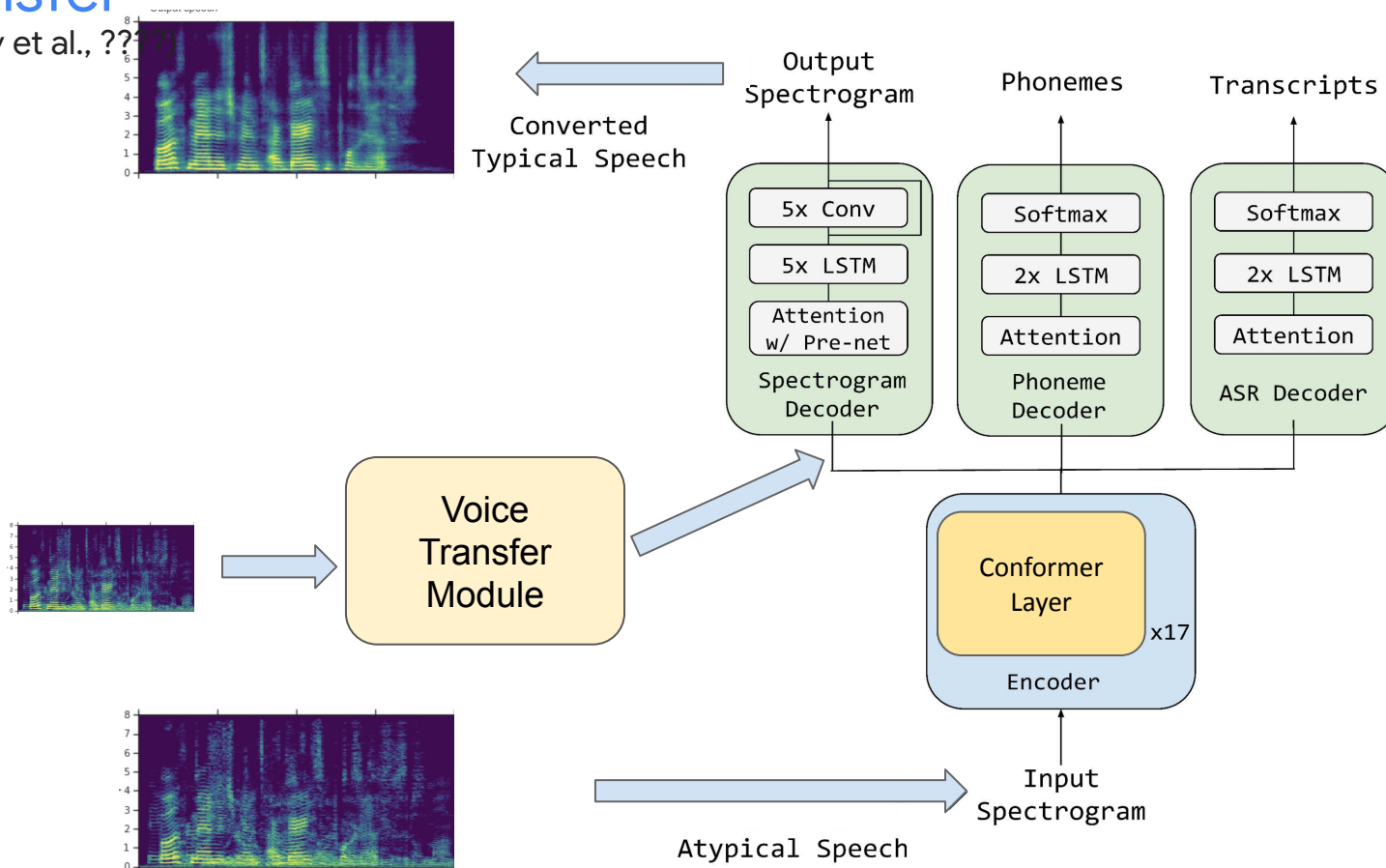
A new unseen speaker: How to open the closed set?

- Assign a random gaussian noise as the speaker embedding vector
- Use the pretrained 8 adapters
- Fine tune submodel (embedding + adapters)
- Also can train multi speakers



Parrotron Model with Voice Transfer

(biadsy et al., ???)



Conclusions

- Although **fine-tuning works well** for model personalization, it **poses significant scalability challenges**
- Residual adapters are a great choice of parameter efficient Submodel for personalization
- **Loading dynamically and activating a Submodel** during prediction per request adds negligible extra latency and **solves serving very large number of specialized models**
- **One-hot-embedding** over Submodels show **substantial benefit for parallalsim** while showing no loss in WER.
- **Real-valued embedding Submodel** (adapters and embedding) learns an **efficient model for a large group of speakers**, utilizing parameter sharing

2023: Launch Relate in India for Hindi



Thank you!

Project Relate:
sites.research.google/relate/

