Mobile Edge AI: Signal and Image Processing Deep Learning Models as Real-Time Smartphone Apps

Nasser Kehtarnavaz
Department of Electrical and Computer Engineering
University of Texas at Dallas

Communications/Signal Processing Society Chapter of IEEE Orange County Section
May 26, 2022

Edge Computing Overview

- Edge computing is about shifting intelligence from the cloud side to the edge side, or establishing intelligence locally at the edge of the network where data get captured or where the sensor/IoT device is located.

- Transmitting data from the edge to the cloud for signal/image processing and then returning response back to the edge takes time, involves costs, and compromises privacy. In general, performing signal/image processing on the cloud side does not scale well as the number of sensors/IoT devices grow.

Attributes of Edge Computing vs. Cloud Computing for Signal/Image Processing Applications

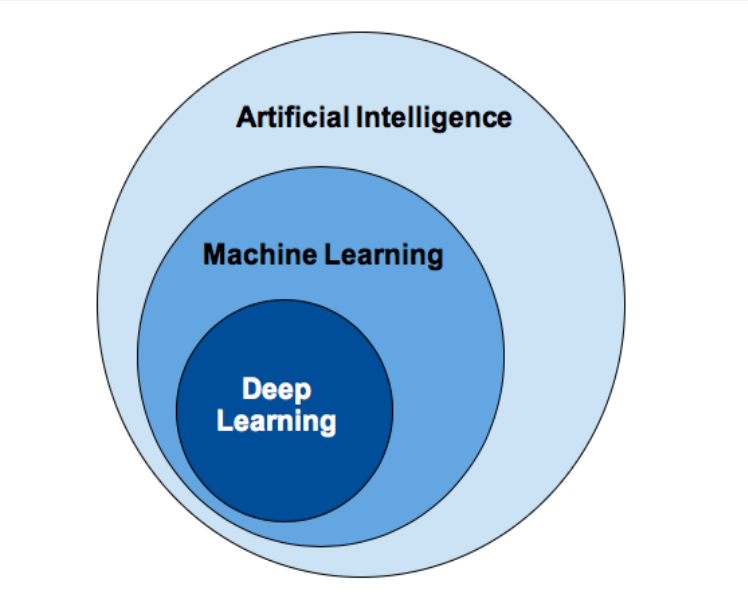
Latency	Signal/image processing apps can run in real-time without waiting for a response from the network.
Availability	Signal/image processing apps would run even when outside of the network coverage.
Privacy	Data do not need to leave the device.
Cost	Not as many server farms needed when all or part of the processing done on the device.

Edge AI or Mobile Edge AI:
Intelligent decision making on the edge side

- Machine learning (ML) is a subset of Artificial Intelligence (AI or human-like intelligent decision making) techniques that allow computers to “learn” from data.

- Deep learning (DL) is a subset of machine learning (ML) techniques conducted by deep neural networks.

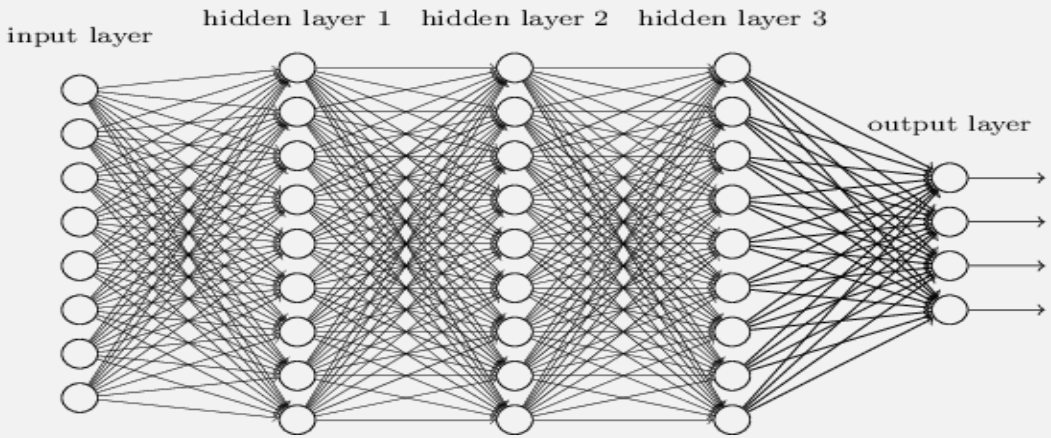
- <u>This talk is about running signal/image processing deep learning models in real-time on smartphones as edge devices or about Mobile Edge AI.</u>



Deep Learning

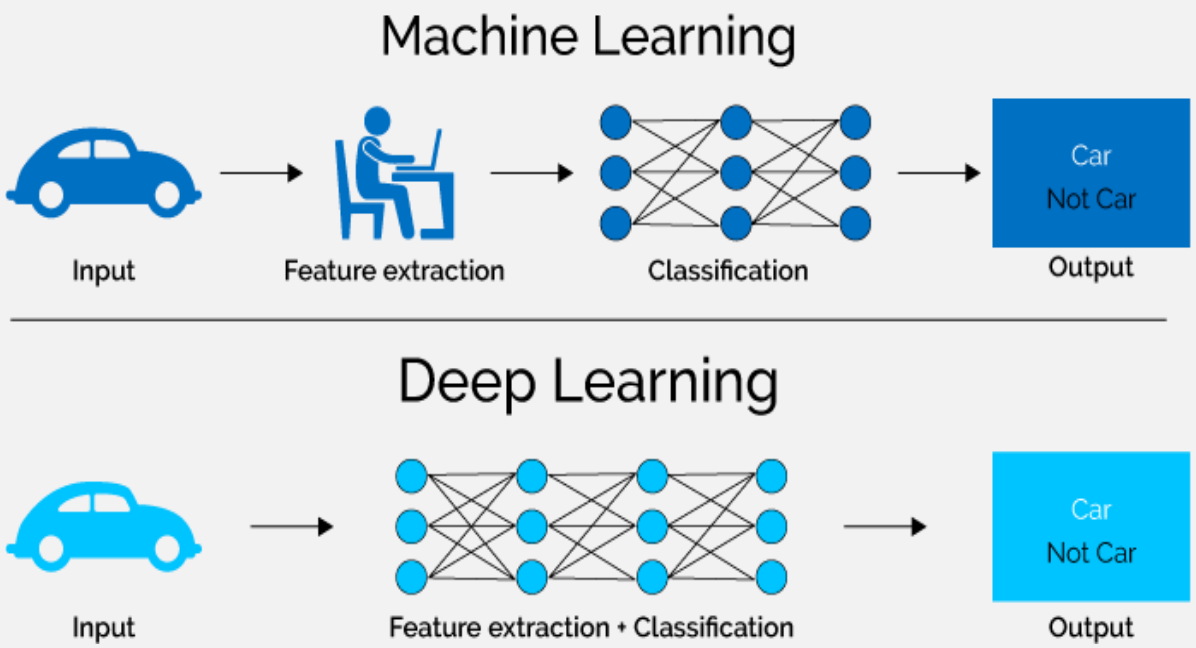
Deep learning is achieved via Deep Neural Networks (DNNs); a large cascade of processing elements arranged in many (deep) layers.

In the last few years, there has been a tremendous growth in the use of deep neural networks for solving various problems; nearly in all engineering conferences these days, one sees many papers involving DNNs.



Deep Learning vs. Conventional Machine Learning Approaches

The growth in the use of deep learning is due to the fact that feature extraction and classification/regression are done together in one structure of deep neural network compared with conventional machine learning in which feature extraction and classification/regression are done separately.



Attributes of smartphones as edge devices

- Smartphones have powerful multi-core ARM and GPU processors that allow running computationally intensive signal/image processing algorithms in real-time as apps. These processors are considerably more powerful than the processors of some commercially available and programmable boards such as Raspberry Pi, Arduino, MSP 430, DSP boards, etc.

- Smartphones are everywhere (nearly 300 million people in the US have or use smartphones – each year more than 1.5 billion smartphones are sold in the world).

- Software development tools of smartphones are free of charge (can be downloaded from Google and Apple websites for free) and are well-maintained and updated by these companies. Basically, smartphones are widely deployed edge devices.

Deep Learning Software Tools (1)

- There are already many deep learning software tools (see below).

- Major difficulty encountered for turning deep learning-based signal/image processing algorithms into real-time smartphone apps - information regarding running deep neural networks in real-time on smartphones are scattered all over the literature and not available in one place.

- A roadmap has been put together so that researchers would need to go to only one place to turn their deep learning-based signal/image processing algorithms into apps running in real-time on smartphones as edge devices.

- The codes developed for doing so are made open-source; GitHub link: ***SIP-Lab.github.io***

TensorFlow mxnet GLUON Chainer Caffe

theano PYTORCH K The Microsoft Cognitive Toolkit Caffe2

Deep Learning Software Tools (2)

- Among existing software tools, the following three tools are found most suited for creating deep learning apps on smartphones:
  - TensorFlow
  - Keras
  - CoreML

- These tools offer:
  - portability to smartphones
  - are actively supported

Deep Learning Software Tools (3)

TensorFlow

- Developed by Google
- Trained models can be deployed using the “TensorFlow Mobile” gradle build

Keras

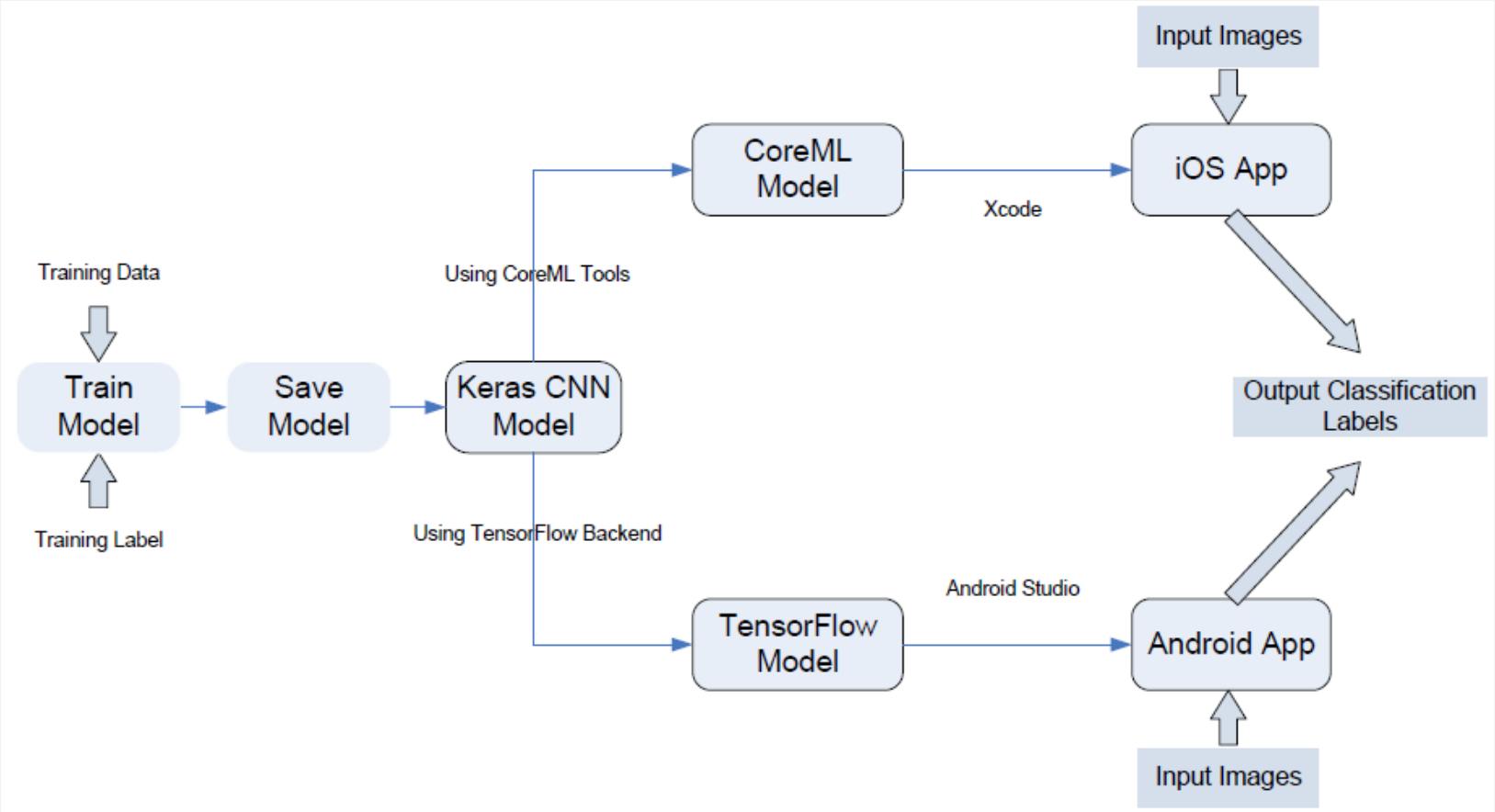
- High-level Python library which runs on top of TensorFlow
- Keras models are effectively TensorFlow models which can be extracted for deployment

CoreML

- Framework to run deep learning models on iOS devices
- Has the Python-based tool CoreMLTools for converting Keras models to CoreML models

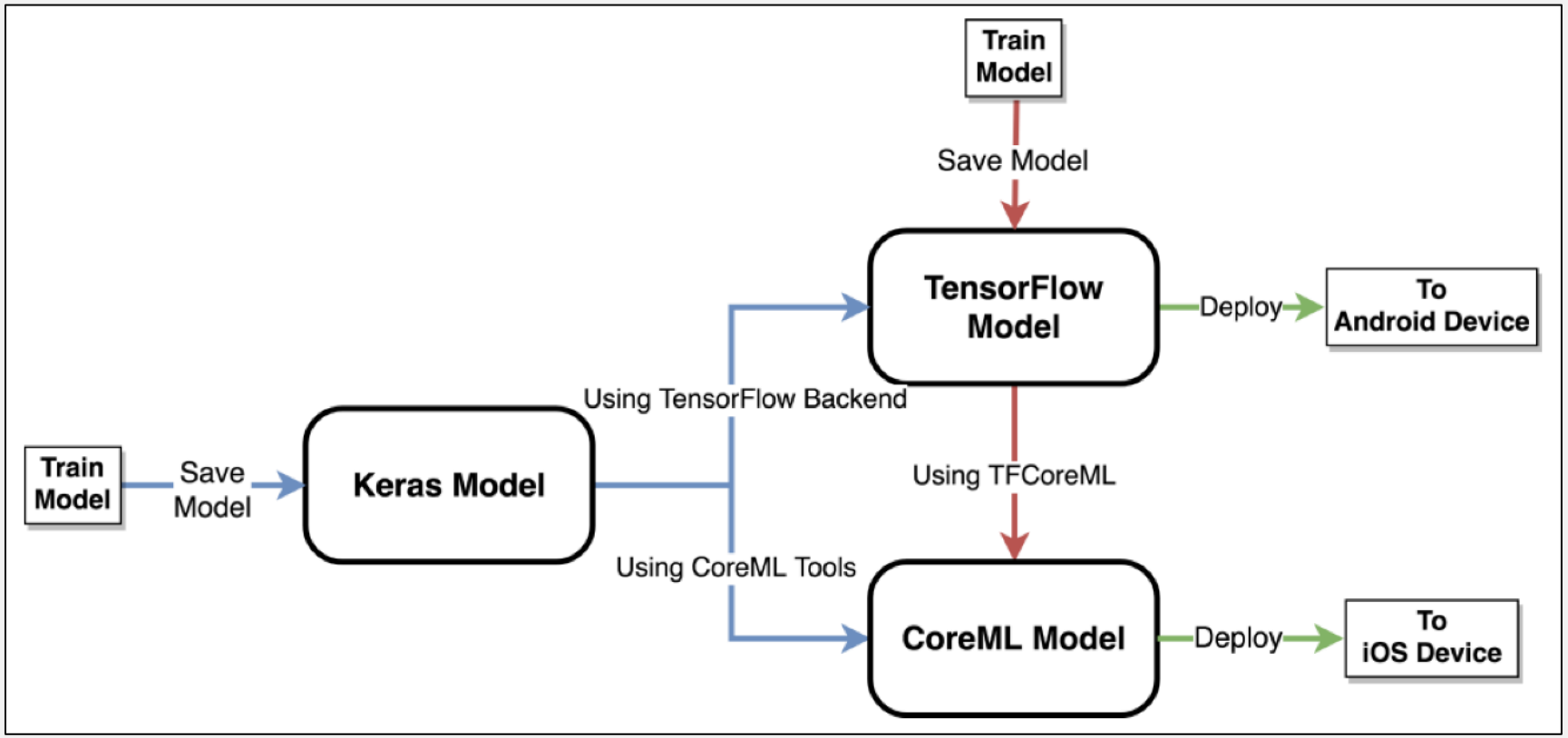
Roadmap devised for deployment of deep learning models as real-time apps on smartphones*

This figure shows the roadmap for creating deep learning apps on smartphone*.



*A. Sehgal and N. Kehtarnavaz, "Guidelines and benchmarks for deployment of deep learning models on smartphones as real-time apps," *Machine Learning and Knowledge Extraction* (open access), pp. 450-465, Feb 2019.

Deployment Path

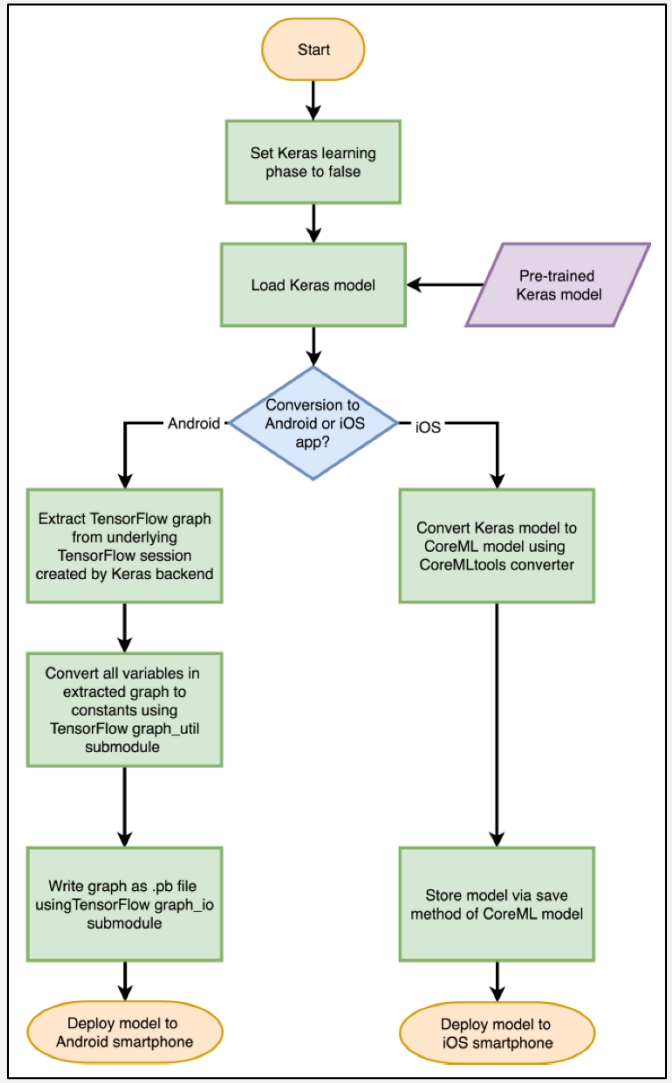


- Blue lines indicate the flow when using Keras for training
- Red lines indicate the flow when using TensorFlow for training

Deployment Steps (Keras)

- For Android conversion:
  - Extract the underlying TensorFlow model
  - Convert weights and biases to constants
  - Save the graph to a protocol buffer format

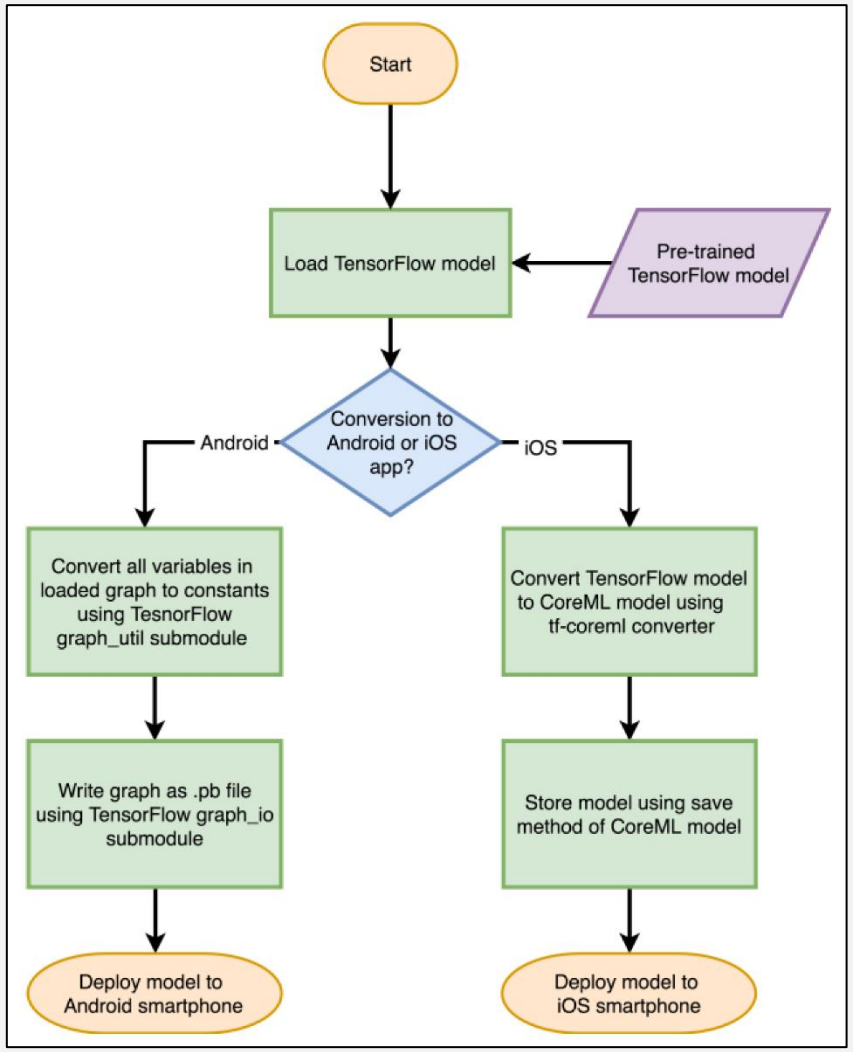
- For iOS conversion:
  - Specify the input pre-processing details of the model, e.g. scaling
  - Use the CoreMLTools converter to convert the model to an MLModel



Deployment Steps (TensorFlow)

- For Android conversion:
  - Convert weights and biases to constants
  - Save the graph to a protocol buffer format

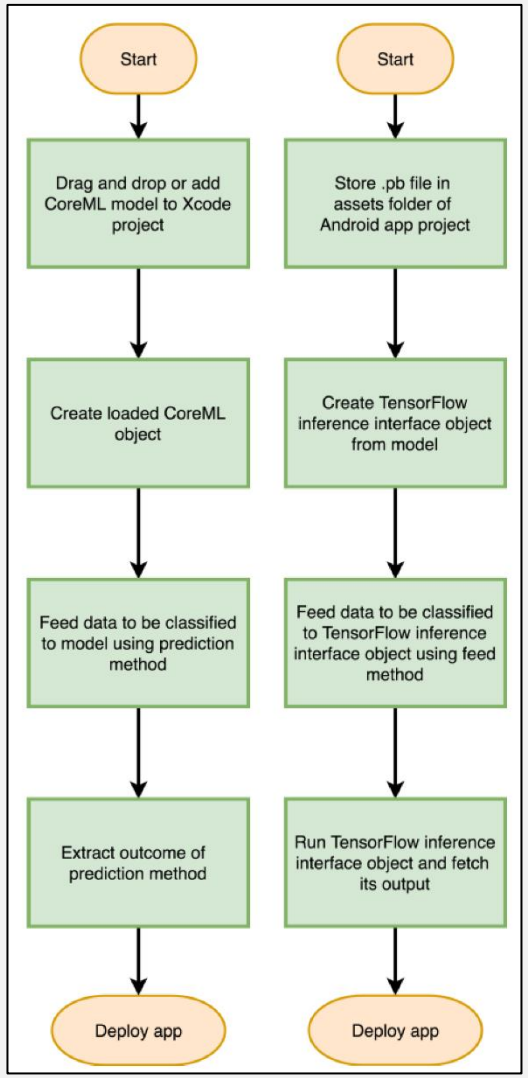
- For iOS conversion:
  - Convert the TensorFlow model to MLModel using the *tf-coreml* converter
  - Specify the input pre-processing details of the model, e.g. scaling
  - Save the model as an MLModel



Creating iOS and Android apps of trained models

- To create iOS app:
  - Add the MLModel to the Xcode project
  - Create a class for the added model
  - Instantiate the object of that class
  - Feed the input to the model in the appropriate format
  - Extract the outcome from the prediction method

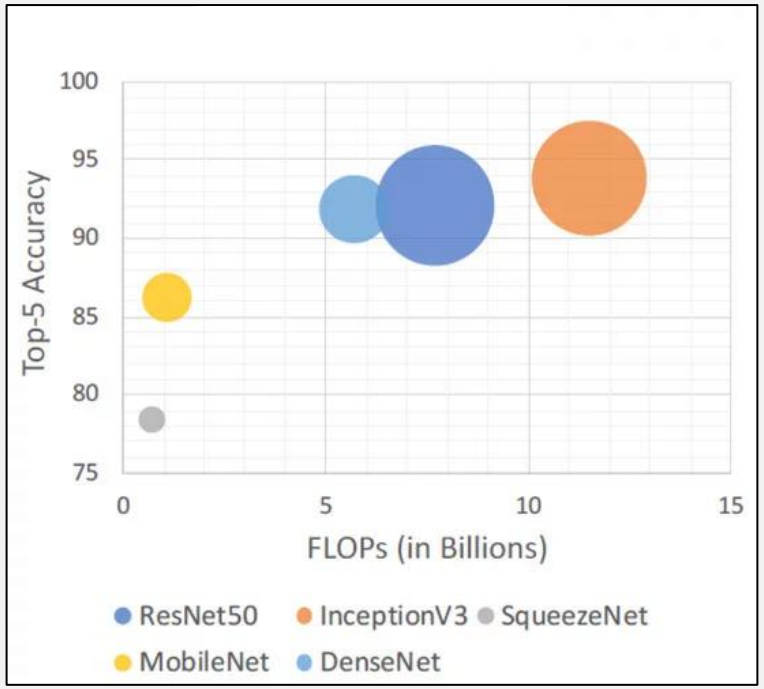
- To create Android app:
  - Store the model in the assets folder of the Android Studio project
  - Create a TensorFlow inference interface object using the model
  - Feed data to be classified to the interface
  - Fetch the output of the interface



Guidelines used to implement ImageNet Challenge deep learning models as real-time apps running on smartphones

Deep Learning Model Apps					
Layer Information	ResNet50	InceptionV3	SqueezeNet	MobileNet	DenseNet121
Input-Size	224x224x3	299x299x3	227x227x3	224x224x3	224x224x3
CONV Layer					
# of CONV Layers	49	95	26	27	120
Depth	49	46	18	27	120
Kernel Size	1,3,7	1,3,5,7	1,3	1,3	1,3,7
Strides	1,2	1,2	1,2	1,2	1,2
# of Channels	3 - 2048	3 - 2048	3 - 1000	3 - 1024	3 - 1024
# of Filters	64 - 2048	32 - 2048	16 - 1000	32 - 1024	32 - 1024
FC Layer					
# of FC Layers	1	1	0	1	1
# of Channels	2048	2048	0	1024	1024
# of Filters	1000	1000	0	1000	1000
Parameters	25.6 M	23.8 M	1.2 M	4.3 M	8 M
(Floating Point Operations) FLOPs	7.7 B	11.5 B	714 M	1.1 B	5.7 B
Model Storage Memory	102 MB	96 MB	5 MB	17MB	33MB
Top-5 Accuracy (Single Crop)	92.1%	93.8%	78.4%	86.2%	91.8%

Benchmarks
ImageNet Challenge deep learning models on smartphones



- Top-5 accuracy vs. number of FLoating Point Operations (FLOPs)
- Size of circles represents the app size
- All running at video rate

Demo (video)

An example deep learning model running in real-time on smartphone – trained on ImageNet (1M+ images, 1000 classes)

Roadmap can be applied to different signal/image processing applications - two example applications presented next

- An image processing application – a first-pass eye exam via a smartphone app
- A signal processing application – a smartphone-based virtual hearing aid testbed

Recent development in eye exam technology

- https://www.fda.gov/newsevents/newsroom/pressannouncements/ucm604357.htm
  FDA website: “The device, called **IDx-DR**, is a software program that uses an artificial intelligence algorithm to analyze images of the eye taken with a retinal camera. A doctor uploads the digital images of the patient’s retinas to a cloud server on which IDx-DR software is installed. If the images are of sufficient quality, the software provides the doctor with one of two results: (1) “more than mild diabetic retinopathy detected: refer to an eye care professional” or (2) “negative for more than mild diabetic retinopathy; rescreen in 12 months.”

Image from IDx-DR https://www.eyediagnosis.net

Objective in this image processing application

- Use smartphones together with lenses that are commercially available (e.g., D-EYE lens about $400) as a low-cost and universally accessible alternative to fundus cameras ($5000-$10,000) in order to carry out first-pass eye exams at the edge or in clinics with no access to fundus cameras/ophthalmologist.

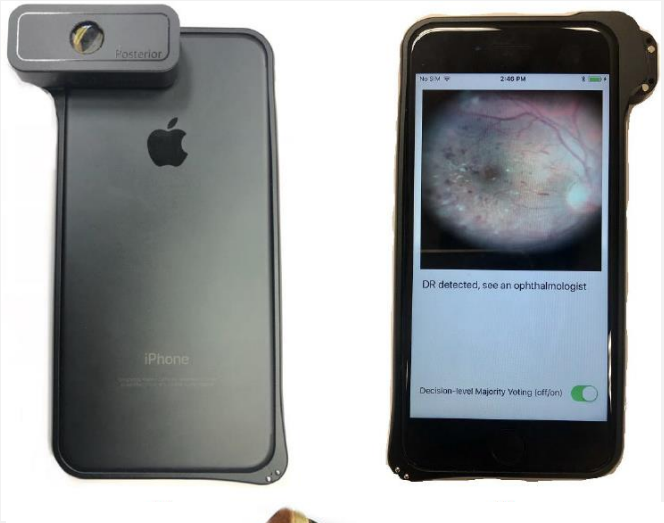
- The roadmap discussed earlier was utilized to develop a deep learning-based app to process eye retina images on smartphones in real-time or in an on-the-fly manner*.

*H. Wei, A. Sehgal, and N. Kehtarnavaz, "A deep learning-based smartphone app for real-time detection of retinal abnormalities in fundus images," *Proceedings of SPIE Conference on Real-Time Image Processing and Deep Learning*, Baltimore, MD, April 2019.

Commercially Available Lenses

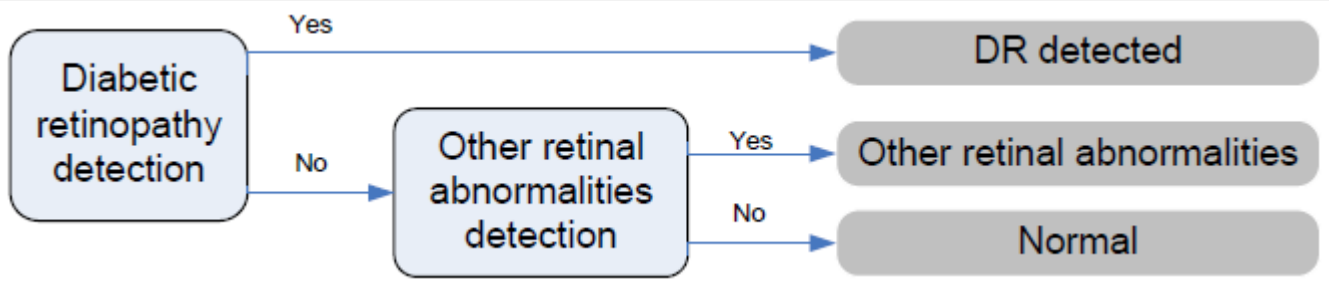
Several lenses are commercially available as attachments to smartphones for examining retina.

Image from Summit Medical Group website
http://www.summitdeanehill.com/author/admin/



A two-step approach app developed

- A **two-step approach** app is designed and implemented. In the first step, the detection is done for diabetic retinopathy (DR). In the second step, if no DR is detected, then the detection is done for other retinal abnormalities.

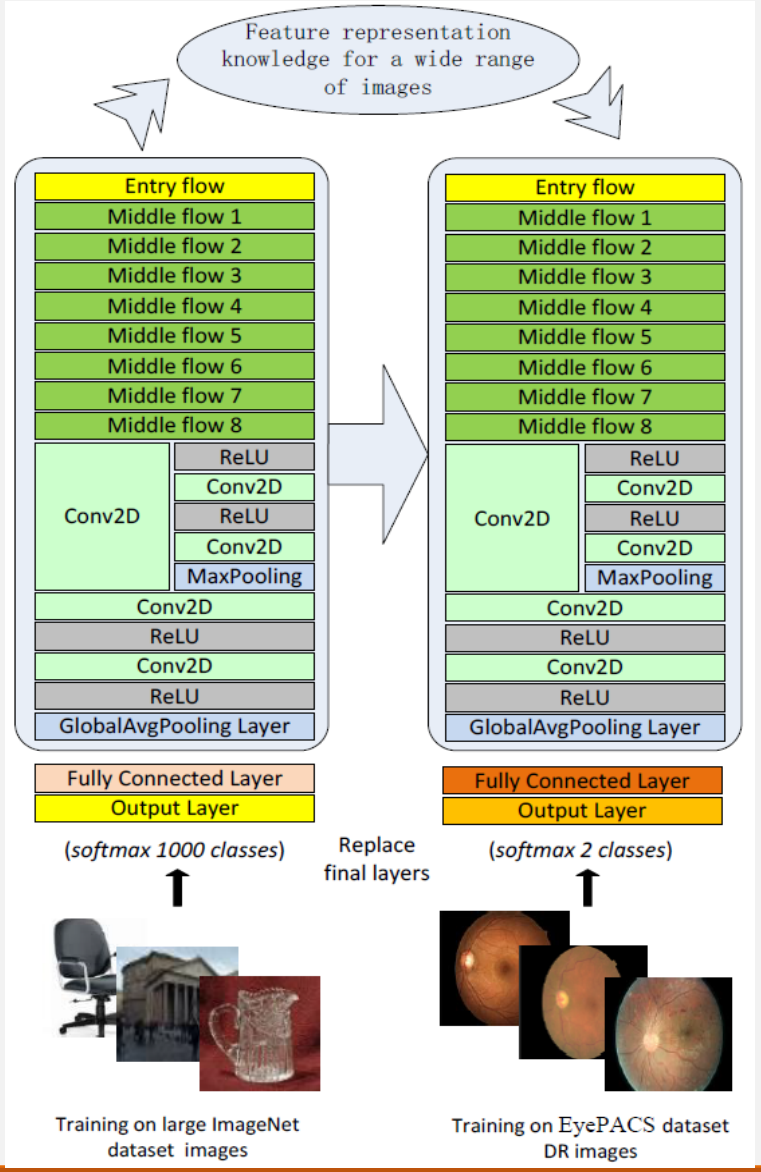


- Deep neural networks require a considerable amount of data in order to get trained well. Since the amount of DR images in the public domain datasets is relatively small for training, the transfer learning approach was used for training.

- In transfer learning, convolution layers are trained using a huge image dataset consisting of various objects. Then, the last fully connected layer is trained using the public domain DR datasets.

DNN architecture of the app – a modification of InceptionV3

30 frames get captured per second and two detection decisions are made per second. A decision-level majority voting is included in the app in order to smooth out fluctuations in detecting abnormalities when examining video streams in real-time.



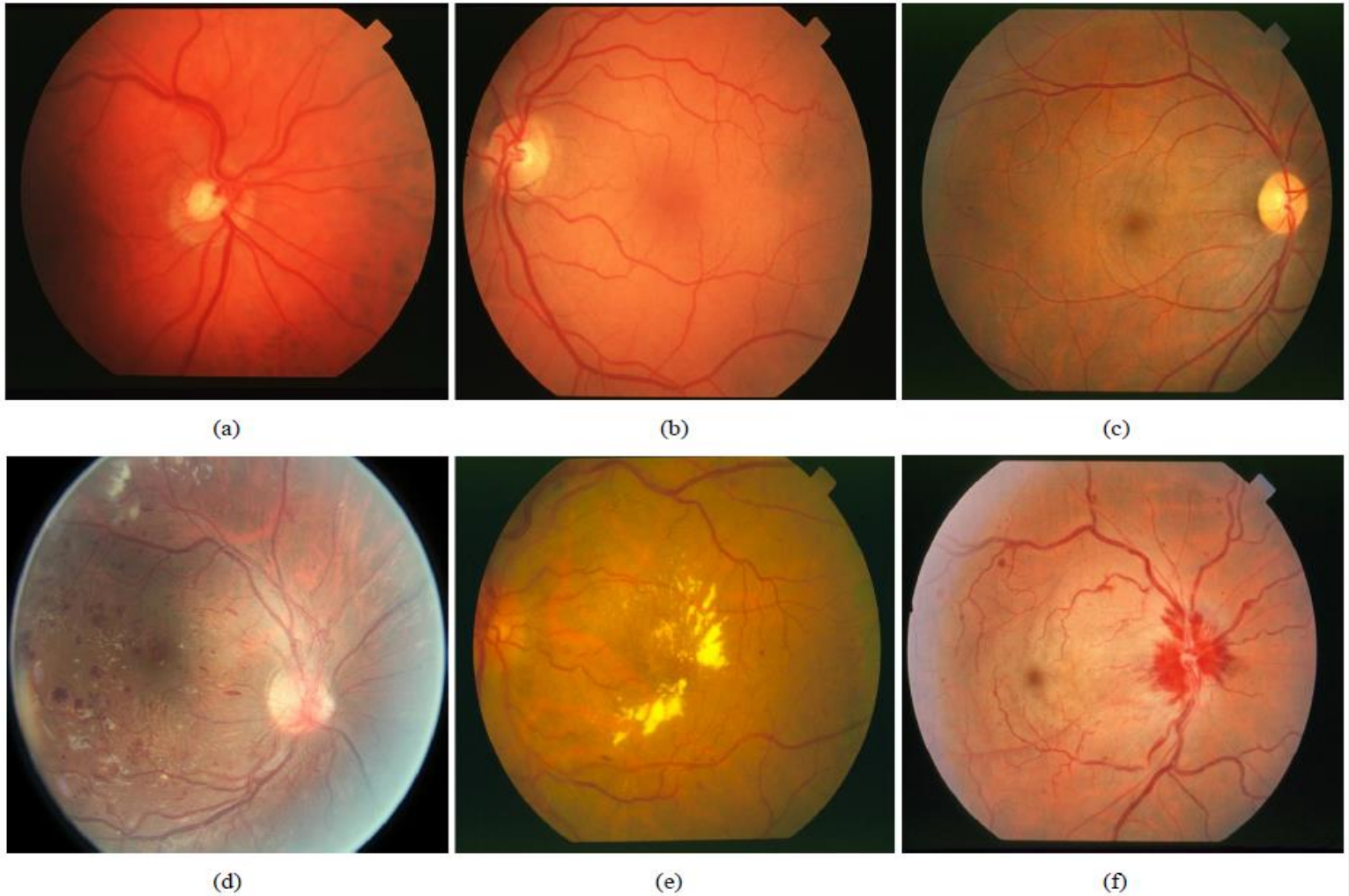
Datasets used for training fully connected layers

- **EyePACS** dataset consists of 35,126 images of retina images taken under a variety of imaging conditions (e.g., underexposed, overexposed, out-of-focus) and camera types.

- **STARE (STructured Analysis of the Retina)** dataset consists of 397 images grouped into 13 types of retinal abnormalities.

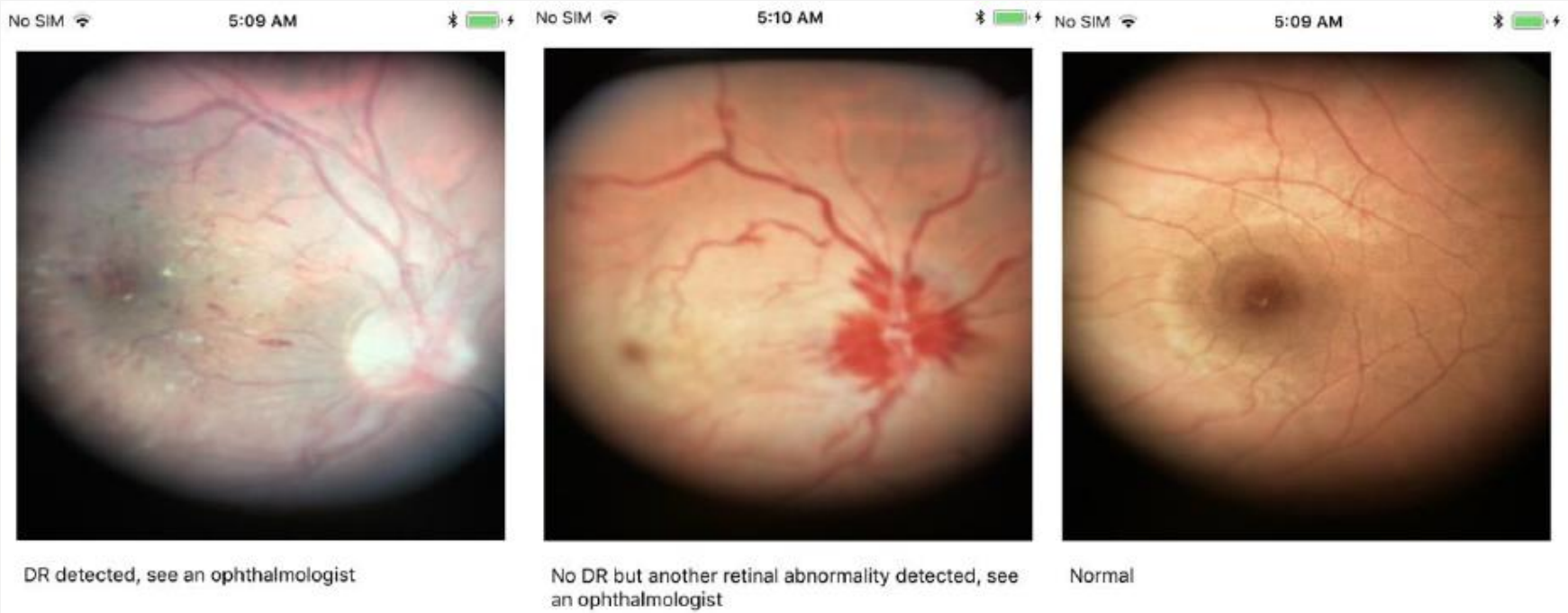
Sample retina images of the publicly available datasets

Image samples from the EyePACS and STARE datasets: (a) Normal, (b) Normal, (c) Normal, (d) Diabetic Retinopathy, (e) Early Stage Diabetic Retinopathy, (f) Central Retinal Artery and Vein Occlusion



App screen

- App Screen: When diabetic retinopathy is detected, the app screen displays "DR detected, see an ophthalmologist". When other retinal abnormalities are detected, the app screen displays "No DR but another retinal abnormality detected, see an ophthalmologist". In the absence of any abnormality, it displays "Normal".



App screen for three cases


Eye examination of the retina
using a smartphone fitted
with a D-EYE lens
Signal and Image Processing Lab

Detection accuracy

- In terms of detection accuracy, for the dataset images (no overlap between training and testing sets), the detection accuracy was found to be more than 95%. However, for deployment, an IRB-approved clinical study is needed to compare the outcome of the app to an ophthalmologist evaluation.

- In a recent follow-up work*, a multitasking deep learning model has been developed to identify all five stages of DR and not just its presence.

*S. Majumder and N. Kehtarnavaz, "Multitasking deep learning model for detection of five stages of diabetic retinopathy," *IEEE Access*, vol. 9, 123220-123230, Aug 2021.

Recently developed multitasking model

Classification task (5 outputs)

- Five output scores corresponding to five classes of DR. Output scores sum to one.

Cross Entropy loss function

Regression task (1 output)

- Estimates the severity score of DR. Lower score for lower severity and higher score for higher severity level of DR.

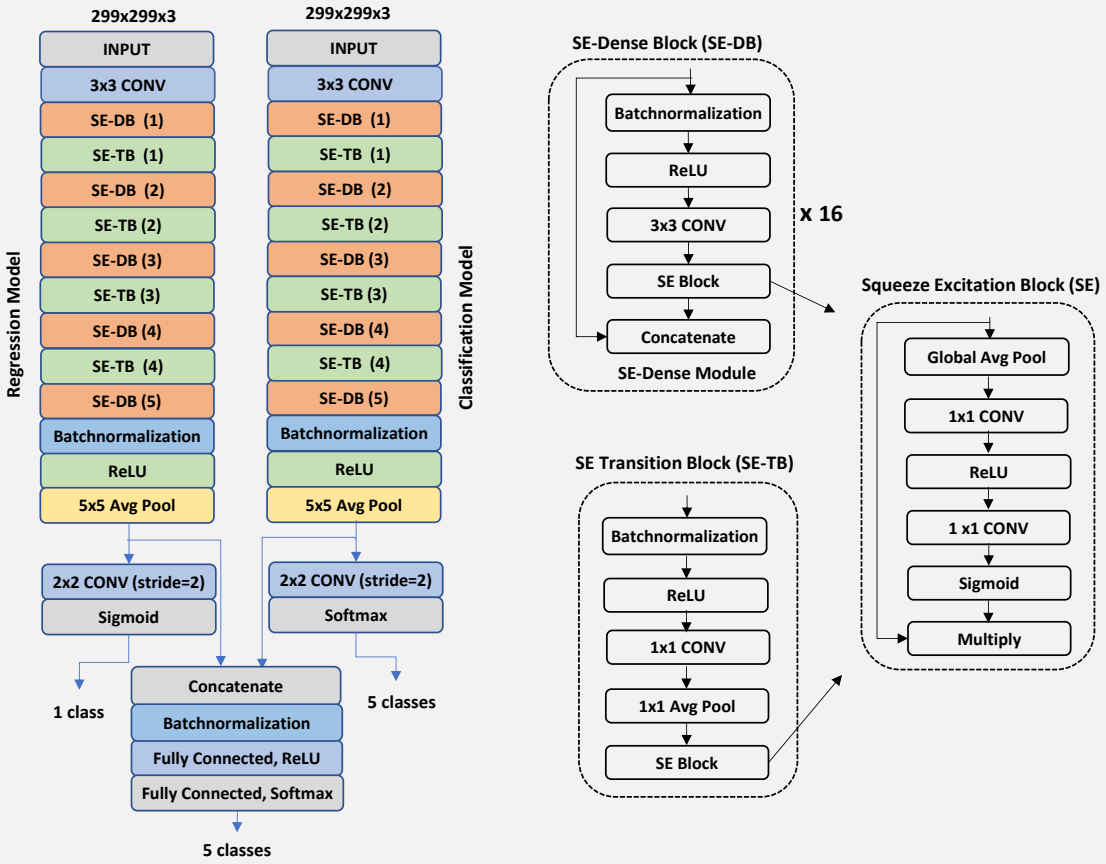
Mean Square Error loss function

Features Concatenate

Final Classification task (5 classes)

Classifies into one of the following five classes:

- No DR (0)
- Mild DR (1)
- Moderate DR (2)
- Severe DR (3)
- Proliferate DR (4)



Architecture of the developed Multitasking Squeeze Excitation Densely Connected Deep Neural Network (MSEDenseNet).

Summary (image processing application)

- In this image processing application, a DNN is developed and implemented as a smartphone app (both Android and iOS versions) to detect diabetic retinopathy abnormalities in real-time as retina video is captured by a smartphone fitted with a lens.

- ***Basically, this work has paved the way for smartphones to be used as edge devices or as alternative to fundus cameras/cloud servers for conducting a first-pass eye exam in a cost-effective and widely accessible way in places with no access to fundus cameras***.

Second Application – A Signal Processing Application: Turning smartphones into a testbed platform in the field or at the edge for hearing enhancement studies (1)

- <u>How hearing aides get fitted in a clinic:</u> Audiologists use the software tools provided by manufacturers to set the amplification gains across a number of frequency bands depending on the patient audiogram.

- These software tools map the gain settings to compression curves.

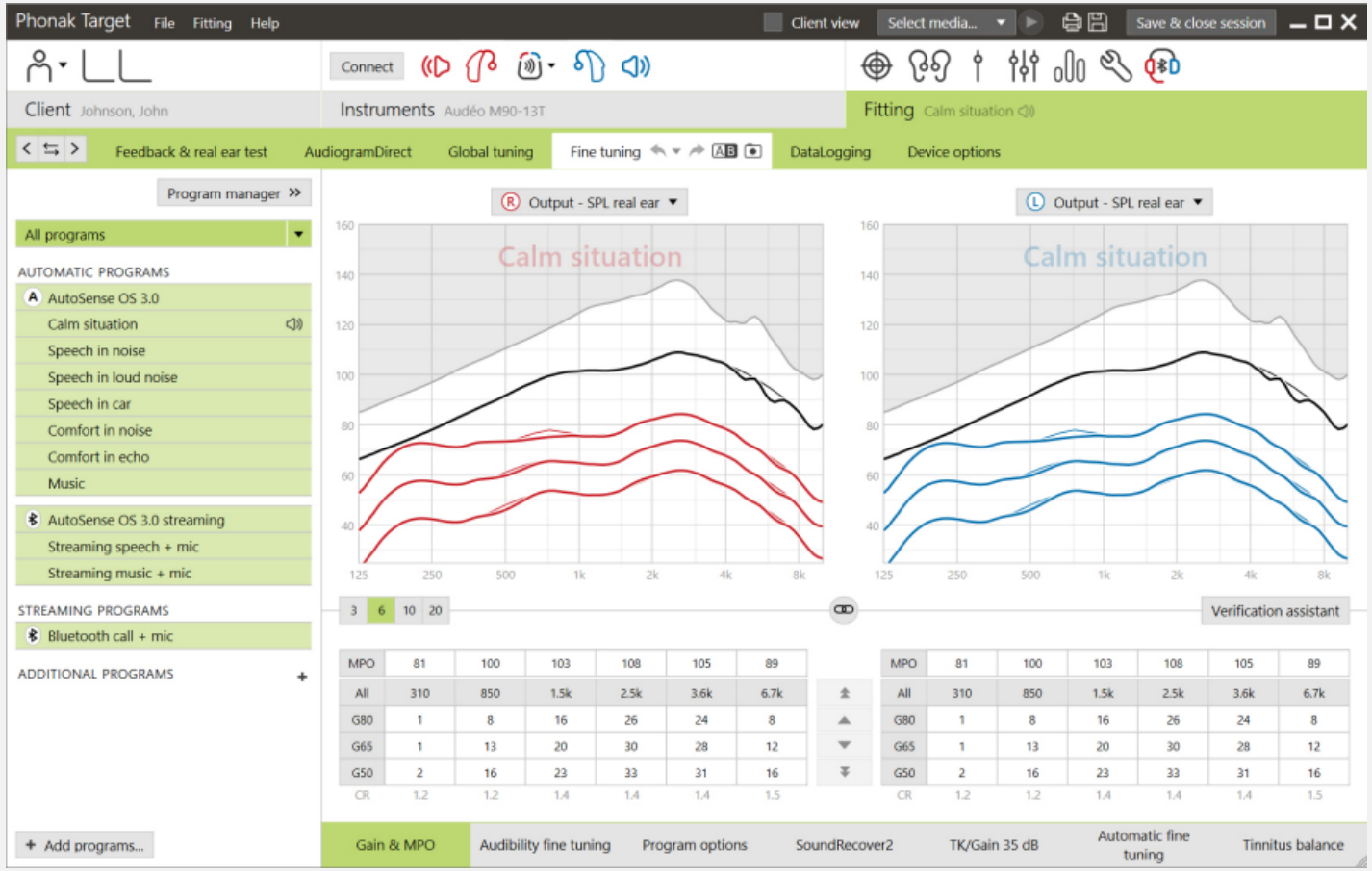


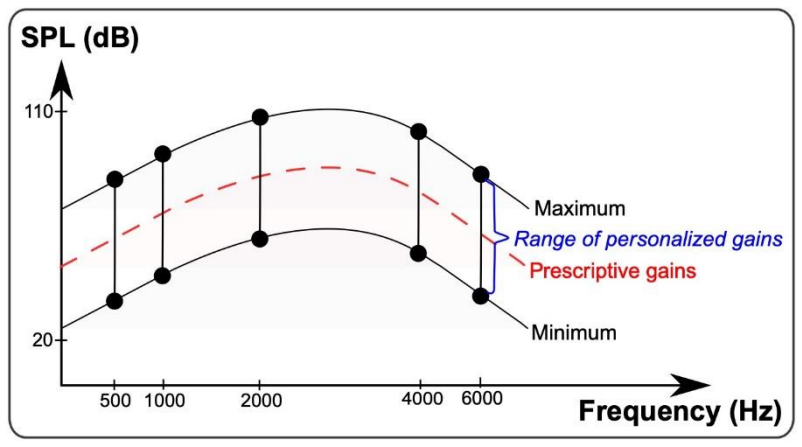
Image from Phonak website

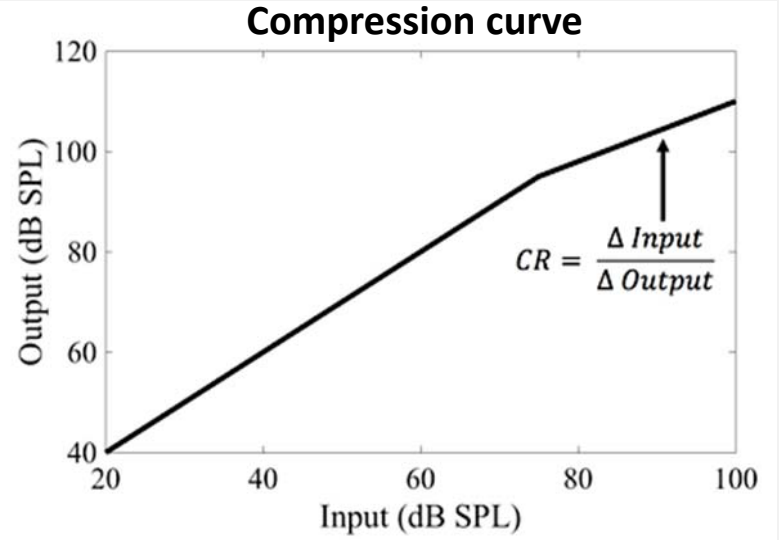
https://www.phonakpro.com/us/en/solutions/va/va-fitting/overview-target.htm

Compression function of hearing aids*

- Compression in hearing aids is used to amplify weak sound signals and suppress loud sound signals to bring them into the hearing range of those suffering from hearing loss.

- Compression is implemented by using compression curves for a number of frequency bands.





*N. Alamdari, E. Lobarinas, and N. Kehtarnavaz, "An educational tool for hearing aid compression fitting via a web-based adjusted smartphone app," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, UK, May 2019.

Hearing Aid Fitting: Sample Compression Prescription

Sample hearing aid DSL-v5 prescription gains for three degrees of hearing loss in 9 frequency bands

Degree of Hearing Loss	Audiogram (dB SPL)	DSL-v5 Prescription Gains (dB SPL)	
Mild	{10, 15, 20, 25, 25, 20, 25, 25, 15}	Soft Speech: Moderate Speech: Loud Speech: MPO:	{4, 5, 10, 13, 11, 13, 25, 27, 9} {2, 4, 9, 11, 11, 16, 25, 25, 9} {1, 2, 5, 7, 6, 12, 22, 21, 7} {88, 89, 90, 92, 90, 91, 98, 95, 86}
Moderate	{60, 60, 65, 70, 70, 75, 80, 85, 85}	Soft Speech: Moderate Speech: Loud Speech: MPO:	{35, 29, 33, 36, 34, 43, 53, 58, 54} {30, 25, 30, 34, 35, 47, 55, 59, 55} {29, 21, 22, 26, 25, 36, 48, 52, 50} {103, 102, 105, 107, 106, 112, 119, 119, 113}
Severe	{60, 60, 65, 70, 85, 85, 95, 100, 110}	Soft Speech: Moderate Speech: Loud Speech: MPO:	{35, 29, 33, 36, 43, 48, 64, 69, 71} {30, 25, 30, 34, 45, 52, 66, 70, 72} {29, 21, 22, 26, 36, 43, 58, 61, 67} {103, 102, 105, 107, 114, 116, 127, 126, 127}

Turning smartphones into a testbed platform in the field or at the edge for hearing enhancement studies (2)

- Problem addressed here - Hearing aid manufacturers do not allow researchers to access codes on their hearing aid processors to run custom algorithms. As a result, one cannot easily examine hearing enhancement algorithms, in particular in the field or in real-world audio environments.

- **Solution developed in the EML Lab at UTD – Turn smartphones into a testbed platform to run hearing enhancement algorithms (including compression and noise reduction) in real-time in the field or at the edge.**

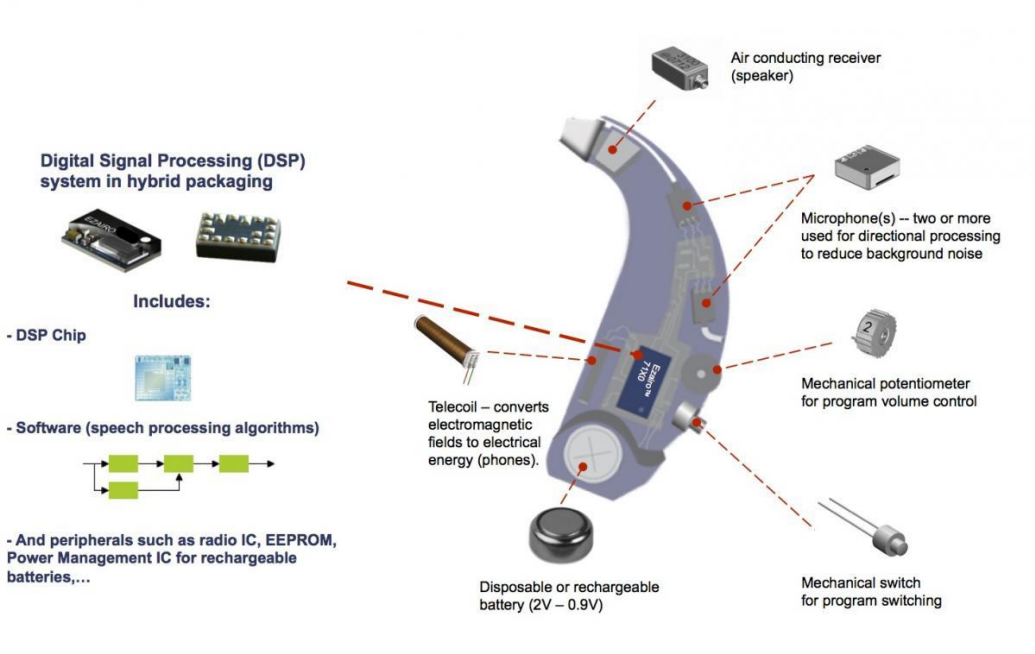


Image from SelTech website
http://www.seltech-international.com/seltech-your-one-stop-shop-hearing-aid-components

These two books which are written for signal processing and signals and systems lab courses show how to run signal processing algorithms (such as audio processing algorithms) in real-time on ARM processors of smartphones

Programming languages most widely used in signal processing: **MATLAB/C**

Android

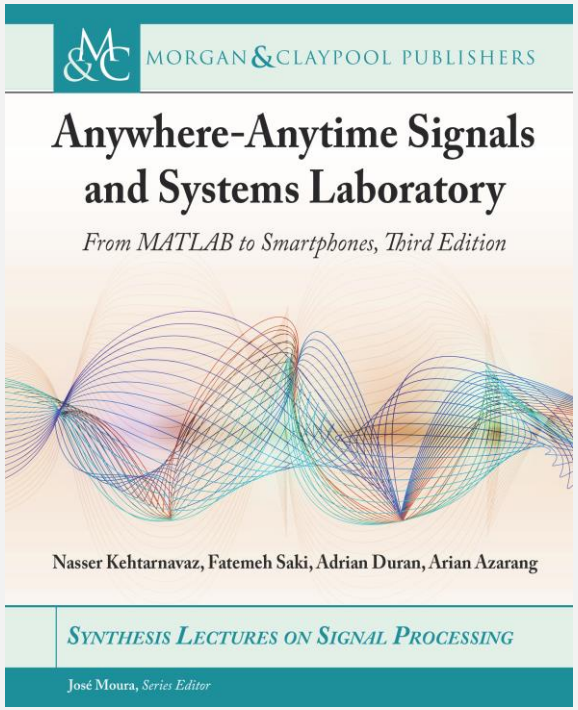
Java

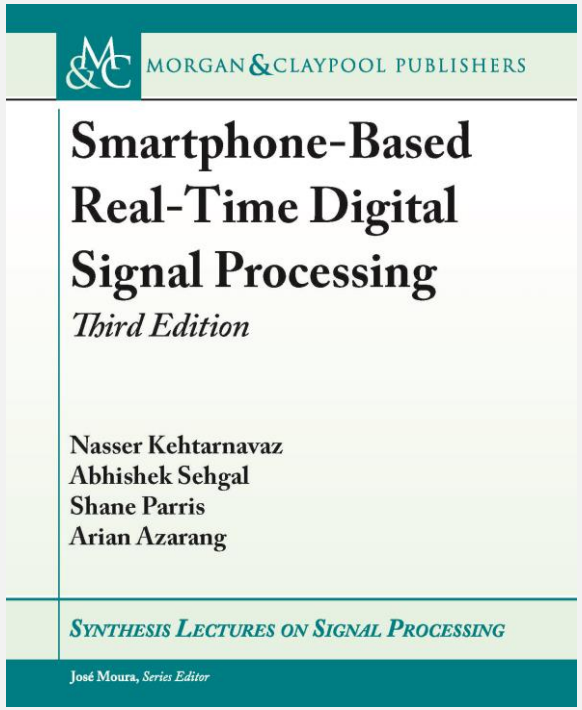
iPhone

Objective-C/Swift

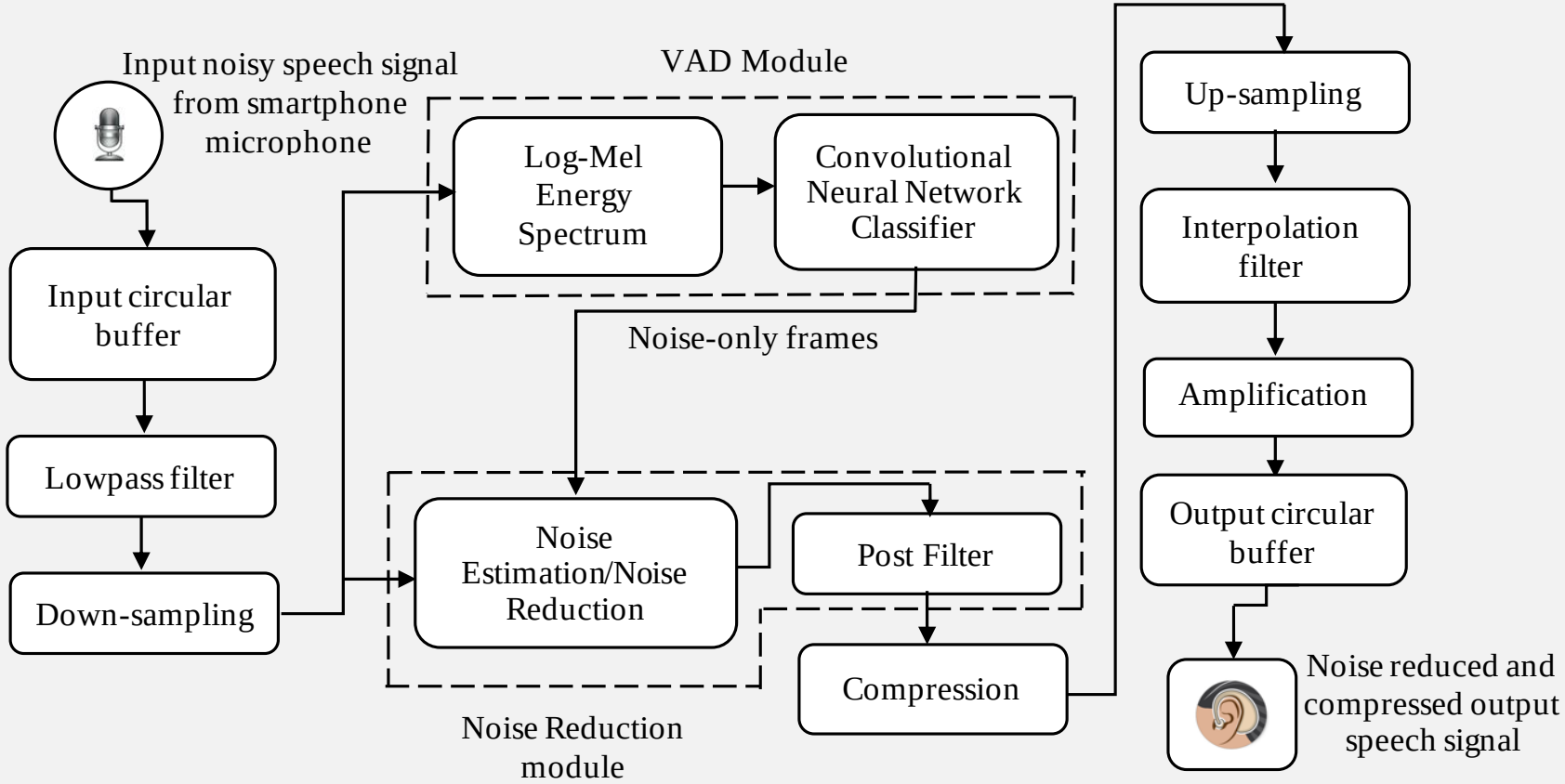
In these books, software shells are developed to enable running C/MATLAB codes within the Java and Objective-C environments of smartphones.

- These books cover how to run signal processing algorithms written in C or MATLAB on smartphones in real-time (both Android and iOS) or show how to use a smartphone as a signal processing hardware board – free download through IEEE Xplore subscription.





Smartphone-Based Real-Time Hearing Aid Pipeline or Testbed for Hearing Enhancement Studies in the Field*



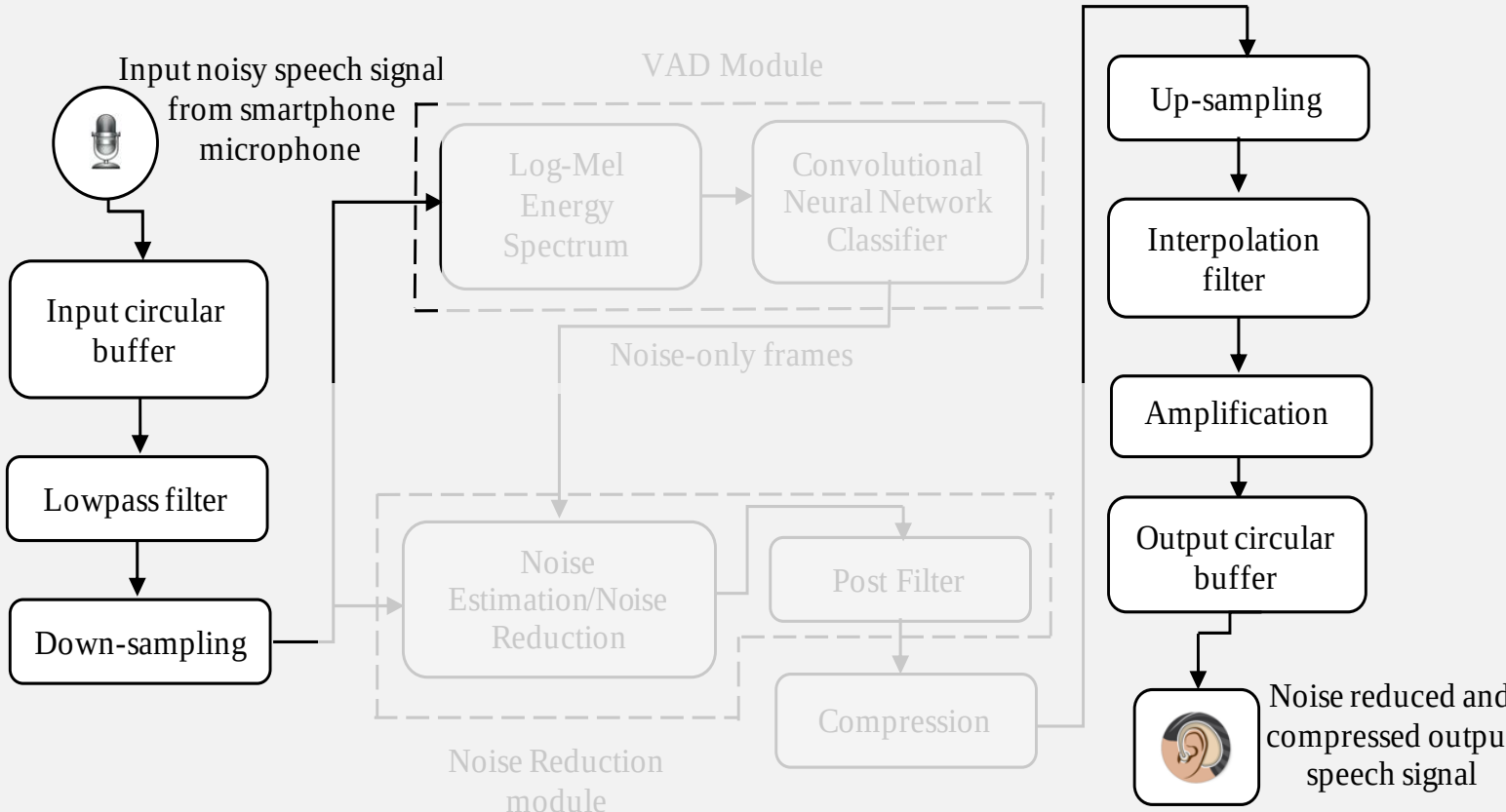
*T. Chowdhury, A. Sehgal, and N. Kehtarnavaz, "Integrating signal processing modules of hearing aids into a real-time smartphone app," *Proceedings of IEEE Engineering in Medicine and Biology Conference*, Honolulu, HI, July 2018.

I/O designed to achieve both low-latency and computational efficiency

- To have the lowest audio latency offered by smartphones (currently 10-20ms on iPhones), the audio sampling needs to be set to 48kHz. Also, the i/o frame size needs to be kept at 64 samples or 64/48000=1.3ms for iOS smartphones and 192 samples or 192/48000 = 4ms for Android (Pixel) smartphones.

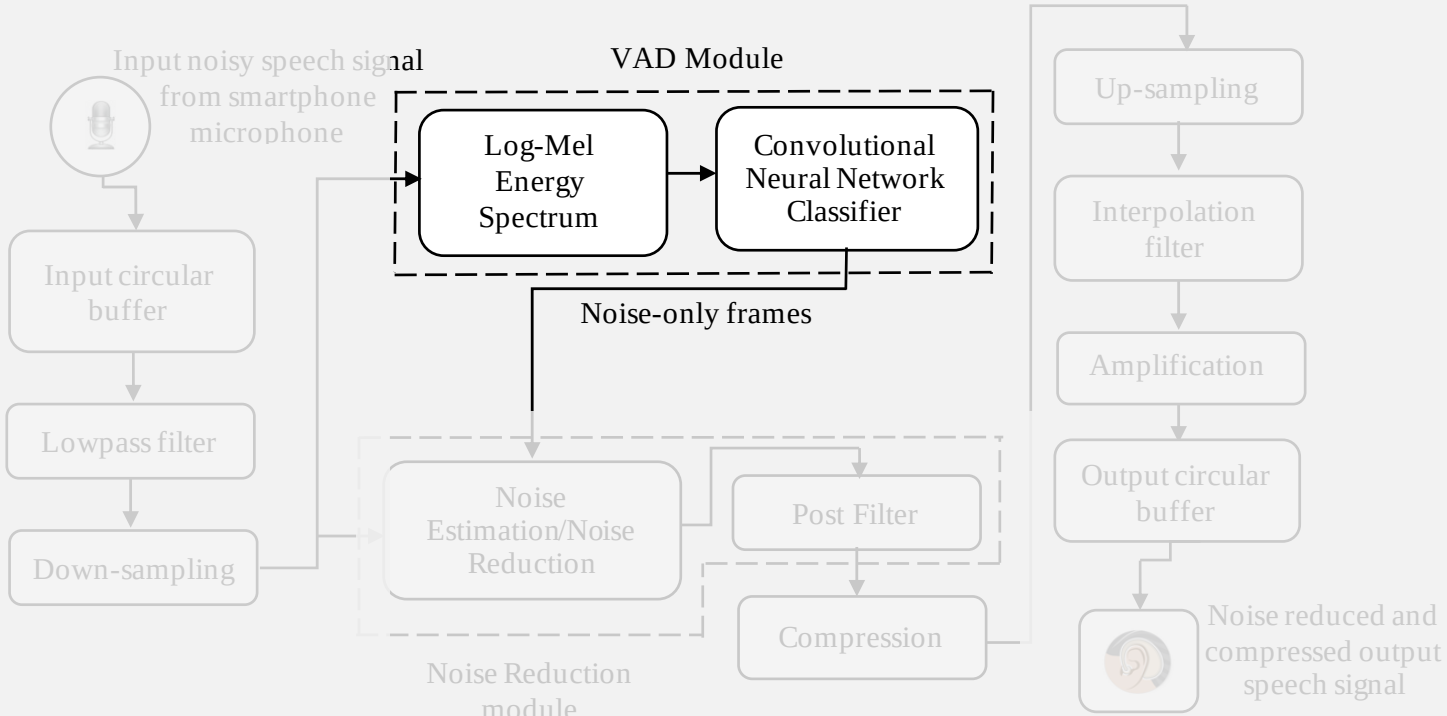
- To gain computational efficiency, frames are down-sampled and decimated by a factor of 3 (48/3 = 16kHz) for the signal processing modules.

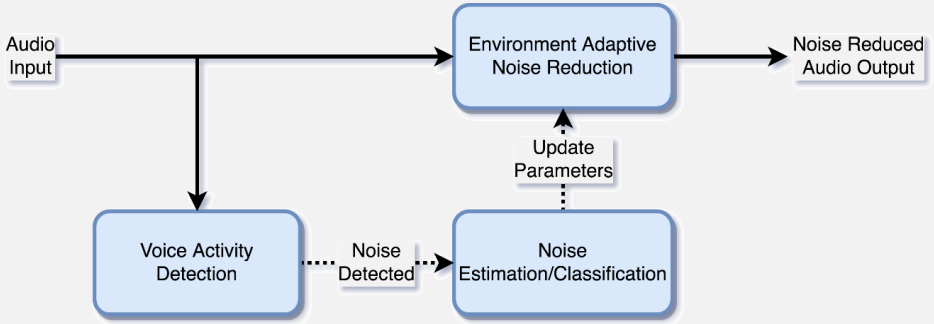
- To synchronize A/D and signal processing modules, an input circular buffer is used to collect data from the microphone till a desired frame size is reached and an output circular buffer is used to output processed frames to the speaker at the optimum i/o frame rate.



Deep Learning-Based Voice Activity Detection (1)*

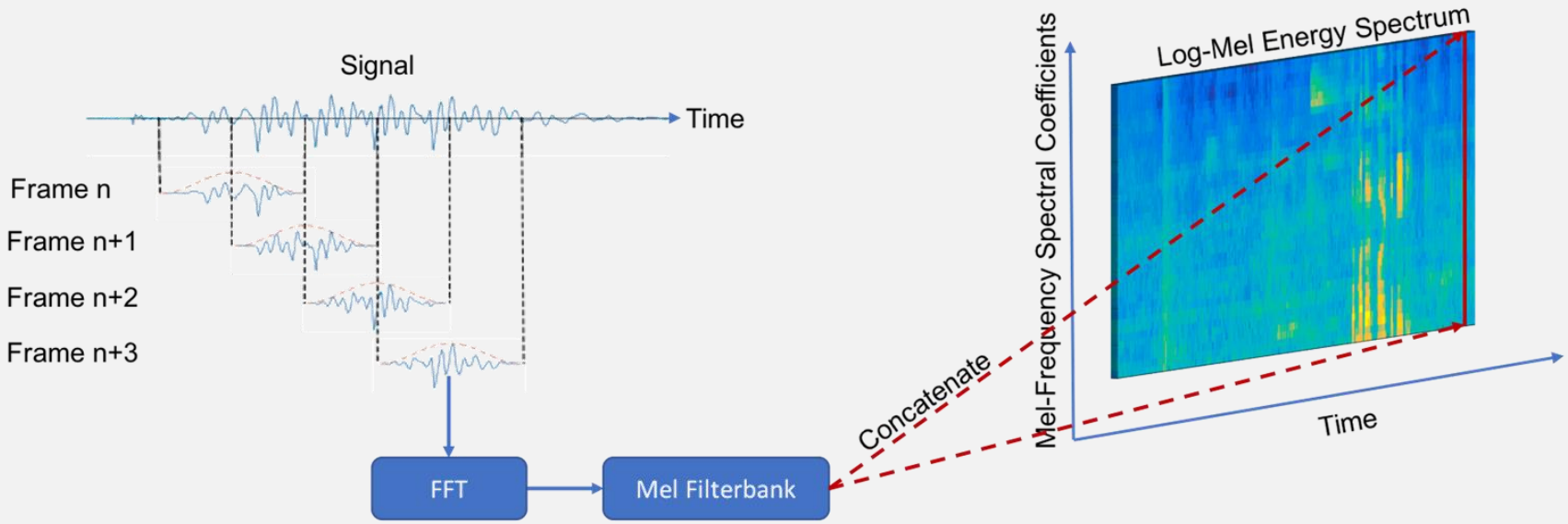
- A Voice Activity Detector (VAD) is included in the pipeline.
- VAD separates noise segments from speech+noise or noisy speech segments.
- As a result, noise reduction can be made adaptive based on the output of a noise estimator.





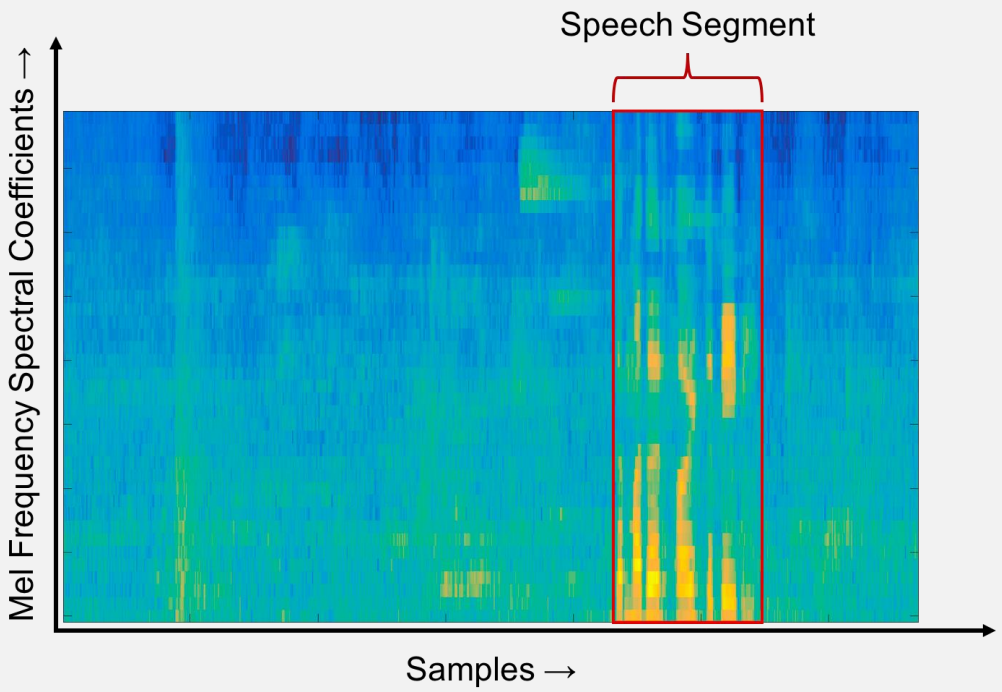
*A. Sehgal and N. Kehtarnavaz, "A convolutional neural network smartphone app for voice activity detection," *IEEE Access* (open access), vol. 6, pp. 9017-9026, Feb 2018.

Deep Learning-Based Voice Activity Detector (2)



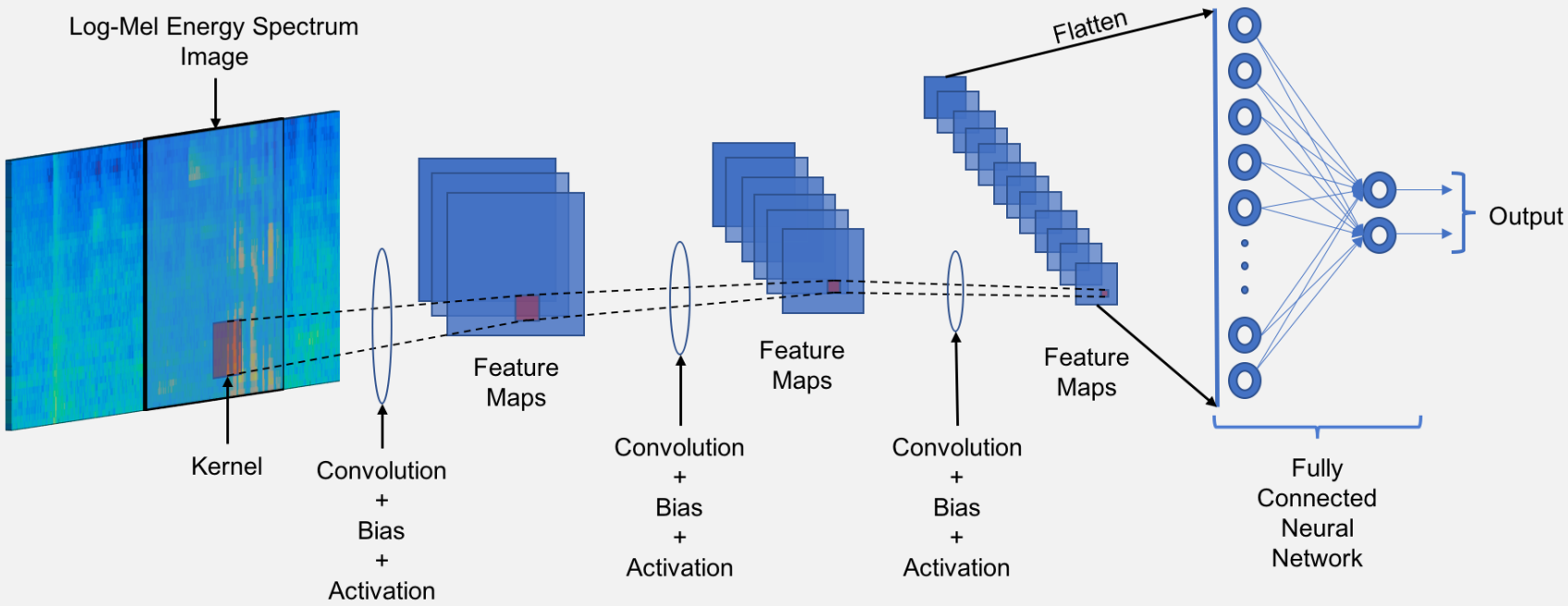
50% overlap between frames is used to form mel-frequency energy spectrum images for a duration of 0.5 sec.

Deep Learning-Based Voice Activity Detector (3)



A log-mel energy spectrum image example showing a part or segment containing speech activity.

Deep Learning-Based Voice Activity Detector (4)



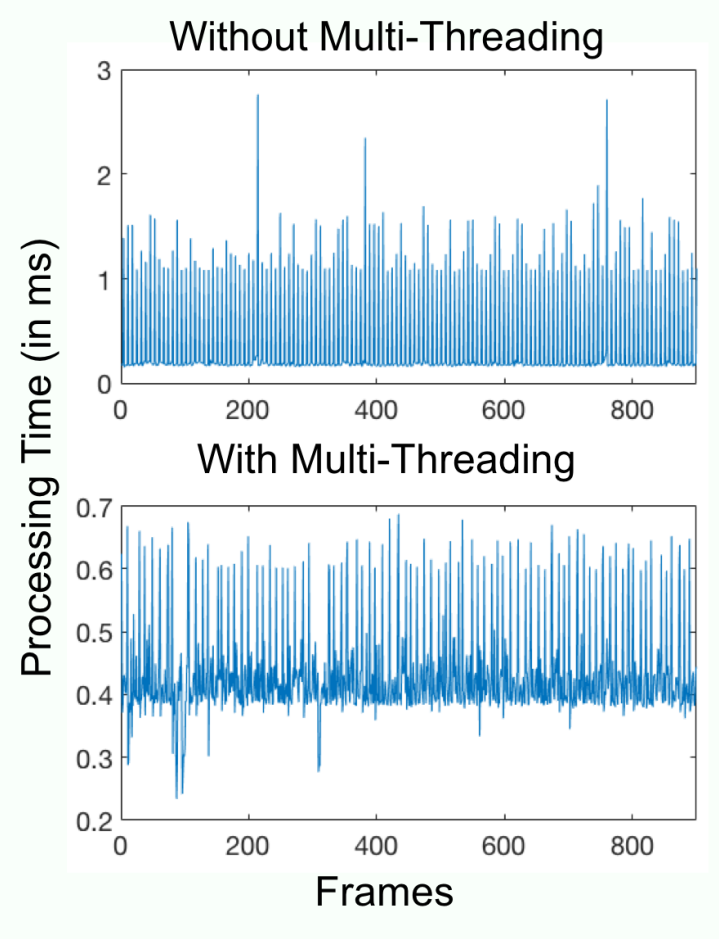
Log-mel energy spectrum images are then fed into a CNN deep learning model.

Multi-threading to achieve real-time

- Without multithreading, the frame processing time crossed 1.3ms on iOS and 4ms on Android (Pixel) smartphone, i.e. frames got skipped and real-time throughput could not be met.

- With multi-threading, the timings remained within these time limits.

- This was crucial as to use this VAD with other hearing aid signal processing modules, the multi-threading approach freeed up computational bandwidth on the main thread.



Deep Learning-Based VAD Results

Lab Testing - Average Noise Hit Rate (NHR) (in %)

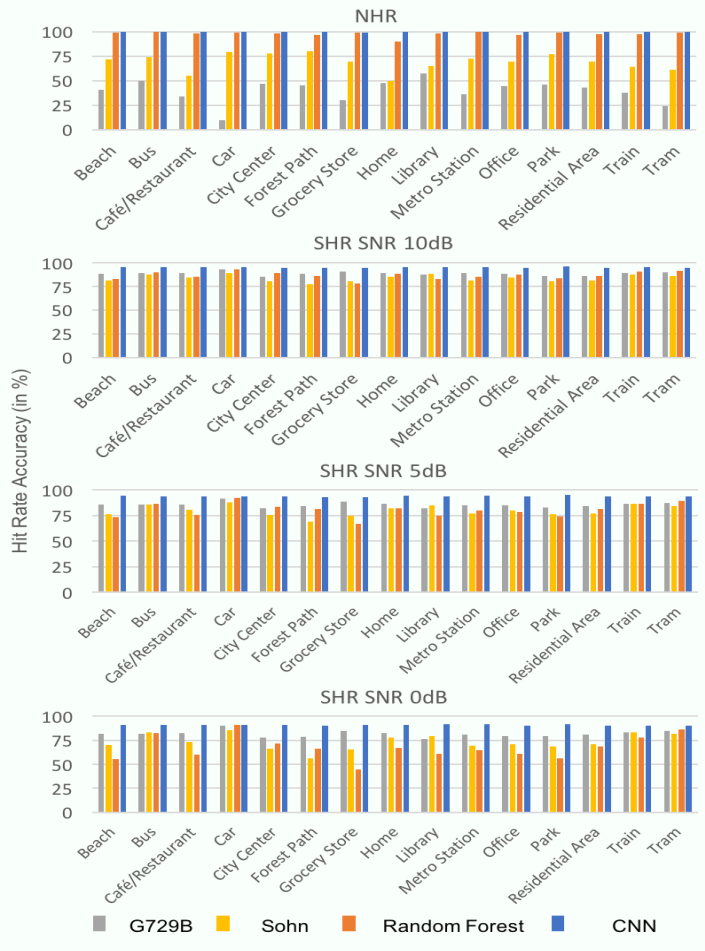
	G729B	Sohn	Random Forest	CNN
NHR	39.3	69	97.4	99.3

Lab Testing - Average Speech Hit Rate (SHR) (in %)

SNR (dB)	G729B	Sohn	Random Forest	CNN
10	88.8	83.9	85.6	94.8
5	85.4	79.8	78.9	92.8
0	81.8	73.5	66.7	90.0

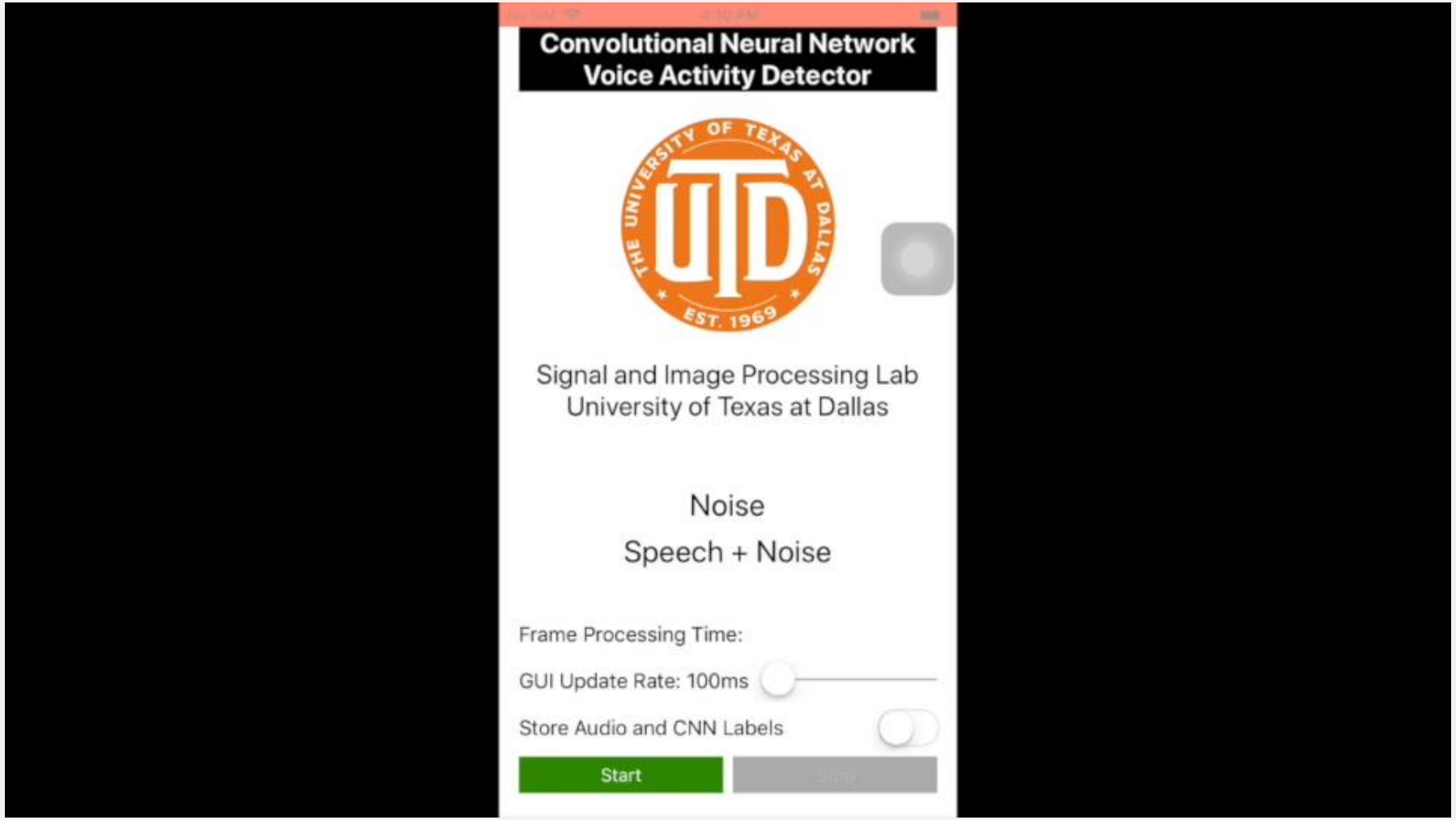
Field Testing - Average NHR and SHR (in %)

	G729B	Sohn	Random Forest	CNN
NHR	63.6	27.9	98.9	99
SHR	86.9	97.3	86.4	91.3

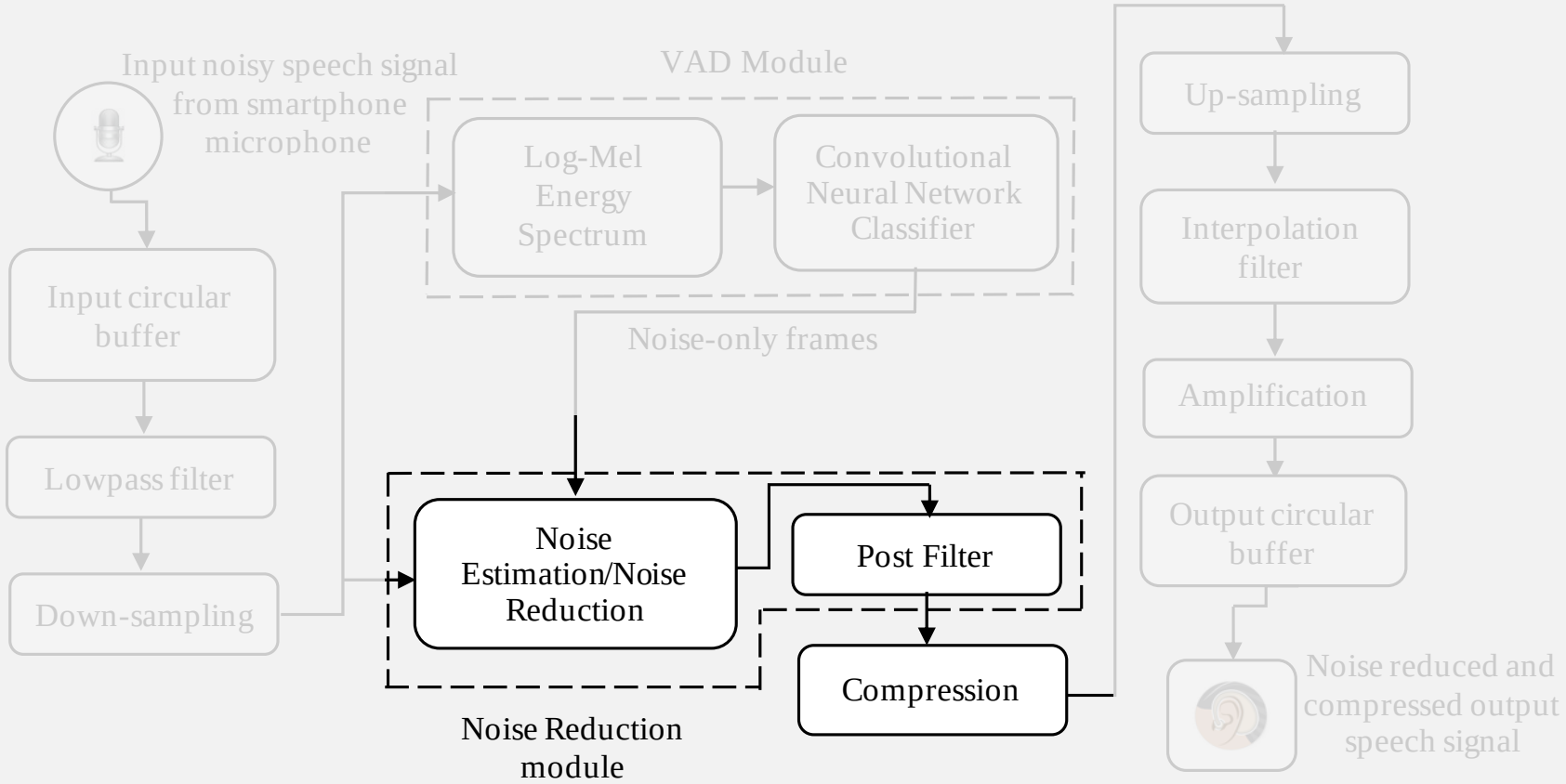


SHR and NHR for the 4 VADs in different noise environments

Videoclip (VAD)

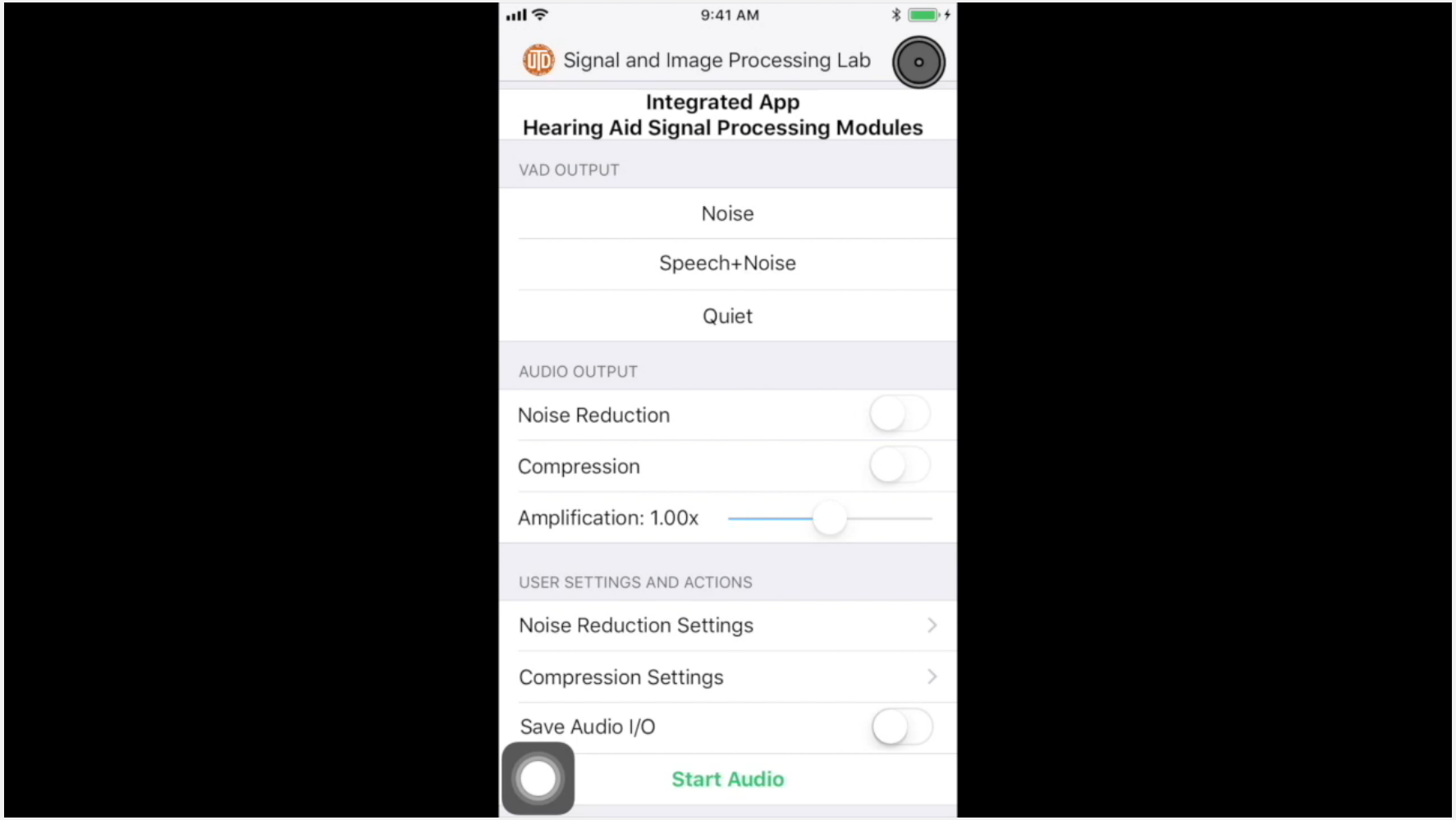


Smartphone-Based Real-Time Hearing Aid Pipeline or Testbed for Hearing Enhancement Studies in the Field: Noise Reduction and Compression Modules*



*N. Alamdari and N. Kehtarnavaz, "A real-time personalized noise reduction smartphone app for hearing enhancement," *Proceedings of IEEE Signal Processing in Medicine and Biology Symposium*, Philadelphia, PA, Dec 2018.

Videoclip (Integrated Pipeline)

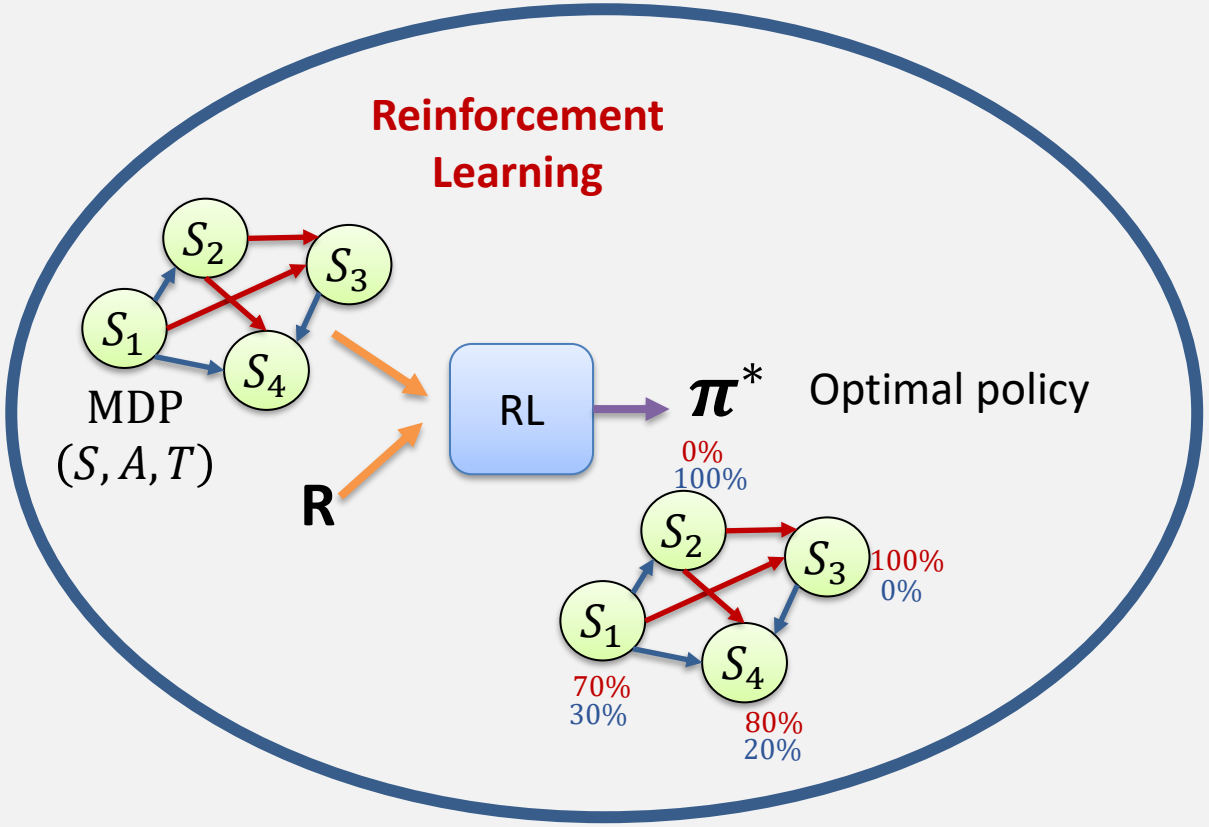


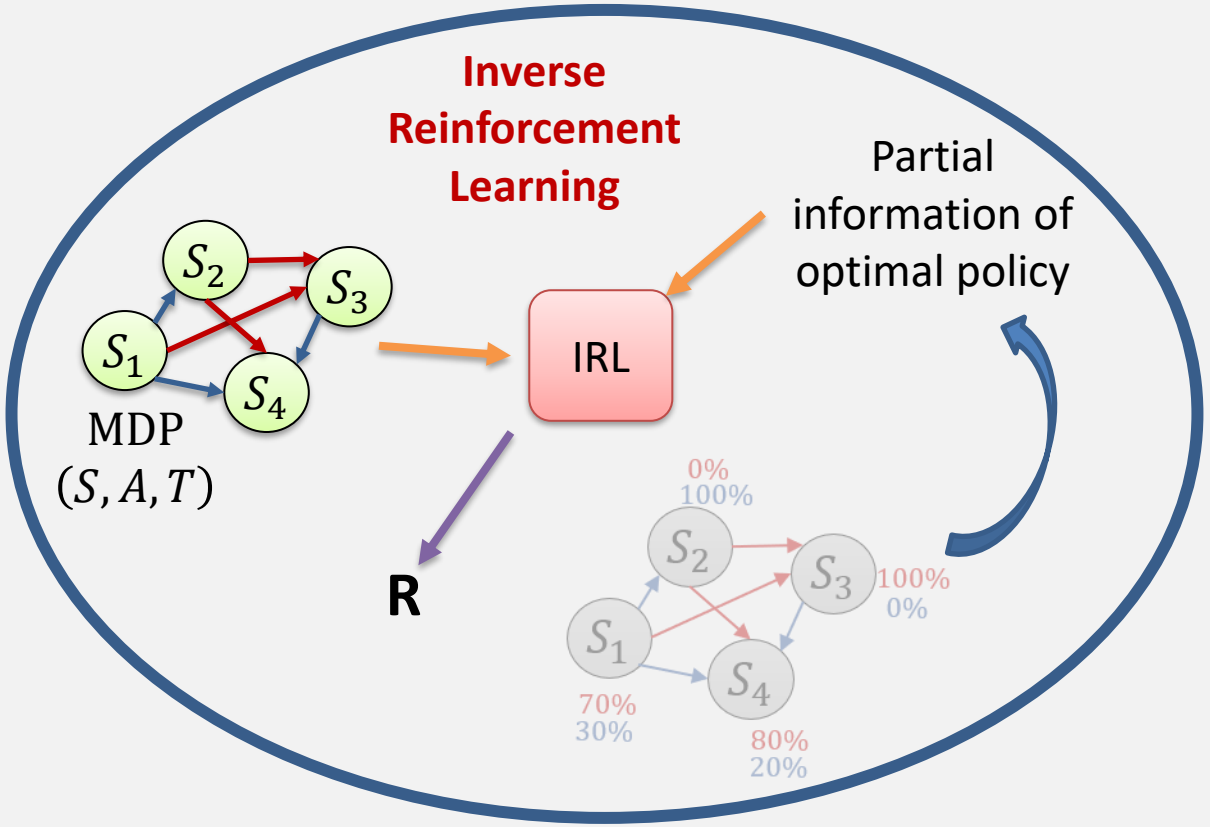
Latest work: Personalization of compression

- It is documented that nearly 50% of hearing aid users prefer amplification settings that differ from those prescribed to them.

- To improve user satisfaction from hearing aids, our latest research aims to personalize compression/amplification settings in real-world audio environments by using inverse reinforcement learning.

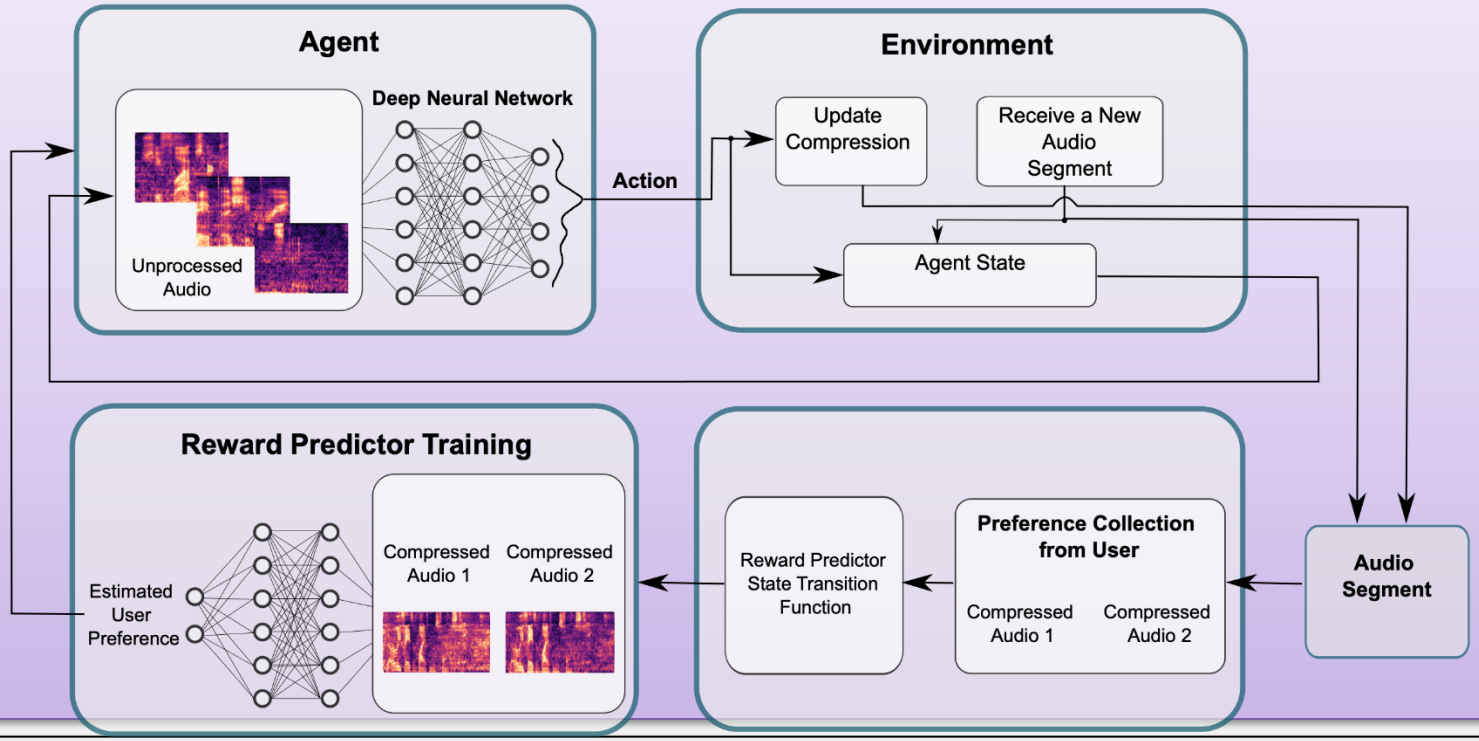
Inverse Reinforcement Learning



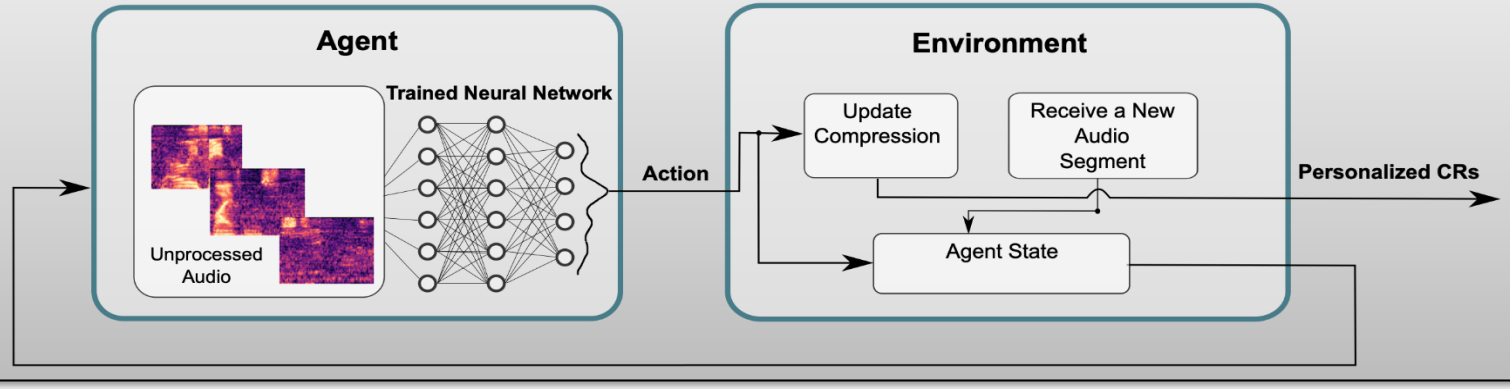


Personalized compression achieved by using deep reinforcement learning (DRL)*

Training Mode →



Testing/Operation Mode →

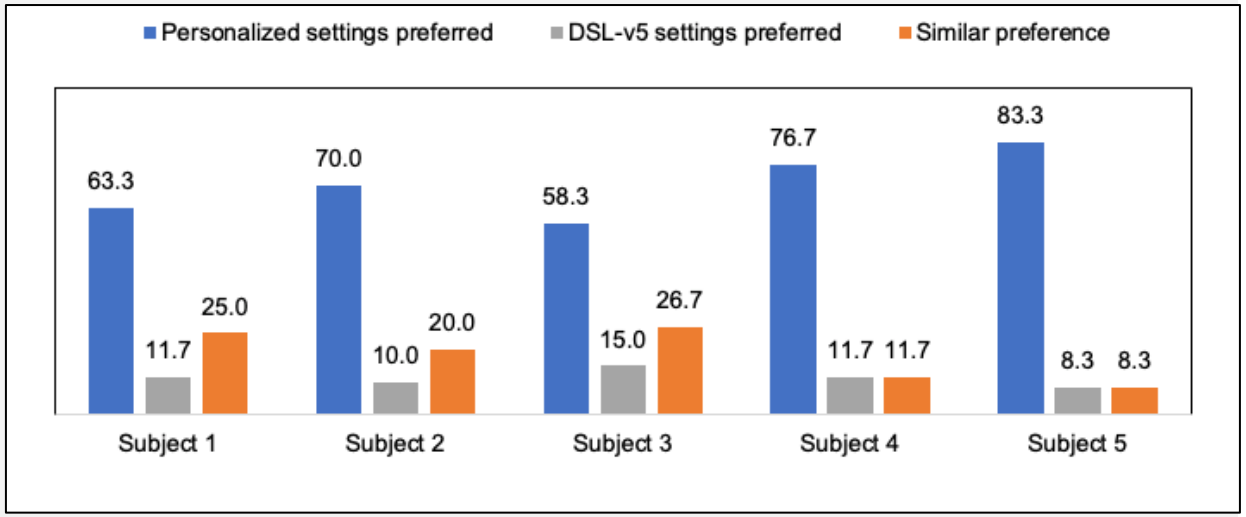


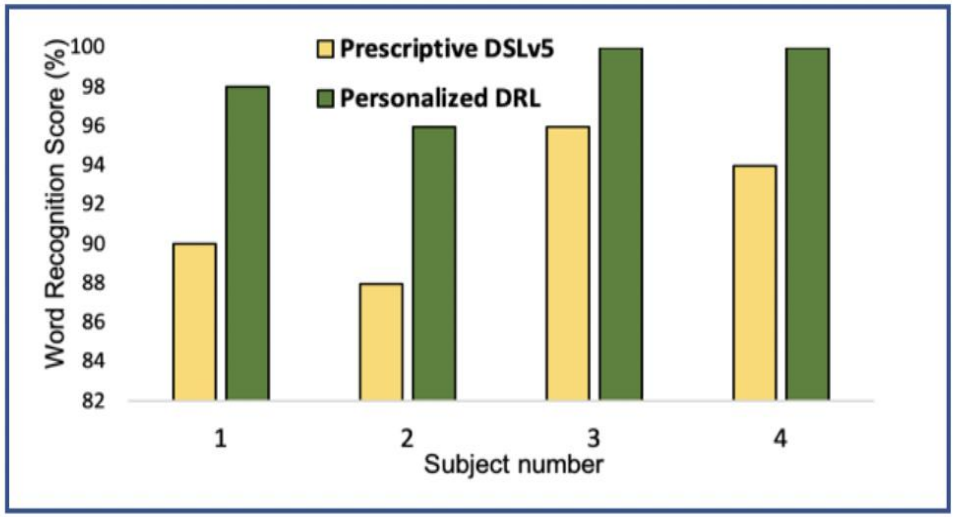
*N. Alamdari, E. Lobarinas, N. Kehtarnavaz, "Personalization of Hearing Aid Compression by Human-in-the-Loop Deep Reinforcement Learning," *IEEE Access*, vol. 8, pp. 203503-203515, 2020.

Subject Testing Results of DRL (based on an approved IRB)

Subject testing experiments: preference percentages
of personalized compression vs. prescriptive DSL-v5 compression

Subject	Audiogram in freq. bands [0.5, 1.0, 2.0, 4.0, 6.0] kHz	DSL-v5 gains for soft speech	DSL-v5 compression ratios	Personalized compression ratios
1	[15, 20, 20, 30, 30]	[7, 8, 14, 17, 15]	[1.1, 1.2, 1.3, 1.2, 1.3]	[1.1, 1.2, 1.3, 4.8, 5.2]
2	[15, 15, 20, 20, 30]	[5, 6, 14, 15, 15]	[1.1, 1.2, 1.3, 1.2, 1.2]	[4.4, 1.2, 5.2, 1.2, 4.8]
3	[20, 20, 40, 50, 60]	[11, 12, 24, 29, 34]	[1.1, 1.2, 1.3, 1.2, 1.4]	[4.4, 1.2, 1.3, 4.8, 5.6]
4	[25, 20, 20, 40, 30]	[13, 11, 14, 22, 15]	[1.1, 1.3, 1.3, 1.3, 1.3]	[4.4, 1.3, 1.3, 5.2, 1.3]
5	[20, 20 ,30 ,40 ,40]	[6, 11, 20, 23, 20]	[1.1, 1.2, 1.3, 1.2, 1.4]	[1.1, 1.2, 1.3, 4.8, 5.6]





Latest Work:

Maximum Likelihood Inverse Reinforcement Learning

(enables online or on-the-fly training in real-world audio environments)

Reward $R_W(s, a) = W^T \varphi(s, a)$

State-action function (known)

$\varphi: S \times A \rightarrow \mathbb{R}^m$

reward for executing action a in state s

Weighting vector of state-action function (unknown)

$CR_{updated}(i) = CR_{initial}(i) \times scale(i)$

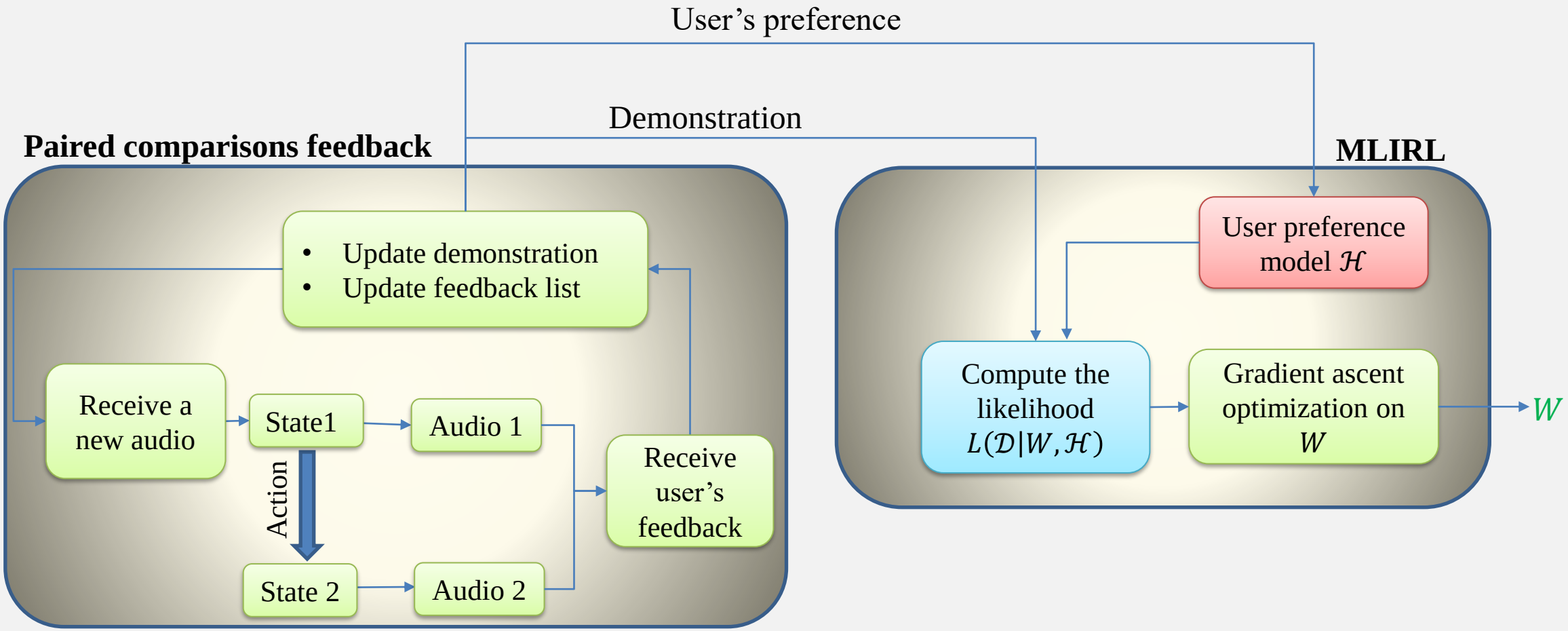
States: Different sets of compression ratios (CRs)
Actions: Applying scales for changing CRs

Demonstration $\mathcal{D} = \{(s_1, a_1), (s_2, a_2), \ldots\}$

Demonstration denotes a set of state-action pairs

Block diagram of developed personalized compression for field (edge) deployment

MLIRL finds the reward by maximizing the likelihood of actions based on a personalized preference model



*S. Akbarzadeh, E. Lobarinas, N. Kehtarnavaz, "Online Personalization of Compression in Hearing Aids via Maximum Likelihood Inverse Reinforcement Learning," to appear in *IEEE Access*, 2022.

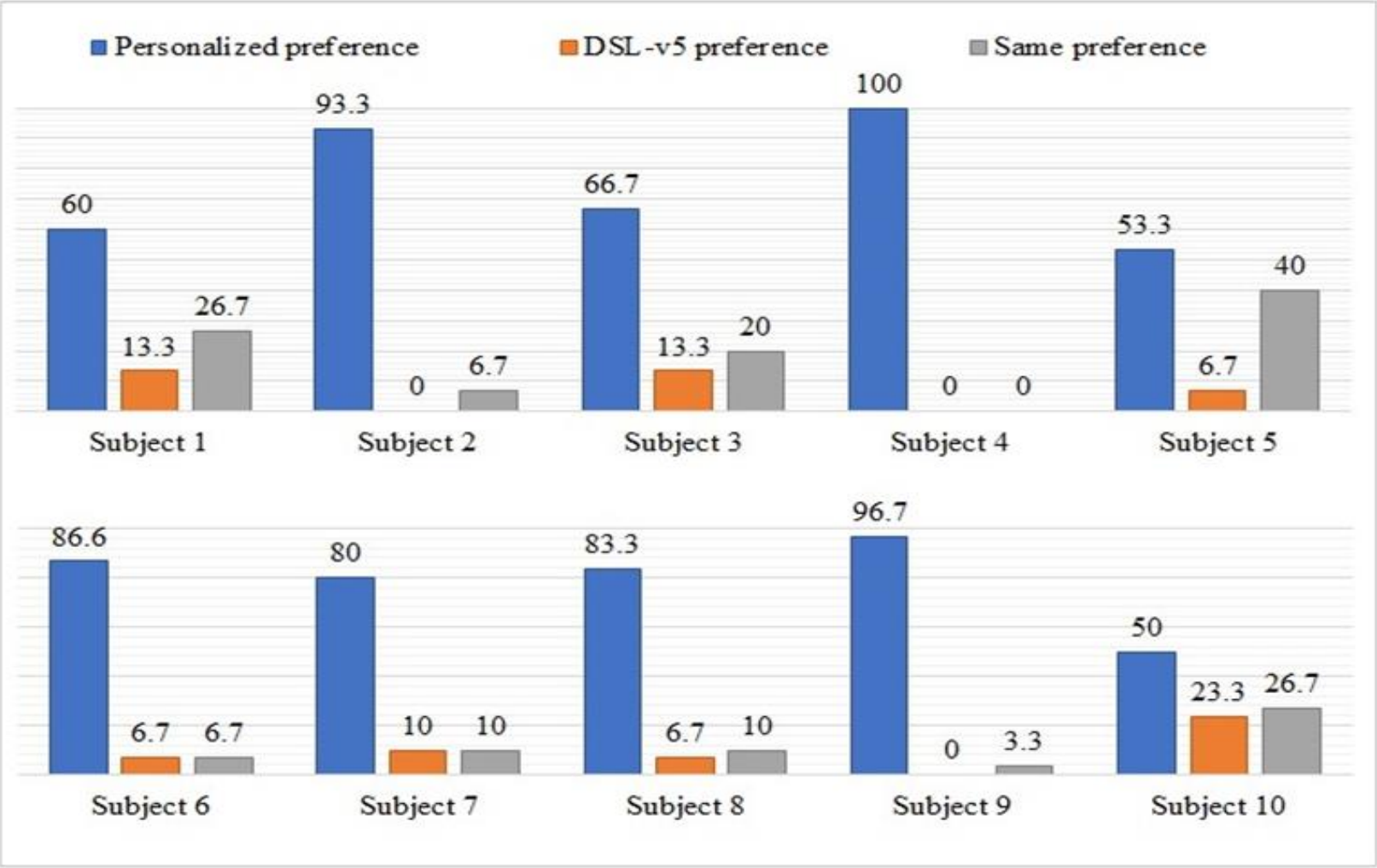
Subject Testing Results of MLIRL (1)

Prescriptive DSLv5 and personalized compression settings for ten subjects participated in the study

Subject	Audiogram for freq. bands [0.5, 1, 2, 4, 6] kHz	DSLv5 gains for soft speech	DSLv5 compression ratios	Personalized compression ratios
1	[15, 15, 15, 25, 40]	[5, 8, 10, 23, 25]	[1.2, 1.2, 1.3, 1.2, 1.4]	[1.2, **4.8**, 1.3, **4.8**, **5.6**]
2	[15, 15, 20, 20, 30]	[5, 6, 14, 15, 15]	[1.2, 1.2, 1.3, 1.2, 1.3]	[**4.8**, **4.8**, **5.2**, 1.2, 1.3]
3	[20, 25, 30, 50, 45]	[10, 12, 19, 36, 29]	[1.2, 1.3, 1.4, 1.5, 1.4]	[**4.8**, 1.3, 1.4, 1.5, 1.4]
4	[25, 30, 30, 40, 45]	[11, 14, 19, 29, 29]	[1.2, 1.3, 1.4, 1.3, 1.4]	[**4.8**, **5.2**, **5.6**, 1.3, 1.4]
5	[15, 15, 15, 30, 30]	[8, 9, 10, 25, 20]	[1.2, 1.3, 1.3, 1.3, 1.3]	[1.2, 1.3, 1.3, 1.3, **5.2**]
6	[5, 10, 25, 30, 40]	[1, 5, 17, 25, 25]	[1.2, 1.2, 1.4, 1.3, 1.4]	[**4.8**, **4.8**, 1.4, 1.3, **5.6**]
7	[15, 25, 20, 30, 35]	[11, 12, 13, 25, 22]	[1.2, 1.3, 1.3, 1.3, 1.3]	[**4.8**, 1.3, **5.2**, **5.2**, **5.2**]
8	[10, 20, 20, 50, 40]	[7, 8 13, 34, 25]	[1.2, 1.2, 1.3, 1.4, 1.4]	[**4.8**, 1.2, **5.2**, 1.4, **5.6**]
9	[20, 20, 25, 35, 25]	[9, 10, 17, 32, 17]	[1.2, 1.3, 1.4, 1.5, 1.2]	[**4.8**, **5.2**, **5.6**, 1.5, 1.2]
10	[15, 20, 25, 35, 40]	[9, 10, 17, 29, 25]	[1.2, 1.3, 1.4, 1.4, 1.4]	[**4.8**, **5.2**, 1.4, 1.4, **5.6**]

Subject Testing Results of MLIRL (2)

Preference percentages of personalized MLIRL compression vs. standard DSL-v5 compression –
On average, personalized compression was preferred about 10 times more than standard compression



Summary (signal processing application)

- A smartphone-based virtual hearing aid has been developed in order to be able to study custom hearing enhancement algorithms in the field or at the edge.

- ***In other words, this work has enabled smartphones to be used as a testbed platform for studying custom hearing enhancement algorithms in real-world audio environments.***