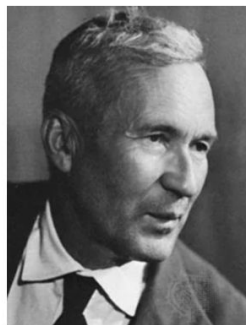
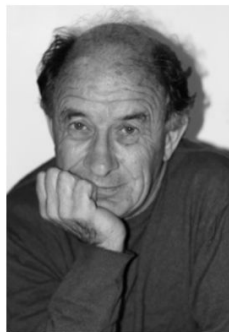


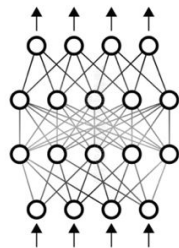
Kolmogorov-Arnold Networks in audio anti-spoofing



+



+



+

AASIST

2024

Agenda:

- Kolmogorov-Arnold Representation theorem
- B-spline
- KAT Generalization
- Implementation details
- KAN Scaling laws
- Using KANs for anti-spoofing
- Practical tips for improving spoofing detectors

MLP – foundation block

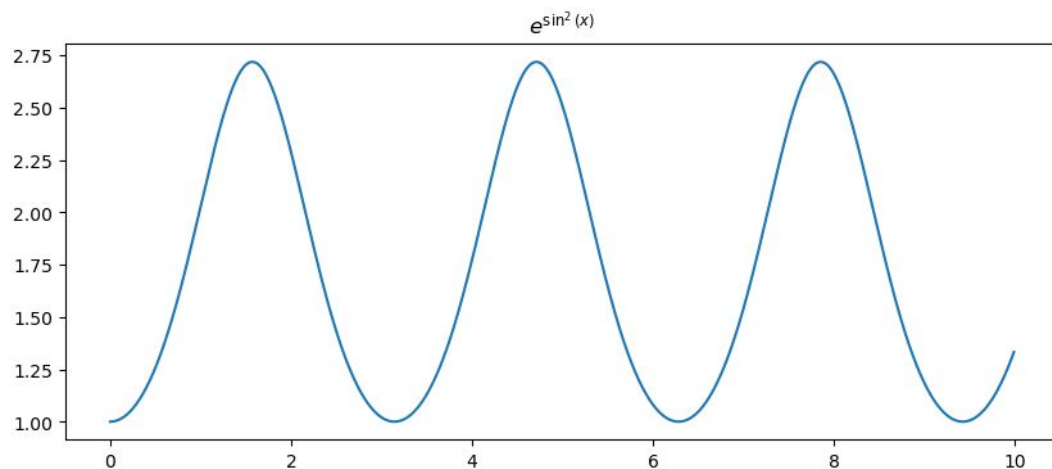
3. Cybenko Theorem(UAT):

For any continuous function $f: [0, 1]^n \rightarrow \mathbb{R}$ and any $\epsilon > 0$, there exists a ANN with one hidden layer, activation σ and a finite number of neurons N , such that

$$\left| f(x) - \sum_{i=1}^N \alpha_i \sigma(w_i * x + b_i) \right| < \epsilon$$

MLP optimization challenges

- Low-dimensional functions
- Oscillating functions
- Invariant functions



It is difficult for MLPs to handle audio signals efficiently

Kolmogorov-Arnold representation theorem

• If f is a multivariate continuous function on bounded domain, then f can be written as finite composition of continuous functions of a single variable and the binary operation of addition. For a smooth $f: [0,1]^n \rightarrow \mathbb{R}$

$$f(x) = f(x_1, \dots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^n \phi_{q,p}(x_p) \right)$$

$$\phi_{q,p}: [0,1] \rightarrow \mathbb{R}, \Phi_q: \mathbb{R} \rightarrow \mathbb{R}$$

- $$f(x) = f(x_1, \dots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^n \phi_{q,p}(x_p) \right)$$

$$f(x_1, x_2) = 4x_1x_2$$

$$\begin{aligned} \phi_{1,1}(z) &= z, & \phi_{1,2}(z) &= z, \\ \phi_{2,1}(z) &= z, & \phi_{2,2}(z) &= -z \end{aligned}$$

$$\Phi_1(z) = z^2, \quad \Phi_2(z) = -z^2$$

$$\begin{aligned} \phi_{i,1}(z) &= 0, & \phi_{i,2}(z) &= 0, & \Phi_i(z) &= 0 \\ i &\in \{3, \dots, 2n+1\} \end{aligned}$$

$$f(x) = f(x_1, \dots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^n \phi_{q,p}(x_p) \right)$$

$$q = 1: \Phi_q \left(\sum_{p=1}^2 \phi_{q,p}(x_p) \right) = \Phi_1 \left(\phi_{1,1}(x_1) + \phi_{1,2}(x_2) \right) = (x_1 + x_2)^2$$

$$q = 2: \Phi_q \left(\sum_{p=1}^2 \phi_{q,p}(x_p) \right) = \Phi_2 \left(\phi_{2,1}(x_1) + \phi_{2,2}(x_2) \right) = -(x_1 - x_2)^2$$

$$f(x_1, x_2) = \sum_{q=1}^{2n+1} \Phi_q(\dots) = (x_1 + x_2)^2 - (x_1 - x_2)^2 = 4x_1x_2$$

KAT: problems

- There are no restrictions on one-dimensional functionals: they may not be smooth or even fractal
- The theorem is described only for the two-layer model
- How to get such functions

Solution of non-smoothness - Splines

A spline of degree k – piecewise polynomial function $S(x)$ defined on an interval $[a, b]$ with breakpoints $x_0 < x_1 < \dots < x_n$ such that on each subinterval $[x_i, x_{i+1}]$, $S(x)$ is a polynomial of degree at most k and belongs to the class $C^{k-1}([a, b])$

$$S(x) = f(x) = \begin{cases} P_0(x), & x \in [x_0, x_1] \\ P_1(x), & x \in [x_1, x_2] \\ \vdots & \\ P_n(x), & x \in [x_{n-1}, x_n] \end{cases}$$

B-splines

B-splines are linear combinations of basis functions, each of which depends on only a limited number of nodes, making B-splines local and controllable.

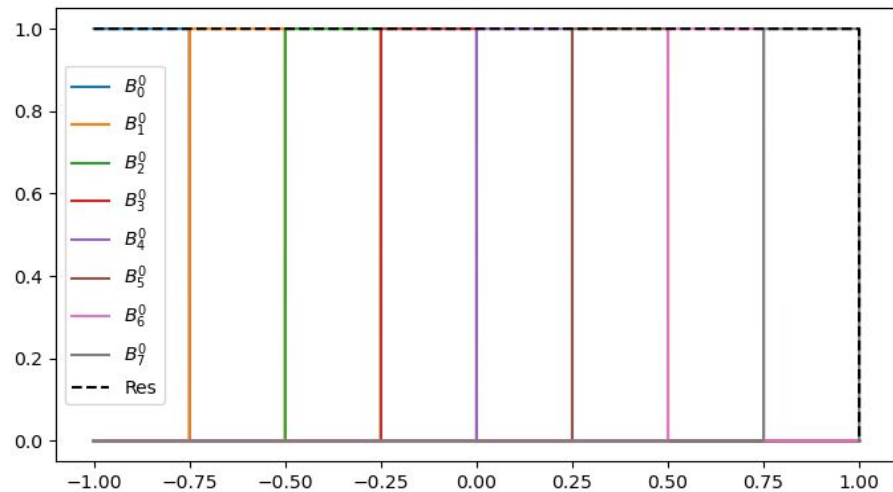
de Boor formula:

$$B_i^0(x) = \begin{cases} 1, & x_i \leq x \leq x_{i+1} \\ 0, & \text{otherwise} \end{cases}$$

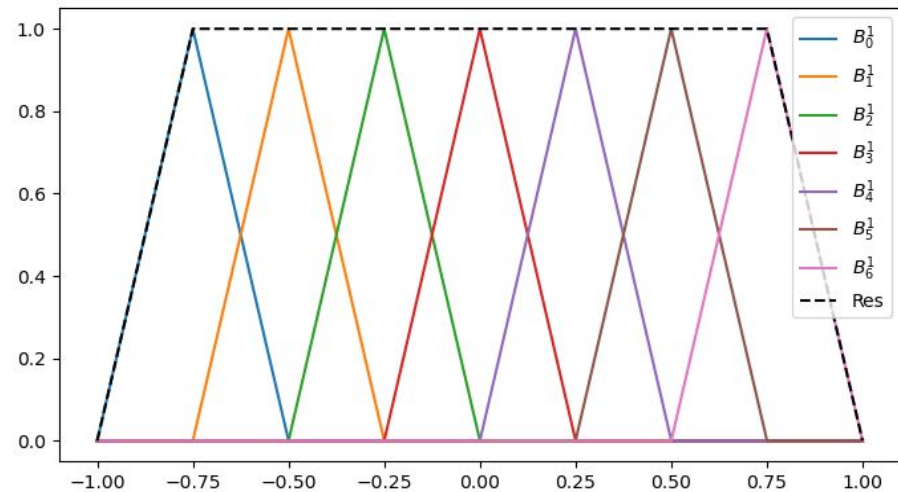
$$B_i^k(x) = \frac{x - x_i}{x_{i+k} - x_i} B_i^{k-1}(x) + \frac{x_{i+k+1} - x}{x_{i+k+1} - x_{i+1}} B_{i+1}^{k-1}(x)$$

$$\text{spline}(x) = \sum_i c_i B_i^k$$

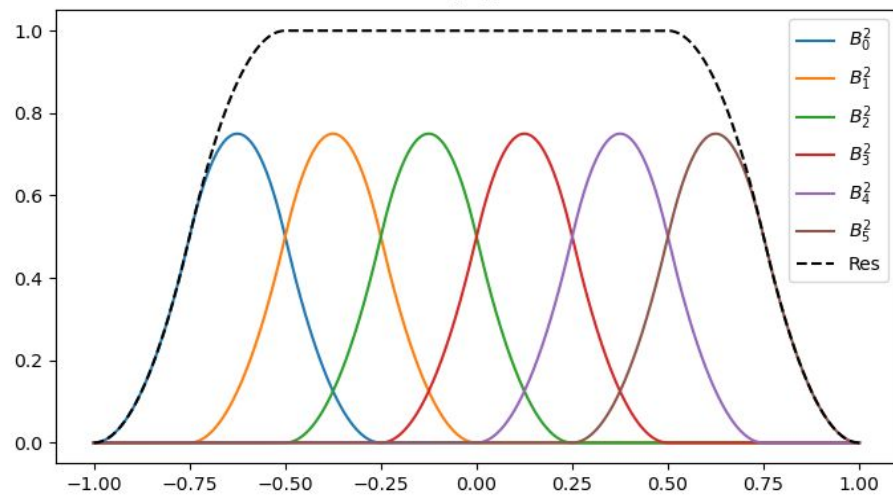
k=0



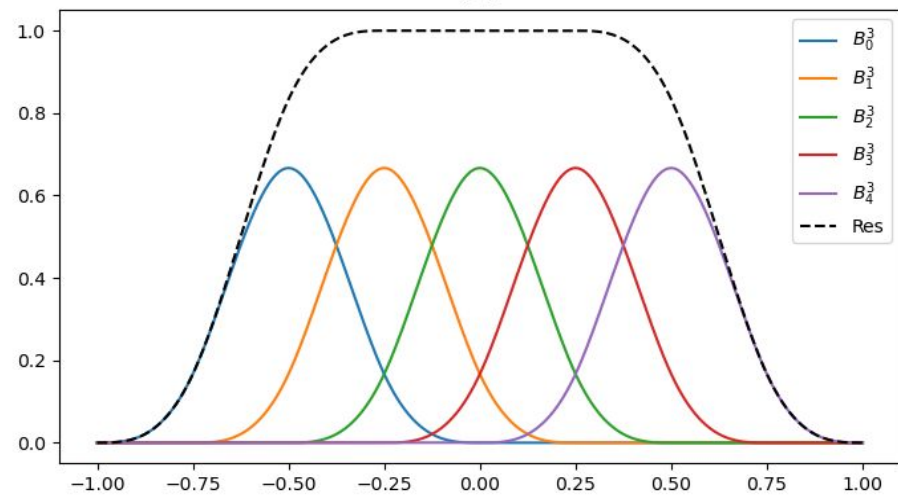
k=1



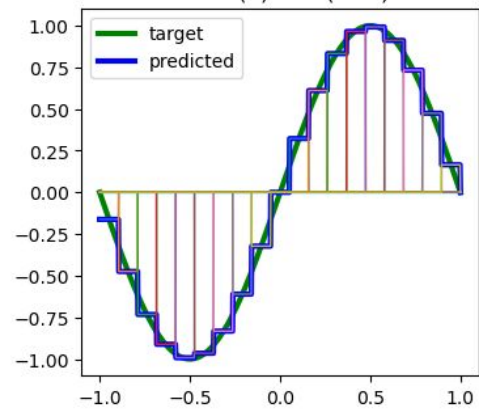
k=2



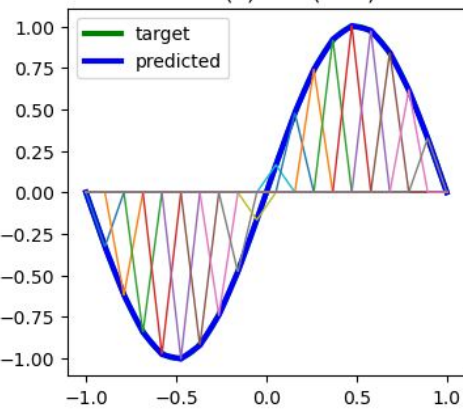
k=3



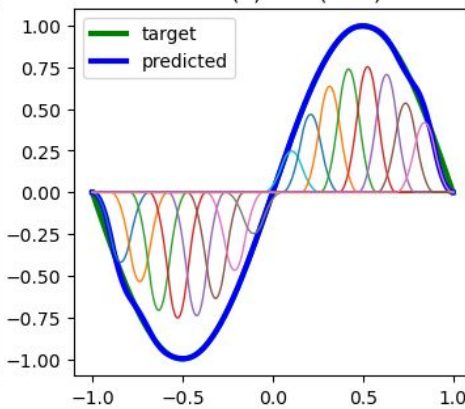
$k=0$ $f(x) = \sin(\pi * x)$



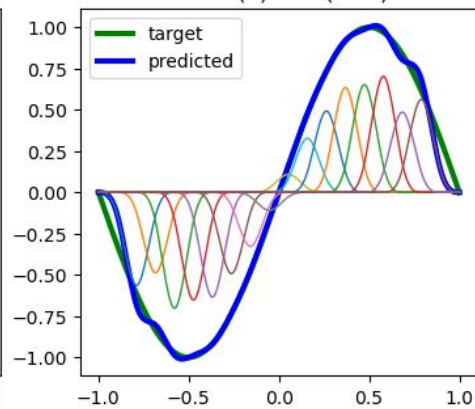
$k=1$ $f(x) = \sin(\pi * x)$



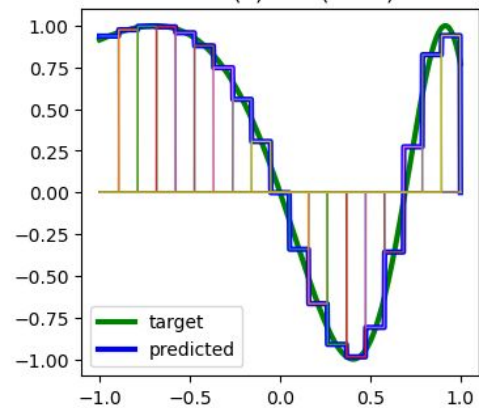
$k=2$ $f(x) = \sin(\pi * x)$



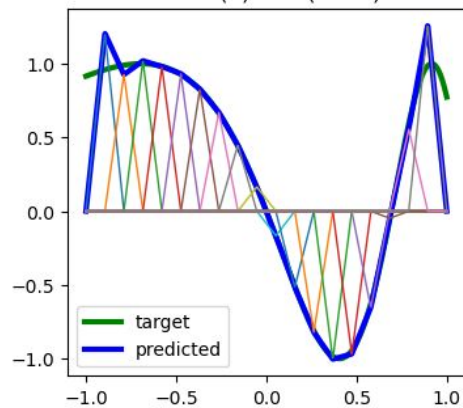
$k=3$ $f(x) = \sin(\pi * x)$



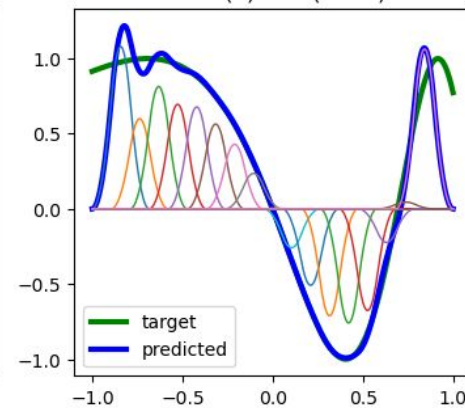
$k=0$ $f(x) = \sin(\pi * e^x)$



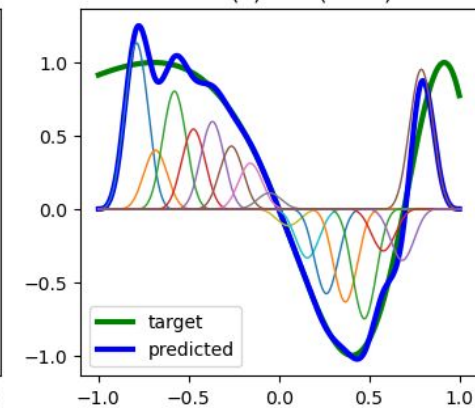
$k=1$ $f(x) = \sin(\pi * e^x)$



$k=2$ $f(x) = \sin(\pi * e^x)$



$k=3$ $f(x) = \sin(\pi * e^x)$



KAT: what is KAN layer?

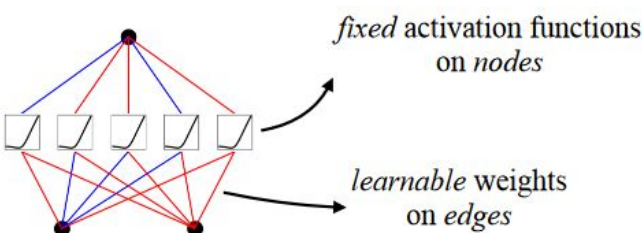
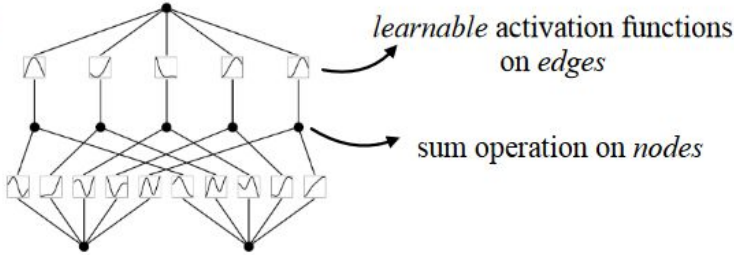
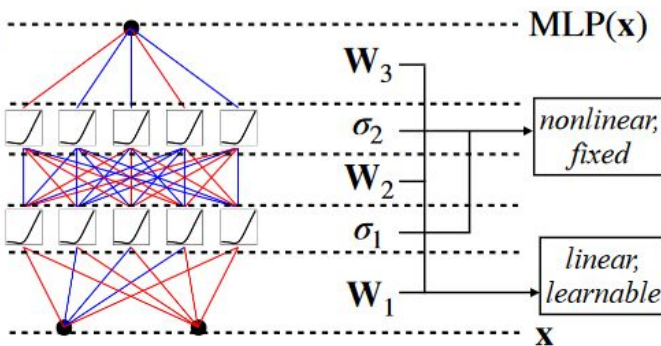
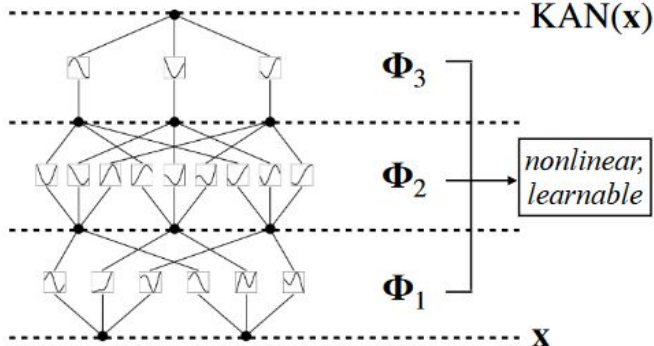
$$f(x) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^n \phi_{q,p}(x_p) \right)$$

Layer hid dims:

$$[n_0, n_1, \dots, n_L]$$

$$\Phi = \phi_{l,q,p}, \quad l = 0, \dots, L-1, \quad p = 1, 2, n_{in}, \quad q = 1, 2, n_{out}$$

$$x_{l+1} = \begin{pmatrix} \phi_{l,1,1}(\cdot) & \phi_{l,1,1}(\cdot) & \dots & \phi_{l,1,1}(\cdot) \\ \phi_{l,1,1}(\cdot) & \phi_{l,1,1}(\cdot) & \dots & \phi_{l,1,1}(\cdot) \\ \vdots & \vdots & \ddots & \dots \\ \phi_{l,1,1}(\cdot) & \phi_{l,1,1}(\cdot) & \dots & \phi_{l,1,1}(\cdot) \end{pmatrix} x_l$$

Model	Multi-Layer Perceptron (MLP)	Kolmogorov-Arnold Network (KAN)
Theorem	Universal Approximation Theorem	Kolmogorov-Arnold Representation Theorem
Formula (Shallow)	$f(\mathbf{x}) \approx \sum_{i=1}^{N(\epsilon)} a_i \sigma(\mathbf{w}_i \cdot \mathbf{x} + b_i)$	$f(\mathbf{x}) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^n \phi_{q,p}(x_p) \right)$
Model (Shallow)	<p>(a)</p>  <p><i>fixed activation functions on nodes</i></p> <p><i>learnable weights on edges</i></p>	<p>(b)</p>  <p><i>learnable activation functions on edges</i></p> <p><i>sum operation on nodes</i></p>
Formula (Deep)	$\text{MLP}(\mathbf{x}) = (\mathbf{W}_3 \circ \sigma_2 \circ \mathbf{W}_2 \circ \sigma_1 \circ \mathbf{W}_1)(\mathbf{x})$	$\text{KAN}(\mathbf{x}) = (\Phi_3 \circ \Phi_2 \circ \Phi_1)(\mathbf{x})$
Model (Deep)	<p>(c)</p>  <p>\mathbf{W}_3</p> <p>σ_2</p> <p>\mathbf{W}_2</p> <p>σ_1</p> <p>\mathbf{W}_1</p> <p>\mathbf{x}</p> <p>$\text{MLP}(\mathbf{x})$</p> <p><i>linear; learnable</i></p> <p><i>nonlinear; fixed</i></p>	<p>(d)</p>  <p>Φ_3</p> <p>Φ_2</p> <p>Φ_1</p> <p>\mathbf{x}</p> <p>$\text{KAN}(\mathbf{x})$</p> <p><i>nonlinear; learnable</i></p>

Implementation details

1. Residual functions

$$\phi(x) = w_b b(x) + w_s spline(x)$$

$$b(x) = PReLU(x) = \max(0, x) + a \min(0, x)$$

$$spline(x) = \sum_i c_i B_i(x)$$

2. Initialization weights

$$w_s = 1, spline(x) \approx 0, c_i \sim \mathcal{N}(0, \sigma^2), \sigma \approx 0.2$$

otherwise \sim Xavier Initialization

3. Grid Extension on the flight

Parameters count

L – depth

N – layer dimensionality

k – spline order

G – spline grid

KAN

$$O(N^2 L (G + 2k)) \sim O(N^2 L G)$$

MLP

$$O(N^2 L)$$

Theorem. Approximation theory

Main representation

$$f = (\Phi_{L-1} \circ \Phi_{L-2} \circ \cdots \circ \Phi_1 \circ \Phi_0)x$$

where $\Phi_{l,i,j}$ are $(k+1)$ -times continuously differentiable.

Then there exist C depending on f and its representation, that bounds approximation error in terms of the grid size G

Theorem. Approximation theory

There exists k -th order B-spline functions $\Phi_{l,i,j}^G$ such that for any $0 \leq m \leq k$, we have the bound

$$\|f - (\Phi_{L-1}^G \circ \Phi_{L-2}^G \circ \cdots \circ \Phi_1^G \circ \Phi_0^G)x\|_{C^m} \leq CG^{-k-1+m}$$

Here we adopt the notation of C^m -norm measuring the magnitude of derivatives up to order m :

$$\|f\|_{C^m} = \max_{|\beta| \leq m} \sup_{x \in [0,1]^m} |D^\beta g(x)|$$

Scaling laws

Neural scaling laws are the phenomenon where test loss decreases with more model parameters, i.e. $\ell \propto N^{-\alpha}$.

A larger α promises more improvement by simply scaling the model.

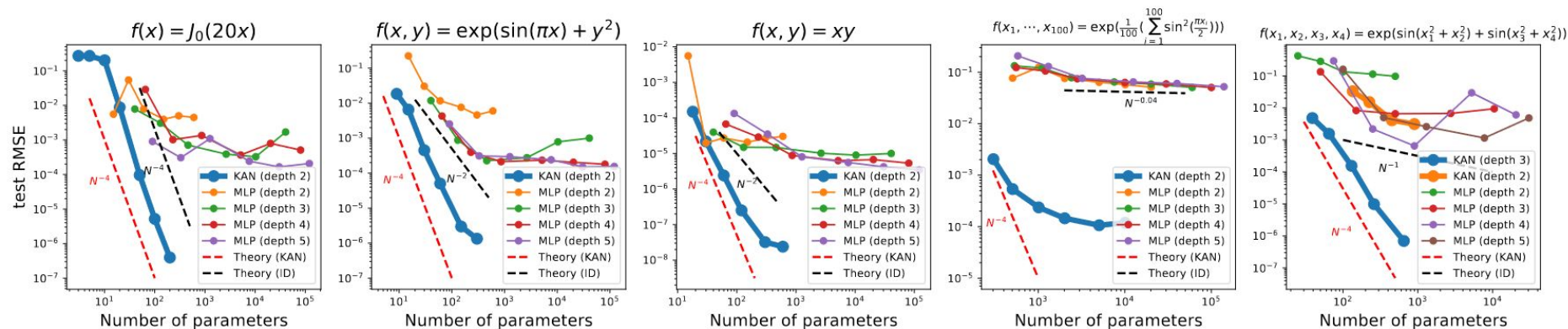
$$\alpha = \frac{(k + 1)}{d}$$

$$\alpha = \frac{k + 1}{d^*}$$

For W_m need $N = O(\epsilon^{-\frac{2}{m}})$ number of parameters to achieve error ϵ

Scaling laws

For $m = 0$ we recover the accuracy in L^∞ norm, which in turn provides a bound of RMSE on the finite domain, which gives a scaling exponent $k + 1$



Grid extension

$$f_{coarse}(x) = \sum_{i=0}^{G_1+2k-1} c_i B_i(x)$$

$$f_{fine}(x) = \sum_{j=0}^{G_2+2k-1} c'_j B'_j(x)$$

$$\{c'_j\} = \arg \min_{\{c'_j\}} \mathbb{E}_{x \sim p(x)} \left(\sum_{j=0}^{G_2+2k-1} c'_j B'_j(x) - \sum_{i=0}^{G_1+2k-1} c_i B_i(x) \right)^2$$

ASVspoof2024

Track 1: stand-alone speech deepfake detection task

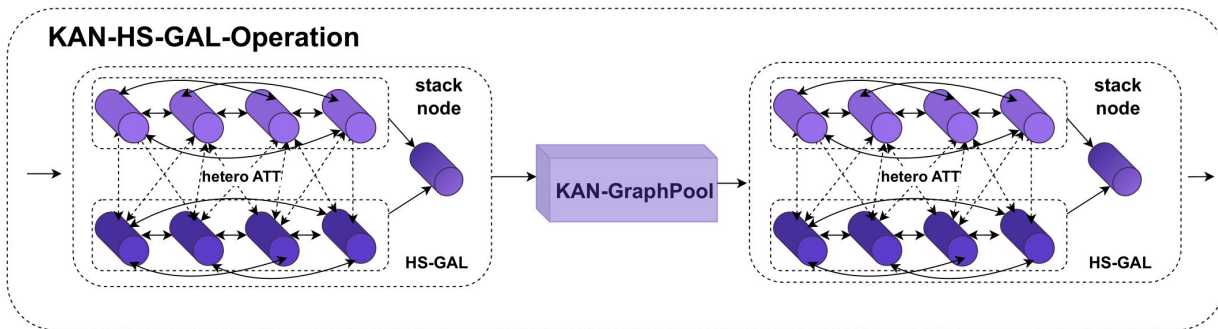
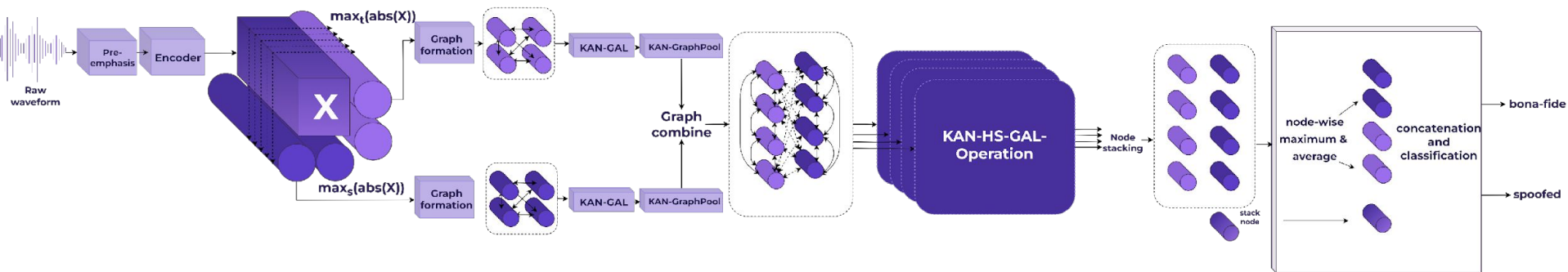
Track 2: spoofing-robust automatic speaker verification task

Closed condition: data within the train partition

Open condition: train partition + external data + pre-trained models (w\out LibriLight, MLS English, MUSAN)

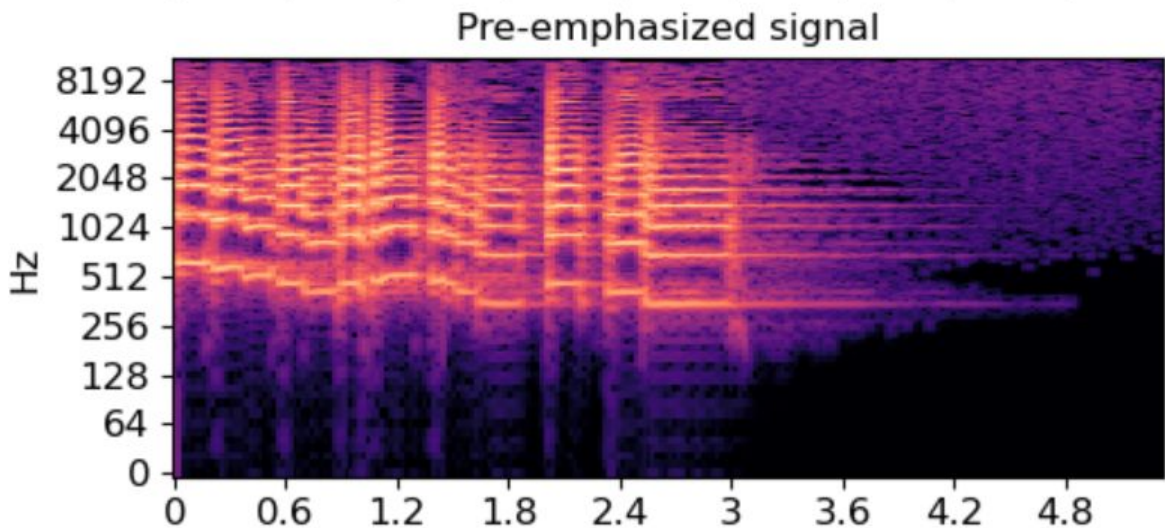
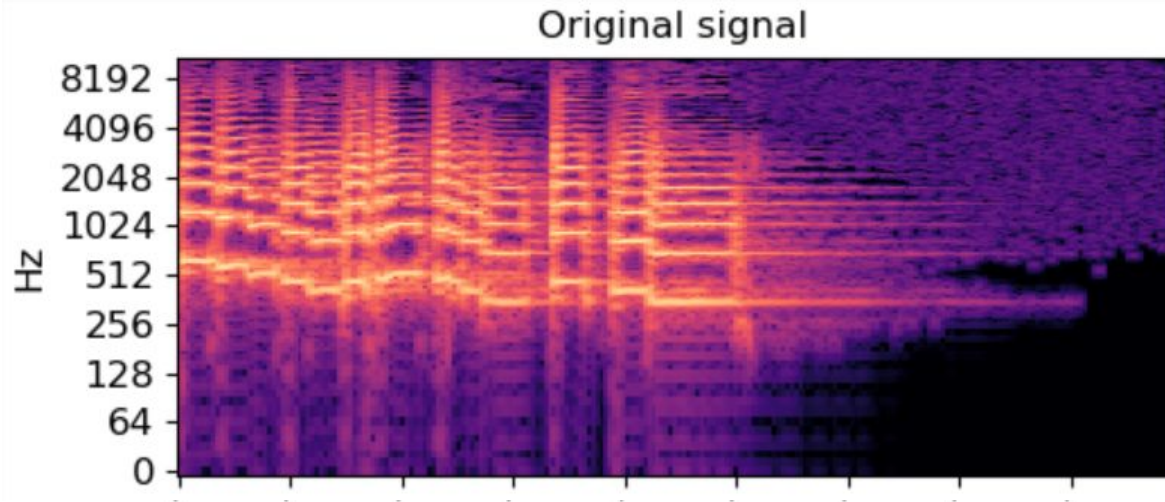
AASIST3

AASIST3 modification	t-DCF
Classic AASIST	0.5671
AASIST3	0.2657



Pre-emphasis

$$x_l = x_l - 0.97x_{l-1}$$



AASIST KANisation

• Attention map in GAL and HS-GAL:

$$A = \textit{softmax} \left(\frac{\tanh(KAN_1(h \times h))W_{att}}{T} \right)$$

Final aggregation in GAL and HS-GAL:

$$[HS-]GAL(h) = \textit{BatchNorm}(KAN_1(Ah) + KAN_2(h))$$

AASIST KANisation

- Graph pooling

$$GraphPool(h, k) = rank \left((\sigma(KAN(h)) \odot h), k \right)$$

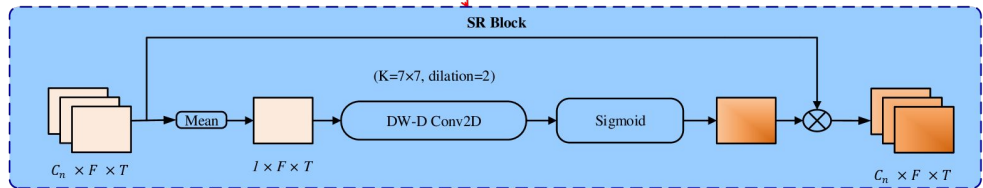
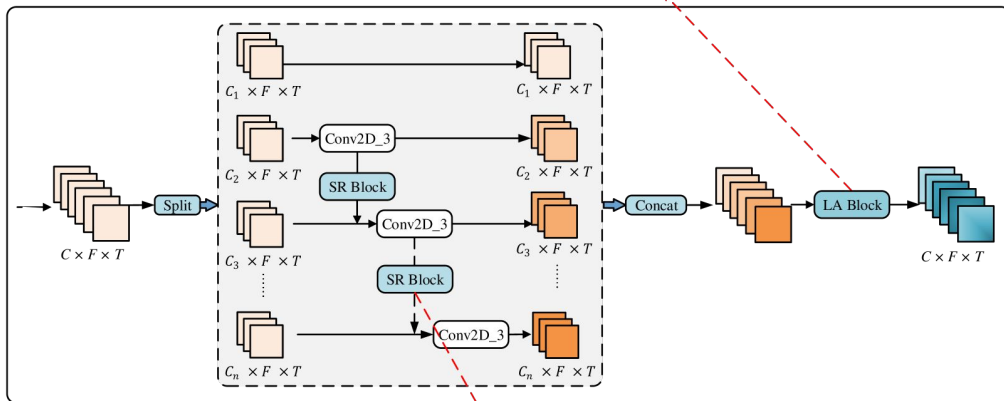
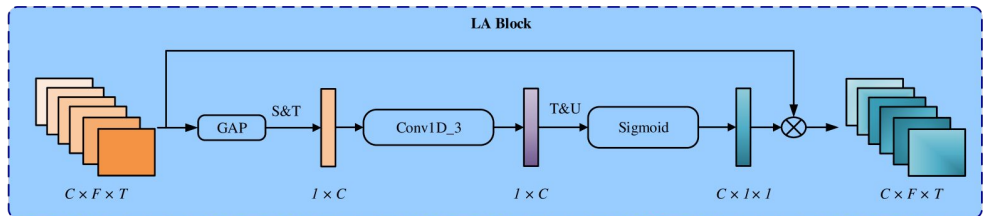
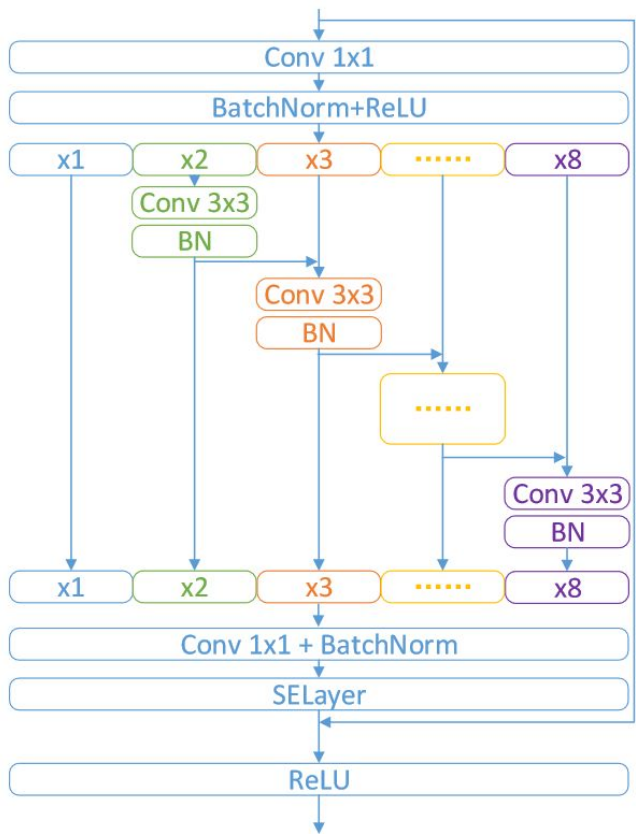
Primary attention map in HS-GAL

$$A = \tanh(KAN(h \times h))$$

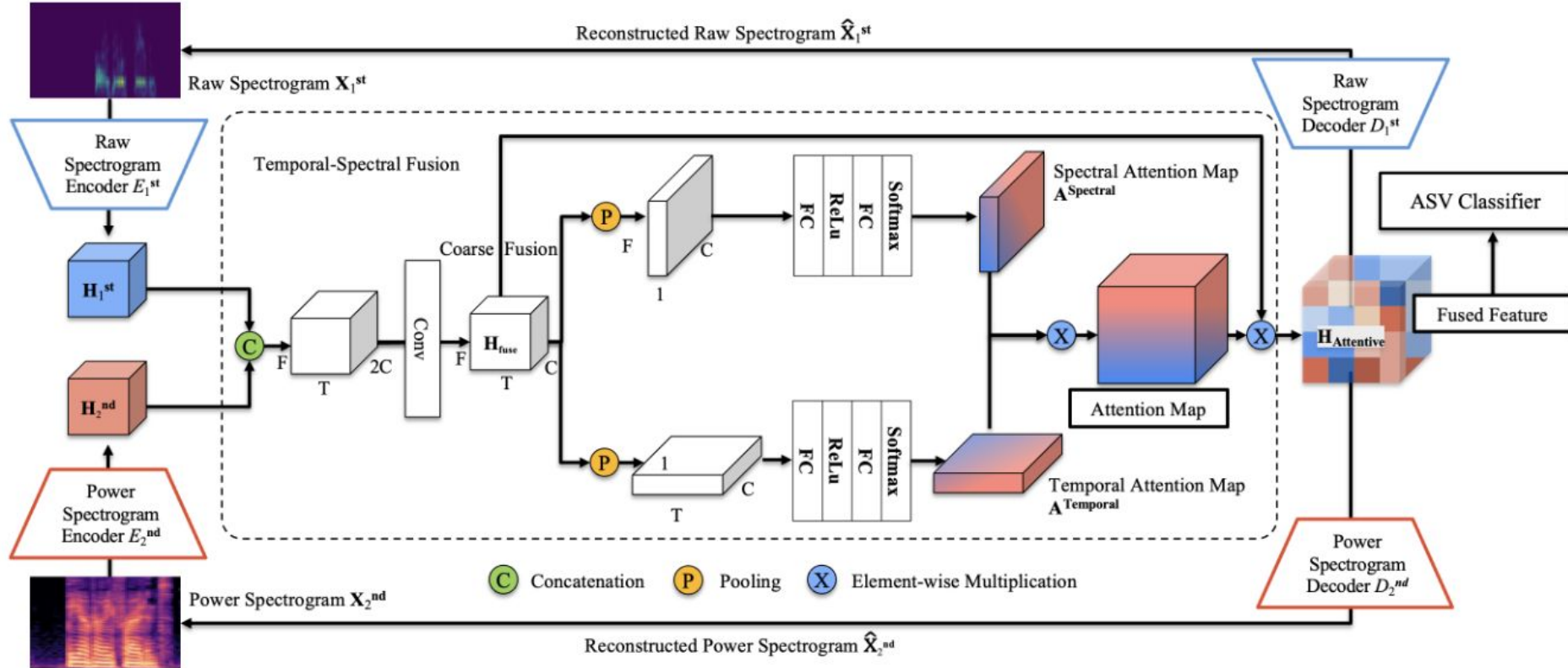
Stack node reweighting

$$\hat{S} = KAN_1(Ah) + KAN(S)$$

AASIST2 – AASIST with Res2Net



S^2 pecNet – AASIST with another encoder



Results

S ² pecNet with 40 batch size	0.4291
S ² pecNet with 28 batch size	0.4225
S ² pecNet with 20 batch size	0.4185
RawNet3 instead of RawNet2	0.4901
Res2Net encoder with lr=1e-6	0.9066
SR LA Res2Net encoder	0.7203
Res2Net encoder with lr=1e-5	0.6413
SR LA Res2Net + f0 subband	0.5463
Res2Net encoder + PRELU with lr=1e-4	0.485
LayerNorm instead of BatchNorm	0.3591
ResNet + effective local attention	0.3542
ResNet with PReLU	0.3216
ResNet + SE	0.2902

Generalized Cross Entropy Loss	0.8438
ArcFace Loss	0.4389
Multitask Loss	0.3933
Focal Loss	0.3489
AM Softmax	0.3363

Lion instead of Adam	0.3702
AdamW instead of Adam	0.3200
NAdam instead of Adam	0.2889
RAdam instead of Adam	0.3006

Results

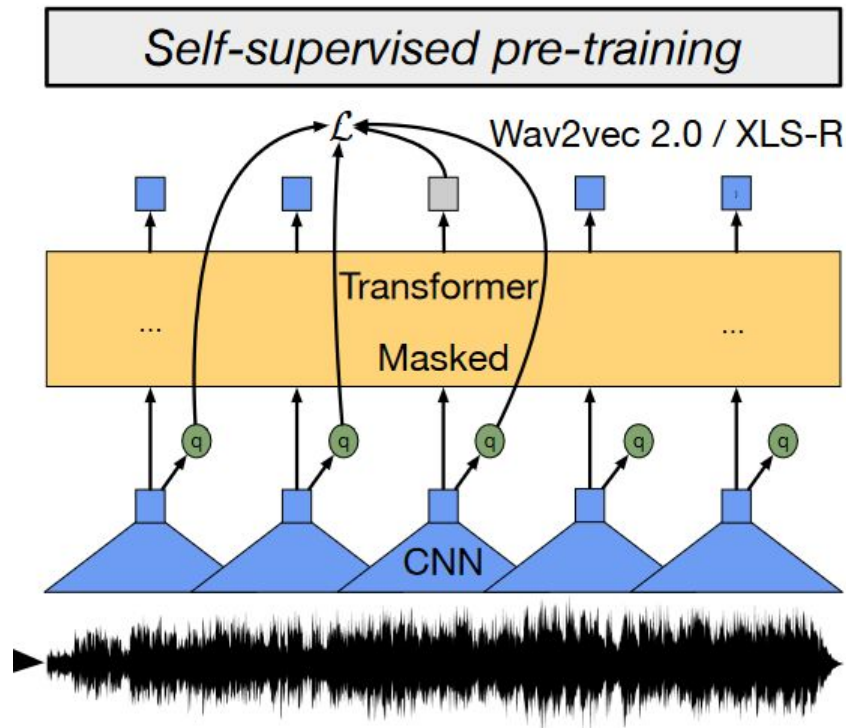
AReLU instead of PReLU in KAN	0.3371
SELU instead of PReLU in KAN	0.3295
RReLU instead of PReLU in KAN	0.3045

Bessel polynomials in KAN	0.6805
Jacobi Polynomials in KAN	0.5666
Legendre Polynomials in KAN	0.4665
Gegenbauer polynomials in KAN	0.4051
RBF in KAN	0.3994
2nd Chebyshev polynomials	0.3392
Fibonacci polynomials in KAN	0.492

ReLUConvKAN instead of Conv	0.699
UKAN in encoder	0.3062
WavKANConv in encoder	0.3047
WavKANConv in encoder + SAM	0.2801

Frontends for open condition

- Wav2Vec2.0 XLS-R 300/1B/2B
- HuBERT
- WavLM
- XEUS



Frontends for open condition

version	#TTL	#FTL	min t-DCF	EER(%)
XLSR-53	12	3	0.0083	0.26
XLS-R(0.3B)	15	6	0.0093	0.29
XLS-R(1B)	12	9	0.0063	0.22
XLS-R(2B)	18	3	0.0098	0.30

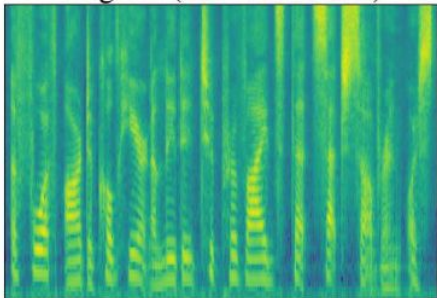
#TTL	#FTL								
	0	3	6	9	12	15	18	33	48
3	8.96	38.23	-	-	-	-	-	-	-
6	7.71	6.54	42.82	-	-	-	-	-	-
9	1.38	0.88	0.98	17.84	-	-	-	-	-
12	0.77	0.41	0.57	0.22	17.85	-	-	-	-
15	0.87	1.04	1.35	0.80	1.43	41.00	-	-	-
18	0.63	0.40	0.61	2.14	2.39	6.68	26.01	-	-
33	2.41	2.01	3.94	3.57	4.26	4.36	4.54	37.51	-
48	2.37	5.75	0.65	2.83	7.52	5.13	9.72	4.53	41.09

Table 1. EER(%) results on XLS-R(1B) and AASIST combination system.

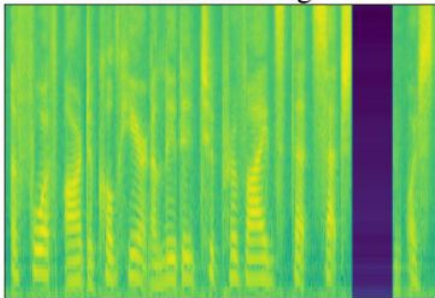
Table 2. Results on various wav2vec 2.0 front-ends.

Augmentations

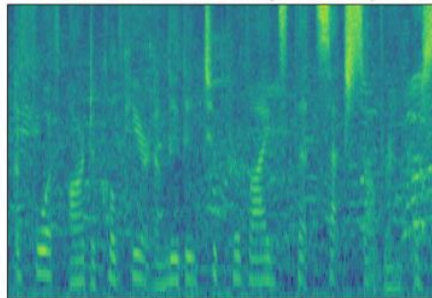
Original (T_0000077083)



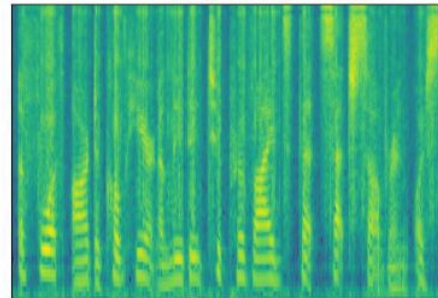
Time masking



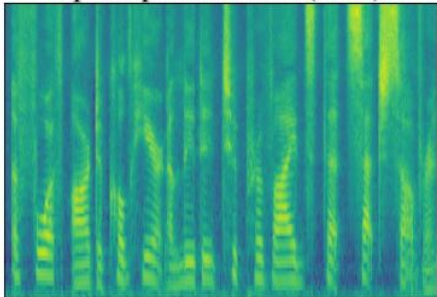
MUSAN noise (SNR=5)



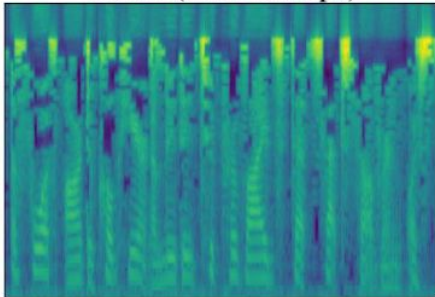
RawBoost



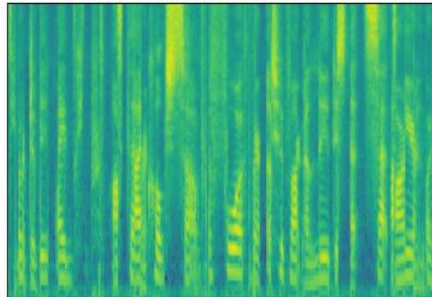
Speed perturbation (0.8x)



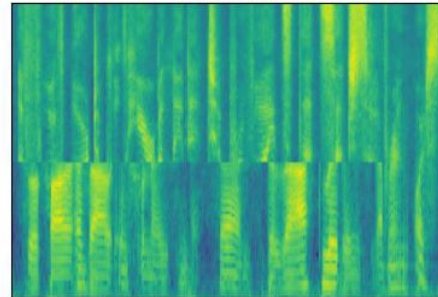
Codec (MP3 16kbps)



Audio shuffle

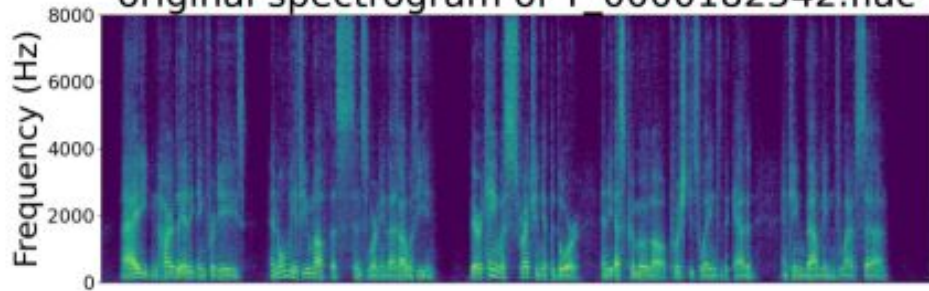


CutMix



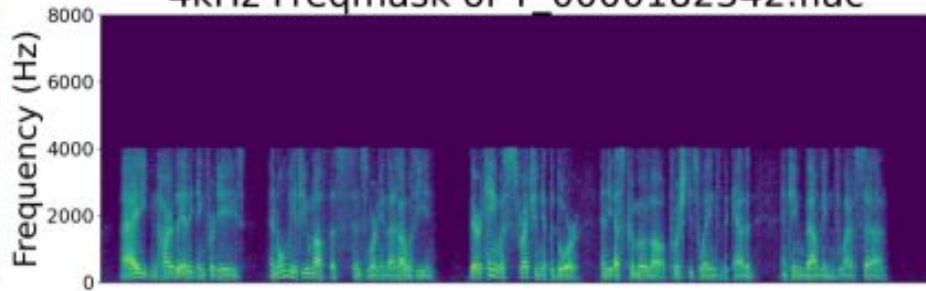
Augmentations

original spectrogram of T_0000182342.flac



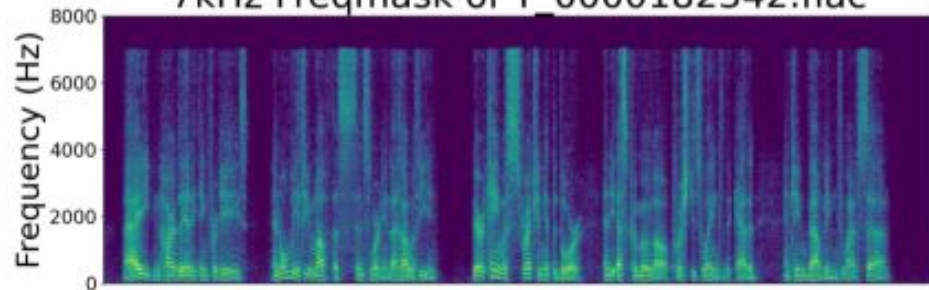
(a)

4kHz Freqmask of T_0000182342.flac



(b)

7kHz Freqmask of T_0000182342.flac



(c)

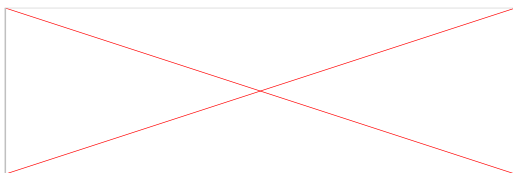
4kHz lowpass filter of T_0000182342.flac



(d)

Noise datasets and codecs

MUSAN: A Music, Speech, and Noise Corpus



	Codec	Bandwidth	Bitrate range
C00	-	16 kHz	-
C01	opus	16 kHz	6.0 - 30.0
C02	amr	16 kHz	6.6 - 23.05
C03	speex	16 kHz	5.75 - 34.20
C04	Encodect [32]	16 kHz	1.5 - 24.0
C05	mp3	16 kHz	45 - 256
C06	m4a	16 kHz	16 - 128
C07	mp3+Encodect	16 kHz	varied
C08	opus	8 kHz	4.0 - 20.0
C09	arm	8 kHz	4.75 - 12.20
C10	speex	8 kHz	3.95 - 24.60
C11	varied	8 kHz	varied

ResNet34 is all you need??

Model	Arch.	Data augmentation			Training params.		Progress set		Development set	
		TimeMask	Noise [†]	Speed	n_mels	n_steps	minDCF	EER (%)	minDCF	EER (%)
X1	R34		N		120	200k	0.0741	2.67	0.0907	3.88
X2	R34		N, M	✓	120	480k	0.0678	2.34	0.1140	4.57
X3	R34	✓	N		120	200k	0.0754	2.72	0.1057	4.26
X4	R34	✓	N, M	✓	120	320k	0.0743	2.60	<u>0.0859</u>	<u>3.26</u>
X5	R34	✓	N, M, S		128	200k	0.0623	2.28	0.1008	4.20
X6	R34	✓	N, M, S		128	280k	0.0728	2.70	0.1232	5.13
X7	R34	✓	N, M, S	✓	120	400k	0.0757	2.62	0.0939	3.66
X8	R34	✓	N, M, S	✓	128	220k	0.0569	2.20	0.1068	4.49
X9	R34	✓	N, M, S	✓	160	200k	<u>0.0614</u>	<u>2.21</u>	0.0971	4.16
X10*	R34	✓	N, M, S	✓	160	200k	0.0739	2.62	0.0795	3.23

Thank you for your

