Machine Learning in Speech Processing

[Brief Talk | STTP Deep Learning, Computer Vision, Speech Analysis]
[Xavier Institute of Engineering | Mumbai]

Dr. Sunil Kumar Kopparapu

SunilKumar.Kopparapu@TCS.COM

TCS Innovation Labs - Mumbai

Tata Consultancy Services Ltd, Yantra Park,
Thane (West), Maharastra 400601,
INDIA.

January 11, 2018

Acknowledgements

The Team (The Work) $\longrightarrow$ Team Work!

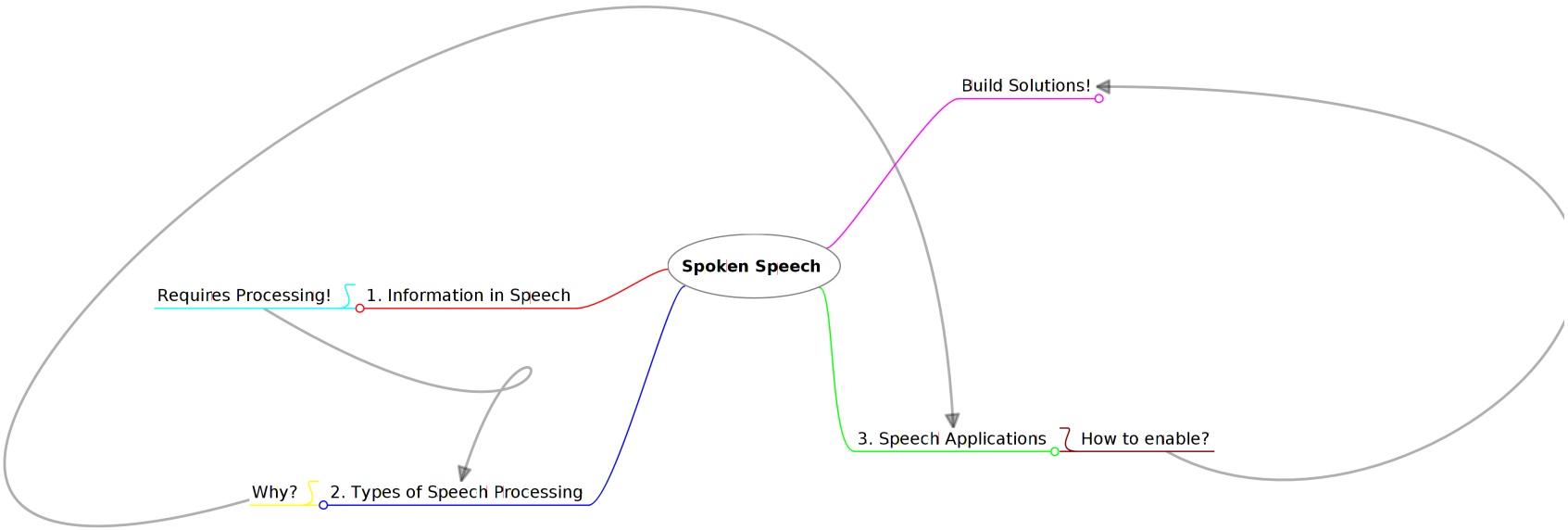
- Ashish (Robust Speaker and Speech Recognition)
- Chitralekha (Speech Analysis for Assistive Technology)
- Imran (Voice Bots)
- Rupayan (Audio Emotion in Sponteanous Speech)
- Anantaram (Reusing gpASR for enterprise Applications)

Acknowledgements

The Team (The Work) $\longrightarrow$ Team Work!

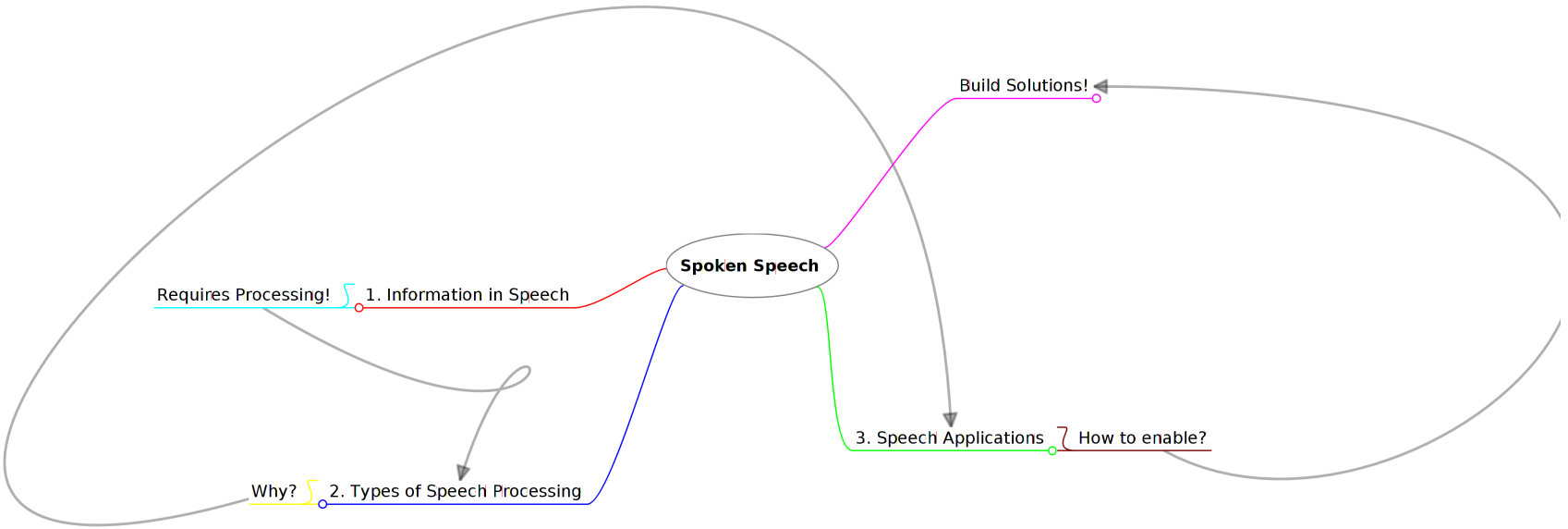
- Ashish (Robust Speaker and Speech Recognition)
- Chitralekha (Speech Analysis for Assistive Technology)
- Imran (Voice Bots)
- Rupayan (Audio Emotion in Sponteanous Speech)
- Anantaram (Reusing gpASR for enterprise Applications)

Some Background



Why Analyze Speech? ▶ Details

Some Background



Why Analyze Speech? ▸ Details

Machine Learning in Speech Processing

This is not an overview!
Will **only** cover **some** of the current (2016-2017) work **actually** done in the lab

Assumptions and Questions

Assumptions

- Feel for "where all one can use Machine Learning" (known)
- Actual ML algorithms | ANN | DNN architectures have been discussed in earlier talks | literature (will not detail this!)

Assumptions and Questions

Questions

- When do you generally use ML | DNN?
  (a) When there is access to a lot of (annotated) data.
  (b) Data is noise free.
  (c) Train-Test match conditions
  (d) Data is balanced
  (e) Black box learning (no specific information about data is exploited)

Assumptions and Questions

Questions

- When do you generally use ML | DNN?
  - (a) When there is access to a lot of (annotated) data.
  - (b) Data is noise free.
  - (c) Train-Test match conditions
  - (d) Data is balanced
  - (e) Black box learning (no specific information about data is exploited)
- What if ...

(a) $\rightarrow$ What does one do when there is not enough data?
(b) $\rightarrow$ Data is noisy?
(c) $\rightarrow$ What do you do when there is a train-test mismatch?
(d) $\rightarrow$ What do you do when the data classes are not balanced?
(e) $\rightarrow$ How does one exploit the prior knowledge about the data | problem?

Assumptions and Questions

Questions

- When do you generally use ML | DNN?
  - (a) When there is access to a lot of (annotated) data.
  - (b) Data is noise free.
  - (c) Train-Test match conditions
  - (d) Data is balanced
  - (e) Black box learning (no specific information about data is exploited)
- What if ...

(a) $\rightarrow$ What does one do when there is not enough data?
(b) $\rightarrow$ Data is noisy?
(c) $\rightarrow$ What do you do when there is a train-test mismatch?
(d) $\rightarrow$ What do you do when the data classes are not balanced?
(e) $\rightarrow$ How does one exploit the prior knowledge about the data | problem?

ML in **What if ....** scenario ... most real world problems are!

We will look at some of these aspects form speech analysis perspective.

Machine Learning in **What if ....** scenario

(a) What does one do when there is not enough data?
(b) Data is noisy?
(c) What do you do when there is a train-test mismatch?
(d) What do you do when the data classes are not balanced?
(e) How does one exploit the prior knowledge about the data | problem?

Examples

1. Learning to use a gpASR for domain specific ASR $\longrightarrow$ [(a), (c)]
2. Robust Front End for ASR $\longrightarrow$ [(b), (c)]
3. Knowledge driven FFNN for Emotion Analysis $\longrightarrow$ [(a), (e)]
4. Simultaneous 2 Class Learning $\longrightarrow$ [(a), (d)]
5. ASR for Dysarthric Speech $\longrightarrow$ [(a), (c), (e)]
6. ECC to Repair ANN output $\longrightarrow$ [(a), (b), (c))]

Conclude

In Conclusion

- Why is Deep Learning getting popular?
  - Abundance of speech data (every time you use google, you are giving out well annotated in terms of language, region, age, gender, $\cdots$ speech data for free)
  - Faster machines or use of GPU's (means quicker experimental analysis, more experiments, $\cdots$)

Two sides of Speech! The feature debate

- Listener Perspective: Speech eventually is heard by someone so we need to model speech from the perception perspective so we should use MFCC.
- Speaker Perspective: Speech is produced by a human, so we need to model speech as a speech generation perspective, so we should use LPC like features.

  - Lesser decisions (which features to use, feature selection to determine how many features to use, $\cdots$)

In Conclusion

- Why is Deep Learning getting popular?
  - Abundance of speech data (every time you use google, you are giving out well annotated in terms of language, region, age, gender, $\cdots$ speech data for free)
  - Faster machines or use of GPU's (means quicker experimental analysis, more experiments, $\cdots$)

Two sides of Speech! The feature debate

- Listener Perspective: Speech eventually is heard by someone so we need to model speech from the perception perspective so we should use MFCC.
- Speaker Perspective: Speech is produced by a human, so we need to model speech as a speech generation perspective, so we should use LPC like features.

  - Lesser decisions (which features to use, feature selection to determine how many features to use, $\cdots$)

A combination of all this is, probably making **"use of DNN"** popular in Speech Analysis!

Thank You

- Queries? | Comments | Suggestions?

Dr Sunil Kopparapu

SunilKumar.Kopparapu@TCS.Com
TCS Innovation Lab - Mumbai
Tata Consultancy Services Limited
Loc: 72.977265, 19.225129 Yantra Park, Thane (West), India.

END

Learning to use a gpASR for domain specific ASR

Based On

- Sunil Kumar Kopparapu, C. Anantaram, "Reusing General Purpose ASR for Domain Specific Speech Recognition Using Posteriors" SLSP 2017, Le Mans, France.
- Sunil Kumar Kopparapu and C. Anantaram, "Correcting General Purpose ASR Errors using Posterior" ICON 2017 : 14th International Conference on Natural Language Processing Jadavpur University, Kolkata, India Dec, 2017

Learning to use a gpASR for domain specific ASR

What is it about? In Brief

Bridging the gap between what is available and what is often required!

- Available
  - Several Speech Recognition Engines
  - Large Available Corpus + Deep Learning Architecture
  - Useful for general purpose
- Need
  - Speech Recognition applications in Enterprise environment
  - Domain specific
  - Specific environments

Question

Can we use the readily available gpASR for use in domain specific Speech Recognition?

Why do we do this?

Why not?

use when something is already available

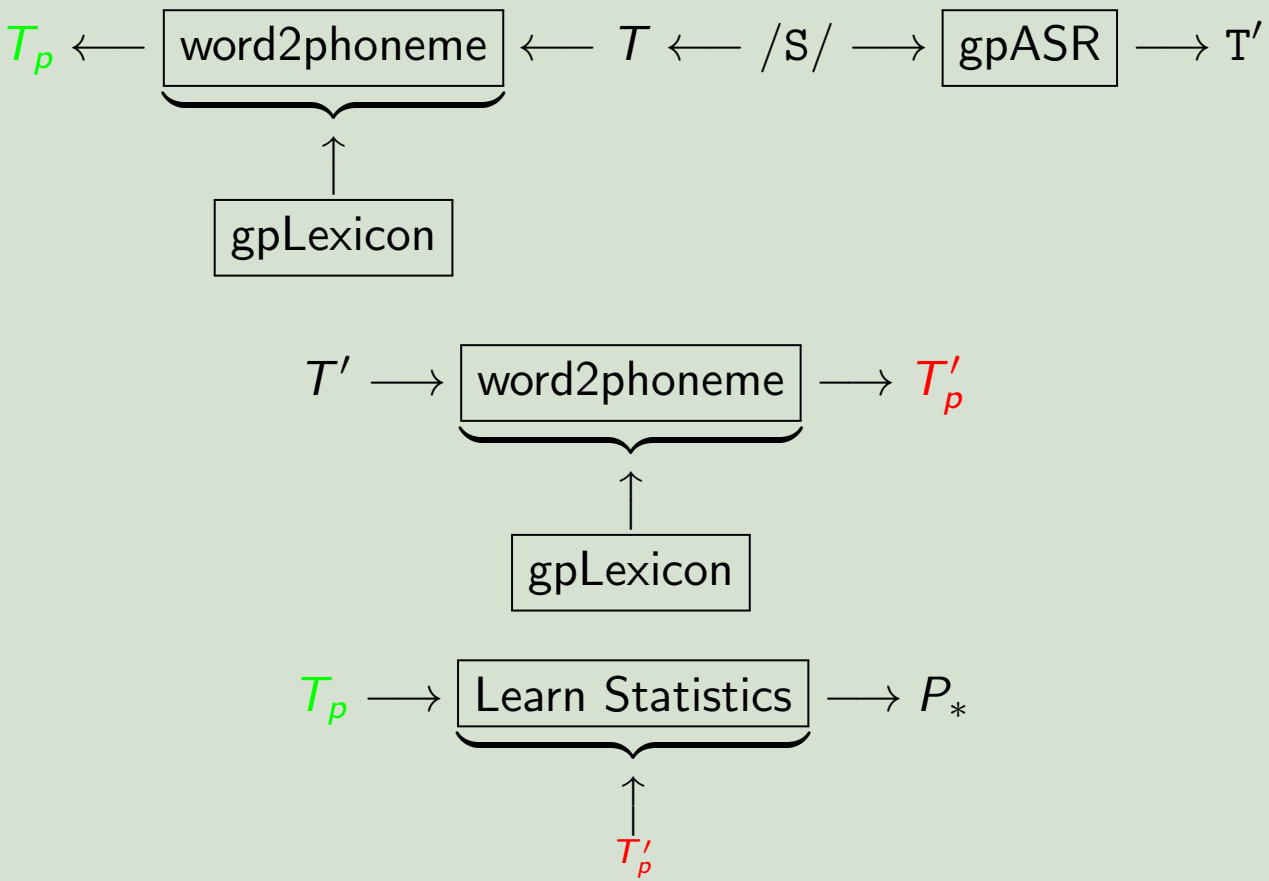
- Building a speech recognition engine is an elaborate process
- Do not want to re-build or re-train a ASR every time
  - a new domain is considered or
  - usage scenario changes (environment, new vocabulary etc)

Why not reuse | re-purpose a gpASR?

How do we do this?

Learn Posteriors

- We learn the acoustic characteristics of the gpASR in our specific usage scenario (domain, environment)
- In terms of phone (not word) substitution, replacement, insertions



How do we do this?

Learn from (K) Data

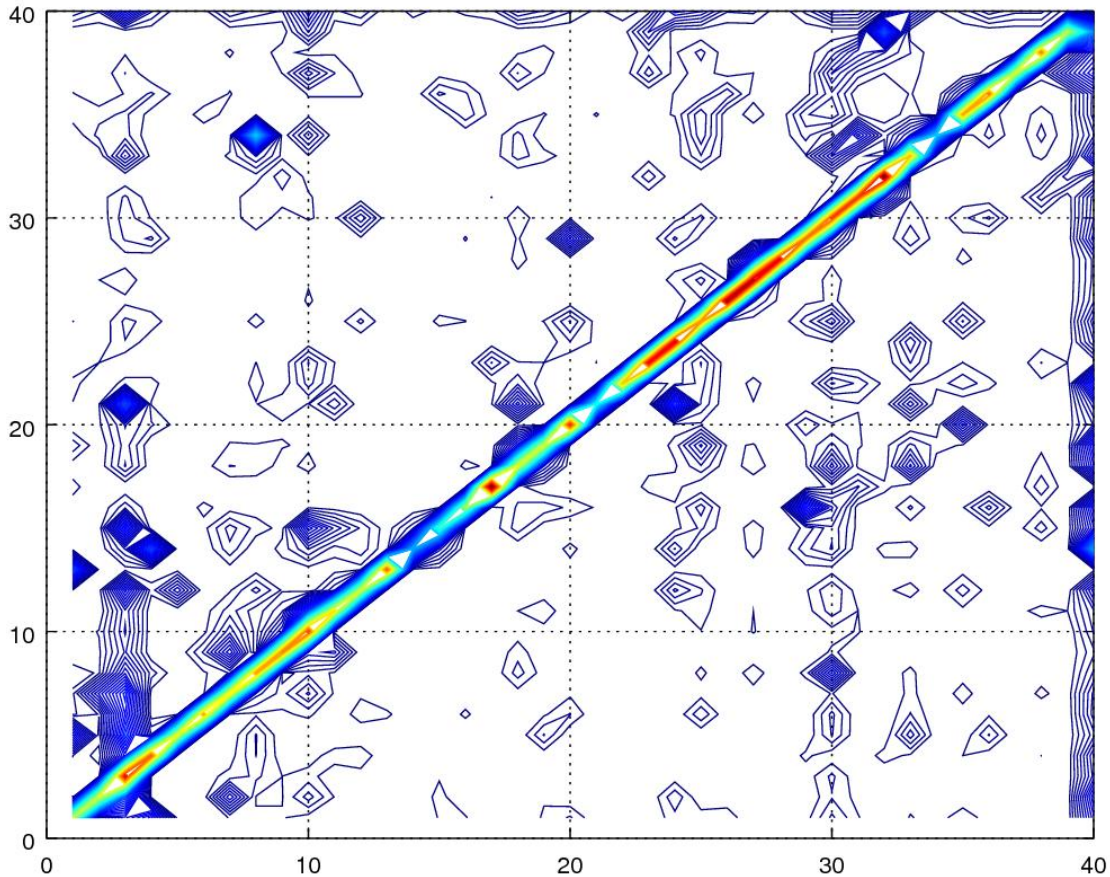
- $P_{sub} = P(w_i \overset{sub}{\rightarrow} w_j) = \frac{\#((w_i \in \vec{T}'_p)\&(w_j \in \vec{T}_p))}{\#(w_i \in \vec{T}'_p)}$ where
  - (a) $\#(w_i \in \vec{T}'_p)$ is the count of the phone w_i occurring in $\{\vec{T}'^{\,i}_p\}_{i=1}^{K}$ and
  - (b) $\#((w_i \in \vec{T}'_p)\&(w_j \in \vec{T}_p))$ is the count of the phone w_i which occurs in $\{\vec{T}'^{\,i}_p\}_{i=1}^{K}$ when w_j occurs in $\{\vec{T}^{\,i}_p\}_{i=1}^{K}$.
- $P_{ins} = P(\phi \overset{sub}{\rightarrow} w_j)$ where w_j occurs in $\vec{T}_p$ but does not occur in $\vec{T}'_p$
- $P_{del} = P(w_i \overset{sub}{\rightarrow} \phi)$ where w_i occurs in $\vec{T}'_p$ but does not occur in $\vec{T}_p$.

Set of Phones ($39 + 1$)

- $w_* \in \boldsymbol{P}$ and
- $\boldsymbol{P} = \{$ 'a', 'ae', 'ah', 'ao', 'aw', 'ay', 'b', 'ch', 'd', 'dh', 'eh', 'er', 'ey', 'f', 'g', 'hh', 'ih', 'iy', 'jh', 'k', 'l', 'm', 'n', 'ng', 'ow', 'oy', 'p', 'r', 's', 'sh', 't', 'th', 'uh', 'uw', 'v', 'w', 'y', 'z', 'zh' $\}$ $\boxed{+\,\phi}$.

How do we do this?

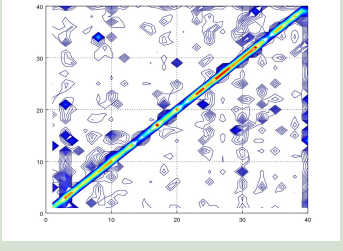
Learnt Statistics (P_*)



How do we do this?

Use

- Using the domain Lexicon we repair the gpASR out (thereby reusing the gpASR output for our domain)

$$T'_p \longrightarrow \boxed{\text{  }} \longrightarrow \mathcal{L}$$

and then search the lattice $\mathcal{L}$ of phones to identify the word W' using a domain lexicon (dLexicon).

$$\mathcal{L} \longrightarrow \underbrace{\boxed{\text{search, phoneme2word}}}_{\uparrow} \longrightarrow W'$$

$$\boxed{\text{dLexicon}}$$

Example

- S = /`who is the accountable person for manufacturing solutions/`
- $\vec{T}'_p$ = who is accountable boston for the men affecting solutions
- Correction Required:
  who is ($\phi \xrightarrow{ins}$the) accountable (boston$\xrightarrow{sub}$person)
  for (the $\xrightarrow{del}$ ϕ) (men $\xrightarrow{del}$ ϕ) (affecting$\xrightarrow{sub}$manufacturing) solutions.

Details Example 1: boston $\longrightarrow$ person

- Algorithm and Details
- Briefly
  - Expand 'b ao s t ah n' $\longrightarrow$ ' ϕ b ϕ ao ϕ s ϕ t ϕ ah ϕ n ϕ '
  - Construct lattice $\mathcal{L}$ using the posterior P_*.
  - Search $\mathcal{L}$

boston												
		b		ao		s		t		ah		n	
1	$\underline{\phi}$	b	$\underline{\phi}$	ao	$\underline{\phi}$	$\underline{\text{s}}$	$\underline{\phi}$	t	$\underline{\phi}$	$\underline{\text{ah}}$	$\underline{\phi}$	**n**	ϕ
2	ah	$\underline{\text{p}}$	ah	ah	ah	sh	ah	$\underline{\phi}$	ah	ϕ	ah	ϕ	ah
3	ih	ϕ	ih	ϕ	ih	ϕ	ih	r	ih	er	ih	l	ih
4	k	r	k	$\underline{\text{er}}$	k	z	k	ah	k	r	k	ah	k
5	t	w	t	aa	t	ah	t	er	t	aa	t	k	t
		$\underline{\text{p}}$		$\underline{\text{er}}$		$\underline{\text{s}}$				$\underline{\text{ah}}$		$\underline{\text{n}}$	
person													

Details Example 2

Handles Errors Across Words!

men							affecting								
	m		eh		n		ah		f		eh		k		t
$\underline{\phi}$	$\underline{\text{m}}$	$\underline{\phi}$	eh	$\underline{\phi}$	$\underline{\textbf{n}}$	$\underline{\phi}$	$\underline{\textbf{ah}}$	$\underline{\phi}$	$\underline{\text{f}}$	$\underline{\phi}$	eh	ϕ	$\underline{\textbf{k}}$	$\underline{\phi}$	t
ah	ϕ	ah	ah	ah	ϕ	ah	ϕ	ah	v	ah	ah	ah	ϕ	ah	ϕ
ih	n	ih	$\underline{\textbf{ae}}$	ih	l	ih	er	ih	ah	ih	$\underline{\textbf{ae}}$	ih	t	ih	r
k	l	k	ϕ	k	ah	k	r	k	ϕ	k	ϕ	k	ah	k	al
t	ih	t	ih	t	k	t	aa	t	p	t	ih	t	f	t	$\underline{\text{e}}$
	m		ae		n		ah		f		ae		k		e
manufacturing															

Summary

Discussion

- gpASR's are continuously being tuned
  - Several applications make use of them
  - One can not predict its performance
- Re-purposing gpASR for domain specific ASR
- Useful for resource deficit languages (no ASR)
- Models gpASR (in some sense in P_*)
- Posterior carry train-test data mismatch information
- Approach (simple, yet) able to rectify error spread across words

References

- Sunil Kumar Kopparapu, C. Anantaram, "Reusing General Purpose ASR for Domain Specific Speech Recognition Using Posteriors" SLSP 2017, Le Mans, France.
- Sunil Kumar Kopparapu and C. Anantaram, "Correcting General Purpose ASR Errors using Posterior" ICON 2017 : 14th International Conference on Natural Language Processing Jadavpur University, Kolkata, India Dec, 2017

back to Main

Robust Front End for Speech Recognition

Based On

- B. Das and A. Panda, A Robust front-end processing for speech recognition in noisy conditions,in Proc. ICASSP, pp. 5235–5239, March 2017.
- Biswajit Das, Ashish Panda, "Vector Taylor Series Expansion with Auditory Masking for Noise Robust Speech Recognition", IEEE SigPort, 2016.
- Computer implemented system and method for identifying significant speech frames within speech signals, Panda, A. and Kopparapu, S.K., 2017, US Patent 9,659,578

Robust Front End for Speech Recognition

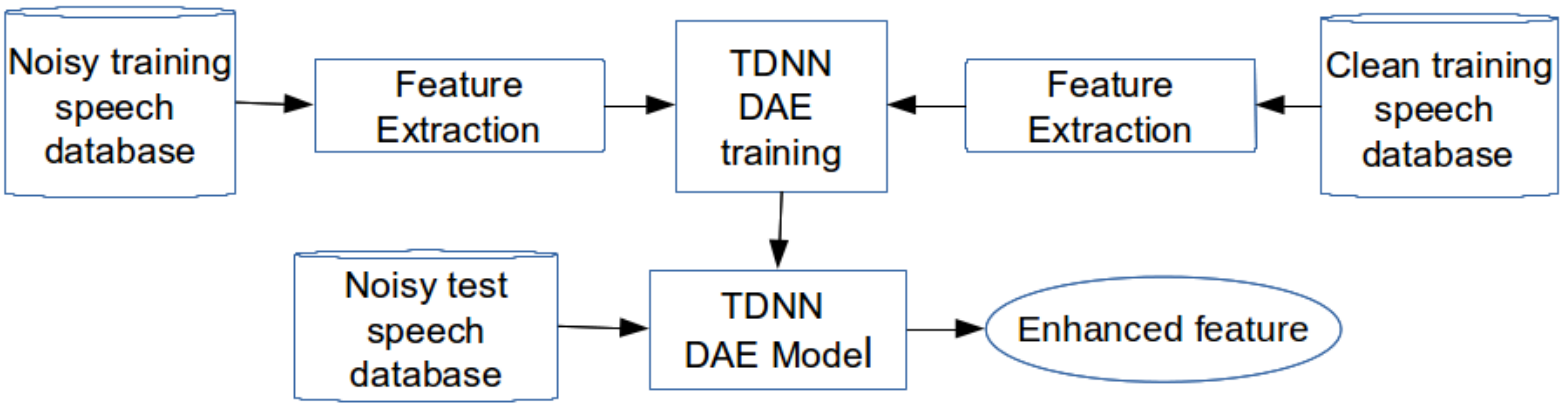
- Automatic speech recognition performance degrades as the system encounters noisy speech.
- One way to deal with the noisy speech is to estimate a robust feature set from a given noisy feature set.

How?

Use Deep Learning, namely, Denoising Auto Encoder (DAE).

Denoising Auto Encoder (DAE)

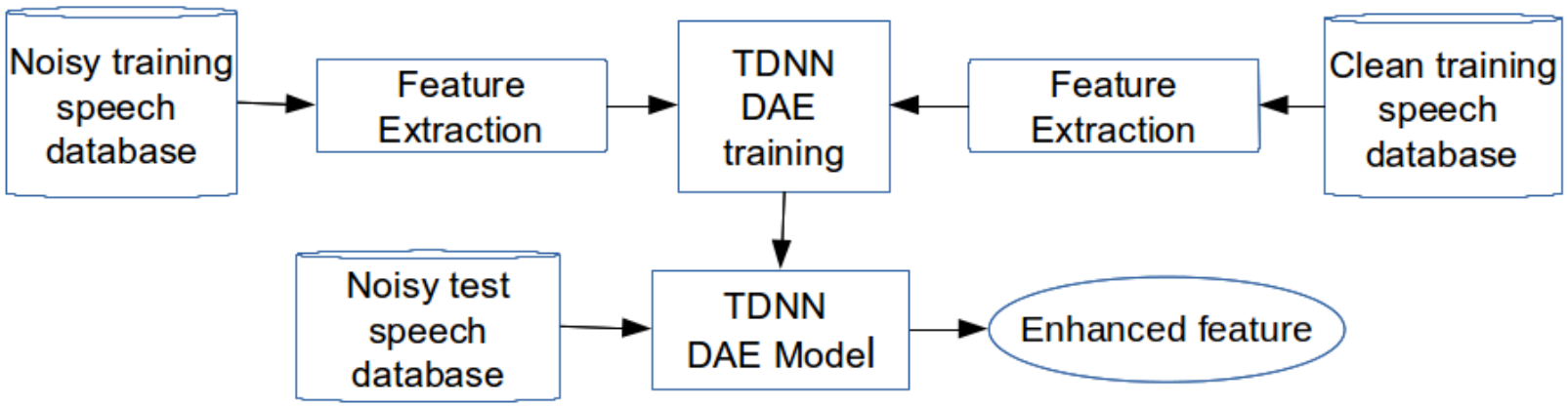
(Deep ML) to reconstruct a clean input from a corrupted version



- Time Delay Neural Network (TDNN) architecture is employed as DAE
- Training: Noisy speech and corresponding clean speech form the input-output pair
- Robust speech recognition. Noisy speech is input to the (trained) DAE. The output of the DAE is input to the speech recognition engine.

Denoising Auto Encoder (DAE)

(Deep ML) to reconstruct a clean input from a corrupted version



Experimental Observations

- Noise encountered by the ASR was seen during DAE training Performance of ASR system improved by absolute 20%. (Librispeech database)
- If not. ASR performance degrades by 7%

Possible Solution

- The DAE performs well if noise encountered in the test speech is similar to the noise used in the training phase ("Seen" noise).
- The performance of the DAE degrades if the noise in the test speech is not used in the training phase ("Unseen" noise; mismatch condition).

Possible Solution

Overcoming mismatch scenario?

combining Vector Taylor Series expansion with Acoustic Masking (VTS-AM) and Denoising Auto Encoder (DAE).

VTS

- Vector Taylor Series (VTS) is used for noisy speech recognition
- Hidden Markov Models (HMMs) are trained (λ) using clean speech (X),
- Clean speech HMM parameters ($\bar{\lambda} \leftarrow \lambda$) adapted to test noisy speech ($\bar{X}$).
- How? by estimating the original clean speech ($\hat{X}$)

$$p(\hat{\vec{x}}|\lambda) = p(\bar{\vec{x}}|\bar{\lambda}), \text{ where, } \hat{\vec{x}} \in \hat{X} \text{ and } \bar{\vec{x}} \in \bar{X}$$

  from the test noisy speech
- Assumes $\vec{y}_e = \vec{x}_e + \vec{n}_e$ (additive)

Possible Solution

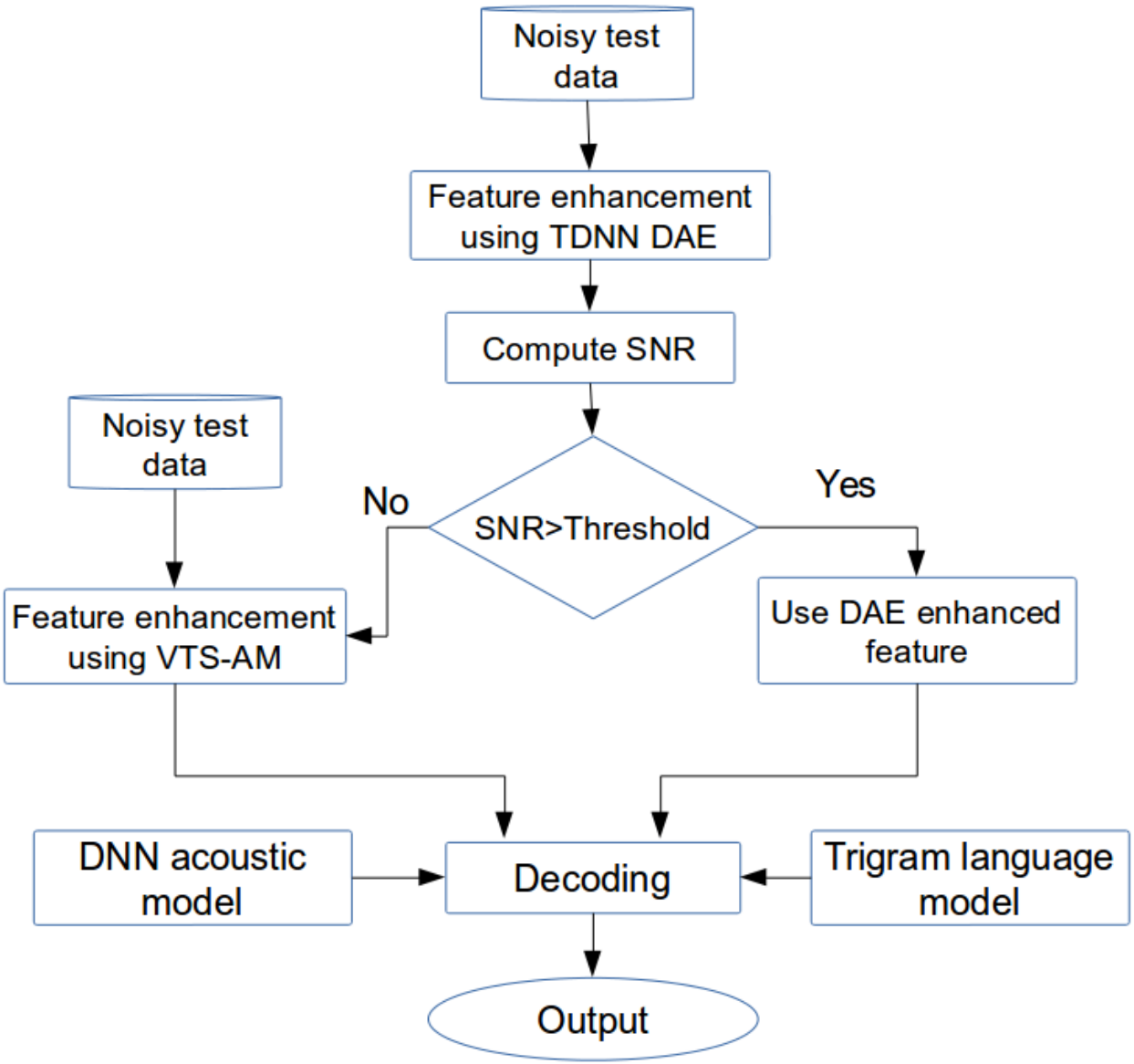
Overcoming mismatch scenario?

combining Vector Taylor Series expansion with Acoustic Masking (VTS-AM) and Denoising Auto Encoder (DAE).

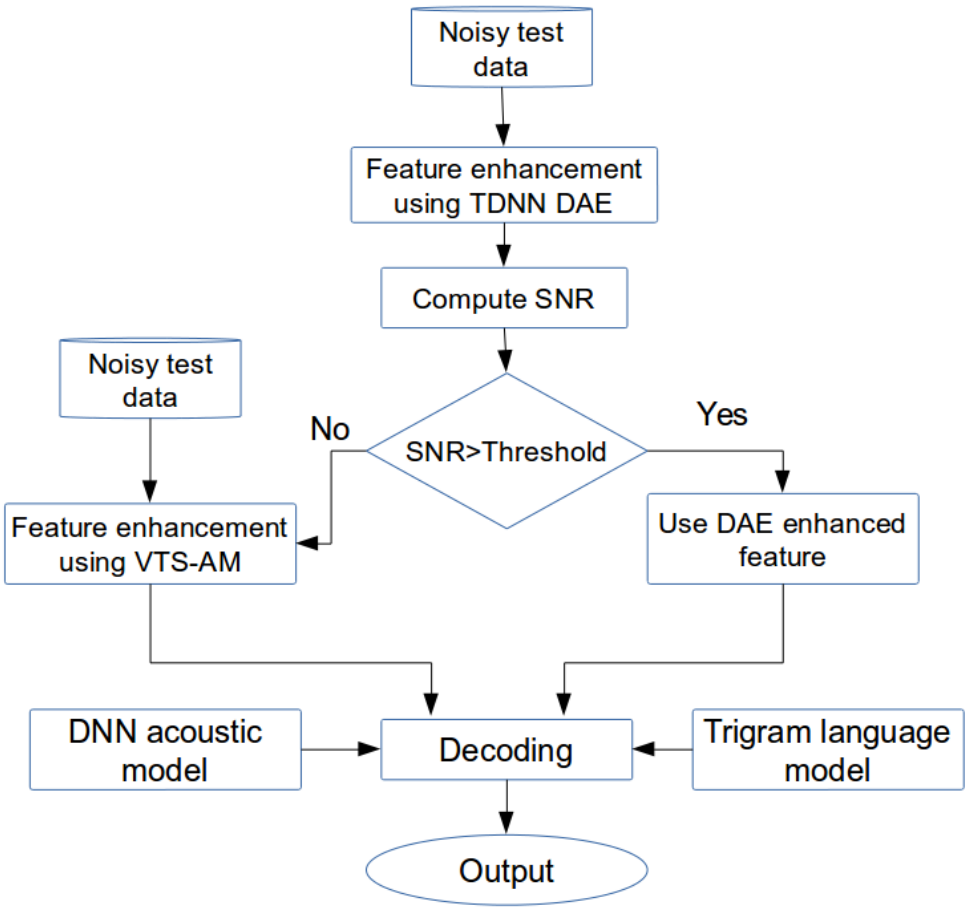
VTS-AM

- $\vec{y}_e = \vec{x}_e + \vec{n}_e$ is not a good approximation of the actual corruption process.
- Why? Clean speech energy $\vec{x}_e$ can mask (render inaudible) a certain portion of the noise energy
- Introduce acoustic masking, namely, $\vec{y}_e = \vec{x}_e + \vec{n}_e - \vec{T}_e$ (based on human perception)
  - $\vec{T}_e$ defined by MPEG standards
- VTS-AM outperforms the VTS by absolute 2% in speech recognition tasks

The Scheme


Noisy test data
Feature enhancement using TDNN DAE
Compute SNR
Noisy test data
No
SNR>Threshold
Yes
Feature enhancement using VTS-AM
Use DAE enhanced feature
DNN acoustic model
Decoding
Trigram language model
Output

The Scheme



Using this integrated approach, the average improvement in performance for unseen noise (mismatch) is 14% absolute, while maintaining the performance improvement for seen noise (namely 20%).
For Librispeech database.

Conclusions

- Train-Test condition mismatch is common
- And data corresponding to test conditions might be sparse
- This often blunts the performance of deep learning architectures (TDNN-DAE)
- One needs to look at the novel ways of overcoming this limitation.

Combining VTS-AM and DAE

overcomes this limitation of the Denoising Auto Encoder in train-test mismatch conditions.

Reference

Biswajit Das, Ashish Panda, "Vector Taylor Series Expansion with Auditory Masking for Noise Robust Speech Recognition", IEEE SigPort, 2016.

go back to Main

Knowledge driven Feed Forward Neural Networks

Based On

Sri Harsha Dumpala, Rupayan Chakraborty and Sunil Kumar Kopparapu, "Knowledge driven feed-forward neural network for audio affective content analysis", The AAAI-18 Workshop on Affective Content Analysis, AffCon2018, Feb 2018

Knowledge driven Feed Forward Neural Networks

Background

FFNN

- Traditional FFNN train on input (I) output (O) pairs
- Fine tune (update) weights so that $\|\hat{O} - O\|_2$ ($\hat{O} \rightarrow$ predicted)
  - Use back propagation algorithm
  - Can work for any input-output pairs (no knowledge about the data either used or exploited)

Recurrent Neural Network

- RNN has the ability to exploit (automatically learn) the temporal relationship in data (even if it does not exist!)
- However, RNN's
  - Requires **substantial amount of training data** for better learning
  - Requires **changes** in the representation of input-output pair

What if you had some prior information about the data?

k-FFNN: Knowledge driven FFNN

What if you had some prior information about the data?

k-FFNN Framework

- Traditional FFNN which allows for incorporating a-priori knowledge about the temporal correlations in the training data
- **No changes** in FFNN architectures
- Only **changes** in the representation of input-output pair

Useful?

- Example: Affective impact of movies: MediaEval 2016
- Task: Given a short video clip (around 10 seconds) predict a score of induced valence (negative-positive) and induced arousal (calm-excited) for the entire clip

Affective impact of movies: MediaEval 2016

Task

Global prediction for short video excerpts.

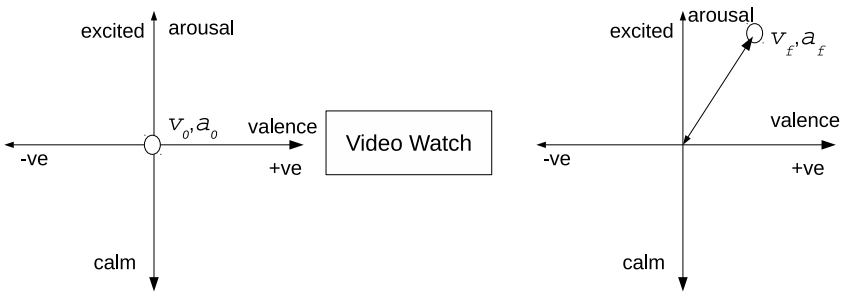
Given a short video clip (around 10 seconds), build a system to predict a score of induced valence (negative-positive) and induced arousal (calm-excited) for the whole clip

Affective impact of movies: MediaEval 2016

Task

Global prediction for short video excerpts.

Given a short video clip (around 10 seconds), build a system to predict a score of induced valence (negative-positive) and induced arousal (calm-excited) for the whole clip



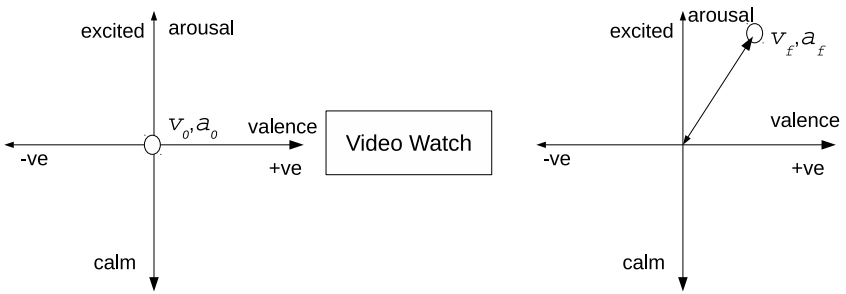
Perceived Emotion after a viewer watches a short video

Affective impact of movies: MediaEval 2016

Task

Global prediction for short video excerpts.

Given a short video clip (around 10 seconds), build a system to predict a score of induced valence (negative-positive) and induced arousal (calm-excited) for the whole clip



Perceived Emotion after a viewer watches a short video

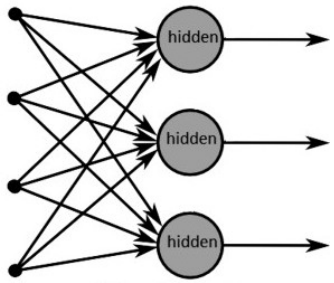
What is given?

A learning set of videos (c_k) and the corresponding emotion (valance v_k)

Possibilities

Use $(c_k; v_k)$ to learn. Break the video into 1 sec frames $\Longrightarrow$ learn $(c_{k0}, \cdots, c_{k9}; v_k)$

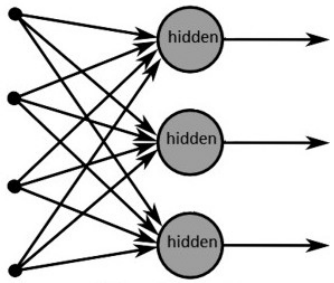
FFNN vs RNN: Architecture and data

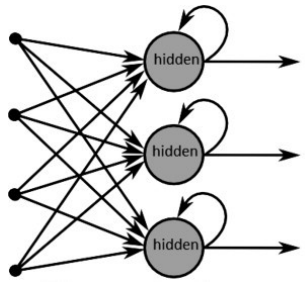


Input				Output
$g^1_{1,1}$	$g^2_{1,1}$	..	$g^I_{1,1}$	v_1
$g^1_{1,2}$	$g^2_{1,2}$	..	$g^I_{1,2}$	v_1
..	..	..	..	..
$g^1_{1,N}$	$g^2_{1,N}$	..	$g^I_{1,N}$	v_1
$g^1_{2,1}$	$g^2_{2,1}$	..	$g^I_{2,1}$	v_2
$g^1_{2,2}$	$g^2_{2,2}$	..	$g^I_{2,2}$	v_2
..	..	..	..	..
$g^1_{2,N}$	$g^2_{2,N}$	..	$g^I_{2,N}$	v_2

Table: FFNN:input-output pair

FFNN vs RNN: Architecture and data





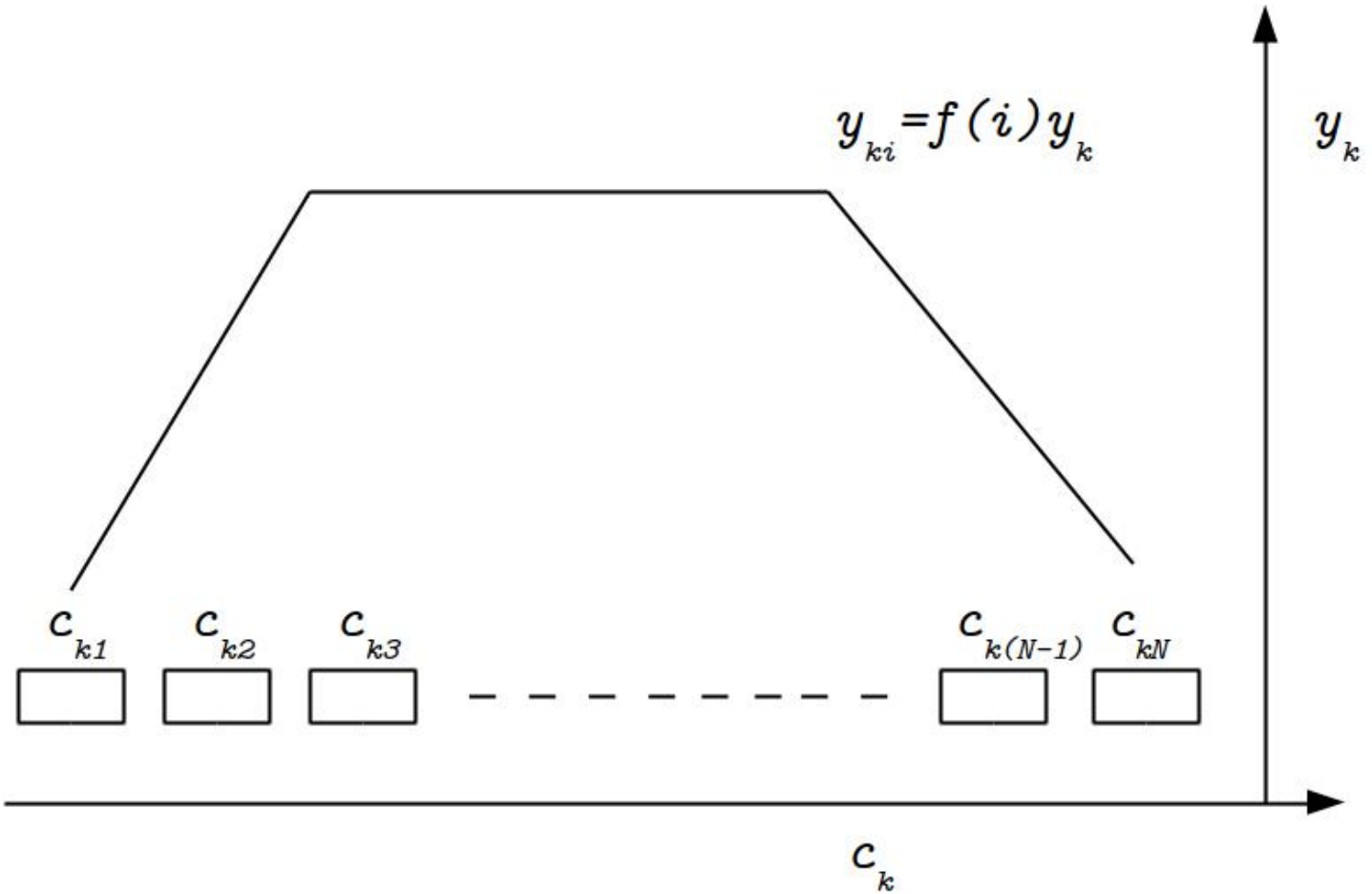
Input				Output
$g^1_{1,1}$	$g^2_{1,1}$	..	$g^I_{1,1}$	v_1
$g^1_{1,2}$	$g^2_{1,2}$	..	$g^I_{1,2}$	v_1
..	..	..	..	..
$g^1_{1,N}$	$g^2_{1,N}$	..	$g^I_{1,N}$	v_1
$g^1_{2,1}$	$g^2_{2,1}$	..	$g^I_{2,1}$	v_2
$g^1_{2,2}$	$g^2_{2,2}$	..	$g^I_{2,2}$	v_2
..	..	..	..	..
$g^1_{2,N}$	$g^2_{2,N}$	..	$g^I_{2,N}$	v_2

Table: FFNN:input-output pair

Input				Output
$g^1_{1,1}$	$g^2_{1,1}$	..	$g^I_{1,1}$	-
$g^1_{1,2}$	$g^2_{1,2}$	..	$g^I_{1,2}$	-
..	..	..	..	-
$g^1_{1,N}$	$g^2_{1,N}$	..	$g^I_{1,N}$	v_1
$g^1_{2,1}$	$g^2_{2,1}$	..	$g^I_{2,1}$	-
$g^1_{2,2}$	$g^2_{2,2}$	..	$g^I_{2,2}$	-
..	..	..	..	-
$g^1_{2,N}$	$g^2_{2,N}$	..	$g^I_{2,N}$	v_2

Table: RNN:input-output pair

Observe Knowledge in Data!

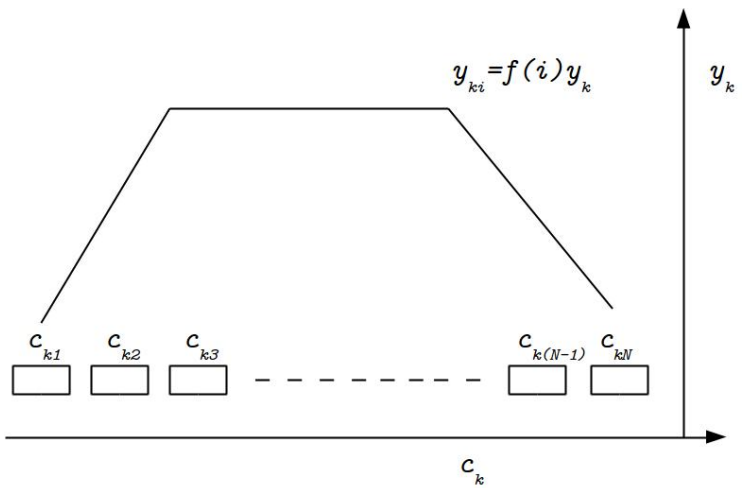


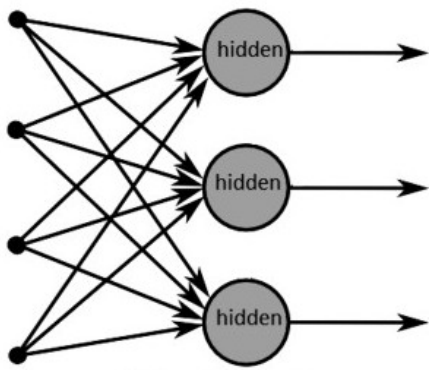
- For any movie clip, to evoke emotion one has to make sure that the emotion (valence) is highest in the middle of the clip (Why?).
- This can be exploited as prior knowledge!

k-FFNN

Input				Output
$g^1_{1,1}$	$g^2_{1,1}$	..	$g^I_{1,1}$	$f(1)v_1$
$g^1_{1,2}$	$g^2_{1,2}$	..	$g^I_{1,2}$	$f(2)v_1$
..	..	..	..	..
$g^1_{1,N}$	$g^2_{1,N}$	..	$g^I_{1,N}$	$f(N)v_1$
$g^1_{2,1}$	$g^2_{2,1}$	..	$g^I_{2,1}$	$f(1)v_2$
$g^1_{2,2}$	$g^2_{2,2}$	..	$g^I_{2,2}$	$f(2)v_2$
..	..	..	..	..
$g^1_{2,N}$	$g^2_{2,N}$	..	$g^I_{2,N}$	$f(N)v_2$

Table: k-FFNN:input-output pair





Conclusions

- Conventional FFNN does not make use of relationship between data
- RNN makes use of these relationships.

  So RNN's perform better that FFNN.
- For RNN to be effective we need lot more data
- In the absence of such data performance of RNN is poor
- Possible solution is to exploit apriori knowledge

  Use k-FFNN: Performance in the absence of large data

Reference

Sri Harsha Dumpala, Rupayan Chakraborty and Sunil Kumar Kopparapu, "Knowledge driven feed-forward neural network for audio affective content analysis", The AAAI-18 Workshop on Affective Content Analysis, AffCon2018, Feb 2018

go back to Main

Simultaneous 2 Class Learning

Based On

- Sri Harsha Dumpala, Rupayan Chakraborty and Sunil Kumar Kopparapu, "A novel approach for effective learning in low resourced scenarios", Workshop Machine Learning for Audio Signal Processing at NIPS 2017 (ML4Audio), Long Beach, USA.
- Indian **Patent** (Provisional) System and Method for Simultaneous Multi-class Learning PS # 201721017694 (29-May-2017) Tata Consultancy Services (Dumpala, S. H., Chakraborty, R. Kopparapu, S. K.)

Simultaneous 2 Class Learning

Motivation

- **Neural networks**, being state-of-the-art discriminative algorithms, require **substantial amount of training data**.
- What do we do when we have **limited data?**
- Can we **generate sufficient amount of data**, from the available **limited data**, for neural networks to learn better?

One such **approach** *is simultaneous 2 class learning*

Objective

- Propose novel approach to **effectively learn** the system parameters even from **limited data**.

Contributions

- Representation of data for learning effectively from limited number of examples.
- Introduce modifications to neural network for handling proposed data representation.
- Novel decision making based on the proposed data representation.

Proposed approach

Conventional learning

$$(\vec{x}_{ij}, C_i), \quad i = 1, 2; \quad j = 1, 2, \cdots, N_i$$

where $\vec{x}_{ij}$ **:** Feature vectors**;** C_i**:** Class label**;** N_i**:** Number of samples in class i.

Proposed approach

Conventional learning

$$(\vec{x}_{ij}, C_i), \quad i = 1, 2; \quad j = 1, 2, \cdots, N_i$$

where $\vec{x}_{ij}$ **:** Feature vectors**;** C_i**:** Class label**;** N_i**:** Number of samples in class i.

Proposed data representation format

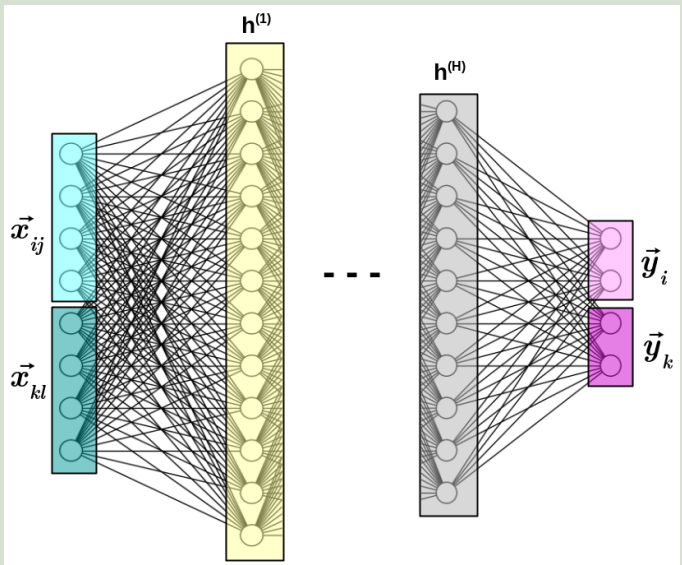
$$([\vec{x}_{ij}, \vec{x}_{kl}], [C_i, C_k]), \quad \forall i, k = 1, 2; j = 1, 2, \cdots, N_i \quad \text{and} \quad l = 1, 2, \cdots, N_k,$$

where $\vec{x}_{ij}$, $\vec{x}_{kl} \in \mathbb{R}^{d \times 1}$**:** Feature vectors**;** C_i, C_k**:** Class labels**;** N_i, N_k**:** Number of class samples.

- Each sample is obtained by simultaneously considering two samples, hence, the name "**simultaneous two sample (s2s)**" representation.
- Using s2s representation, the number of samples are increased to $(N_1 + N_2)^2$ from $(N_1 + N_2)$ samples.

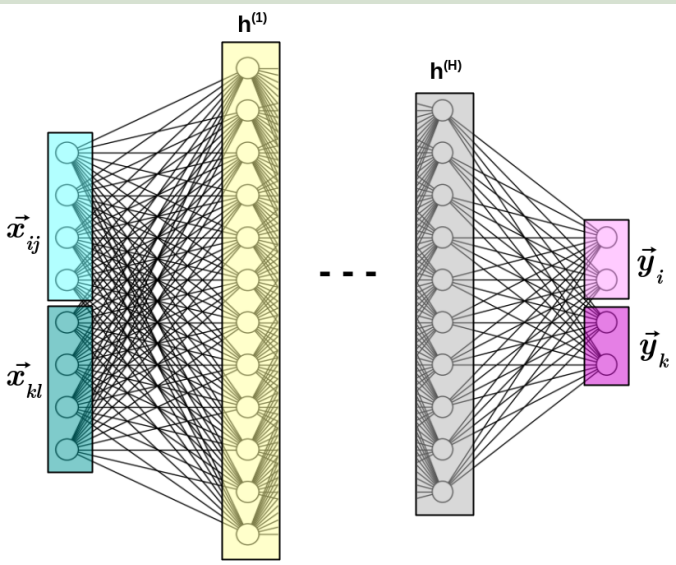
Neural Network:s2s MLP

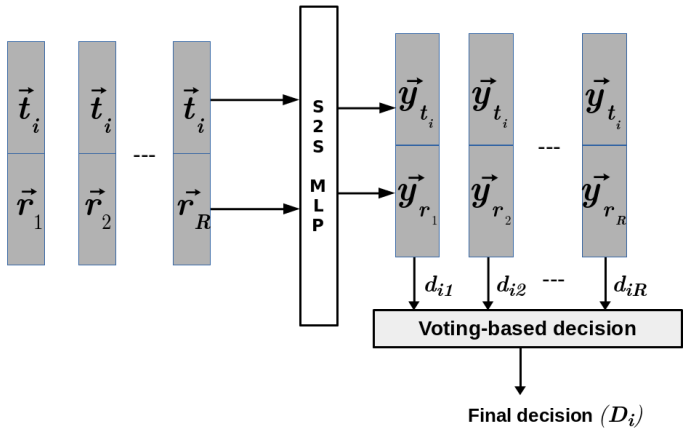
MLP architecture is modified to s2s-MLP as shown



Neural Network:s2s MLP

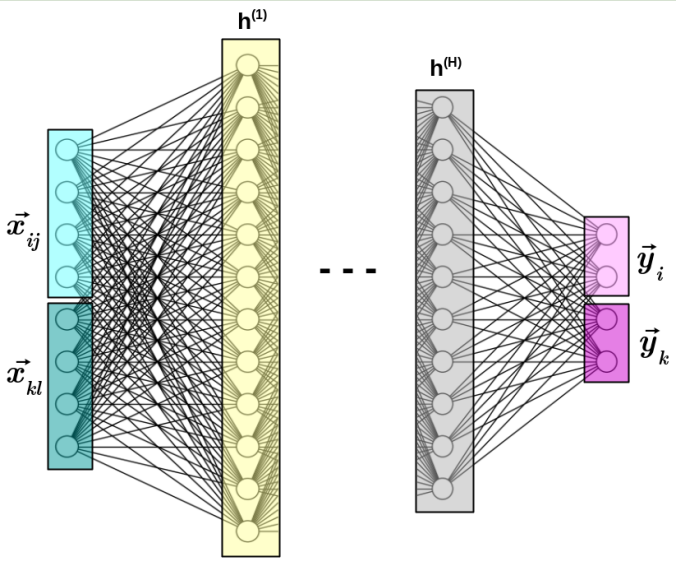
MLP architecture is modified to s2s-MLP as shown

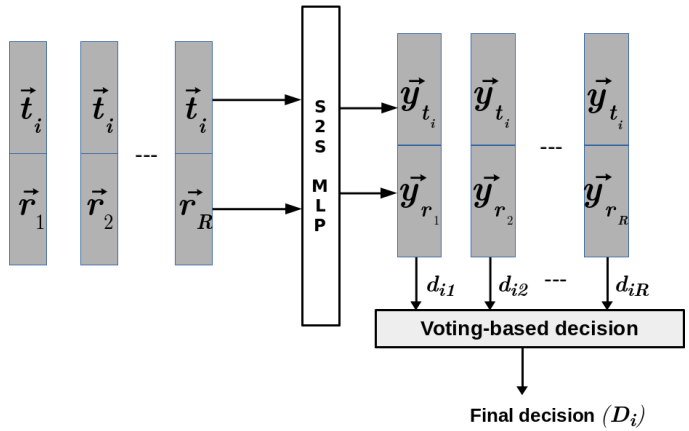




Neural Network:s2s MLP

MLP architecture is modified to s2s-MLP as shown





- $\vec{x}_{ij}$ and $\vec{x}_{kl}$ are two samples considered simultaneously.
- $\vec{y}_i$ and $\vec{y}_k$ are the corresponding output labels.
- Sigmoid units are used at output.
- Training of the s2s-MLP using the s2s data representation format is referred to as **simultaneous two sample learning (s2sL)**.

Conclusions

- Data imbalance is a problem
- Example: "Happy" callers to a call center compared to "angry" callers.
- s2s is a novel approach.
- Experimental Results (irrespective of the size of training set)
  - For music/speech classification, average improvement of 4% achieved
  - For neutral/sad emotion classification, average improvement of 2.5% achieved

Reference

(1) Sri Harsha Dumpala, Rupayan Chakraborty and Sunil Kumar Kopparapu, "A novel approach for effective learning in low resourced scenarios", Workshop Machine Learning for Audio Signal Processing at NIPS 2017 (ML4Audio), Long Beach, USA.

(2) Indian **Patent** (Provisional)
  - System and Method for Simultaneous Multi-class Learning
  - PS # 201721017694 (29-May-2017)
  - Tata Consultancy Services (Dumpala, S. H., Chakraborty, R. & Kopparapu, S. K.)

go back to Main

Dysarthric Speech Recognition

Based On

Bhavik Vachhani, Chitralekha Bhat, Biswajit Das and Sunil Kumar Kopparapu, "Deep Autoencoder based Speech Features for Improved Dysarthric Speech Recognition", INTERSPEECH 2017, Stockholm, Sweden.

Dysarthric Speech Recognition

Problem

- Dysarthria is a motor speech disorder.
- Affects the movement of individual speech muscles.
- It can vary from mild slurring to completely unintelligible speech.
- Poor articulation of phonemes.

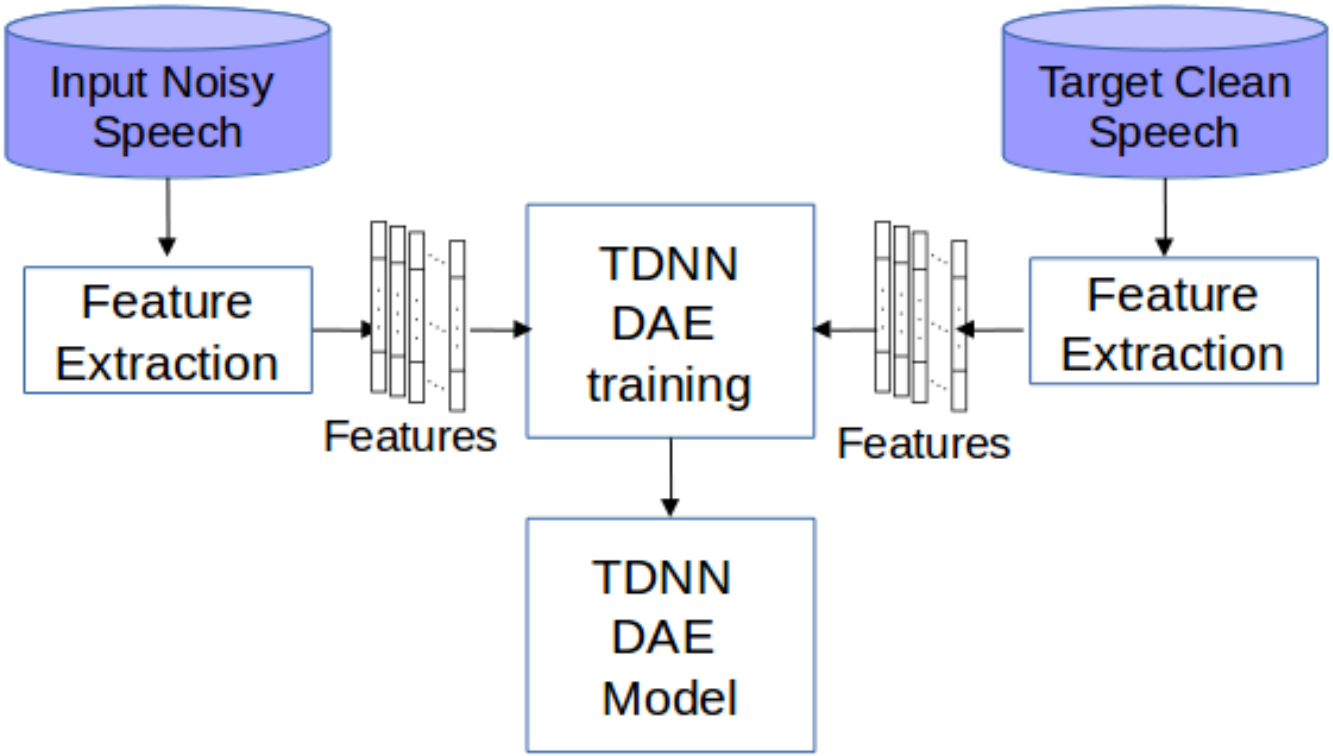
$$\text{Dysarthria} \longrightarrow \downarrow \text{Intelligibility} \longrightarrow \downarrow \text{ASR performance}$$

- Automatic speech recognition performance degrades as the system encounters dysarthric speech.
- One way to deal with the noisy speech is to estimate a robust feature set from a given noisy feature set.

How?

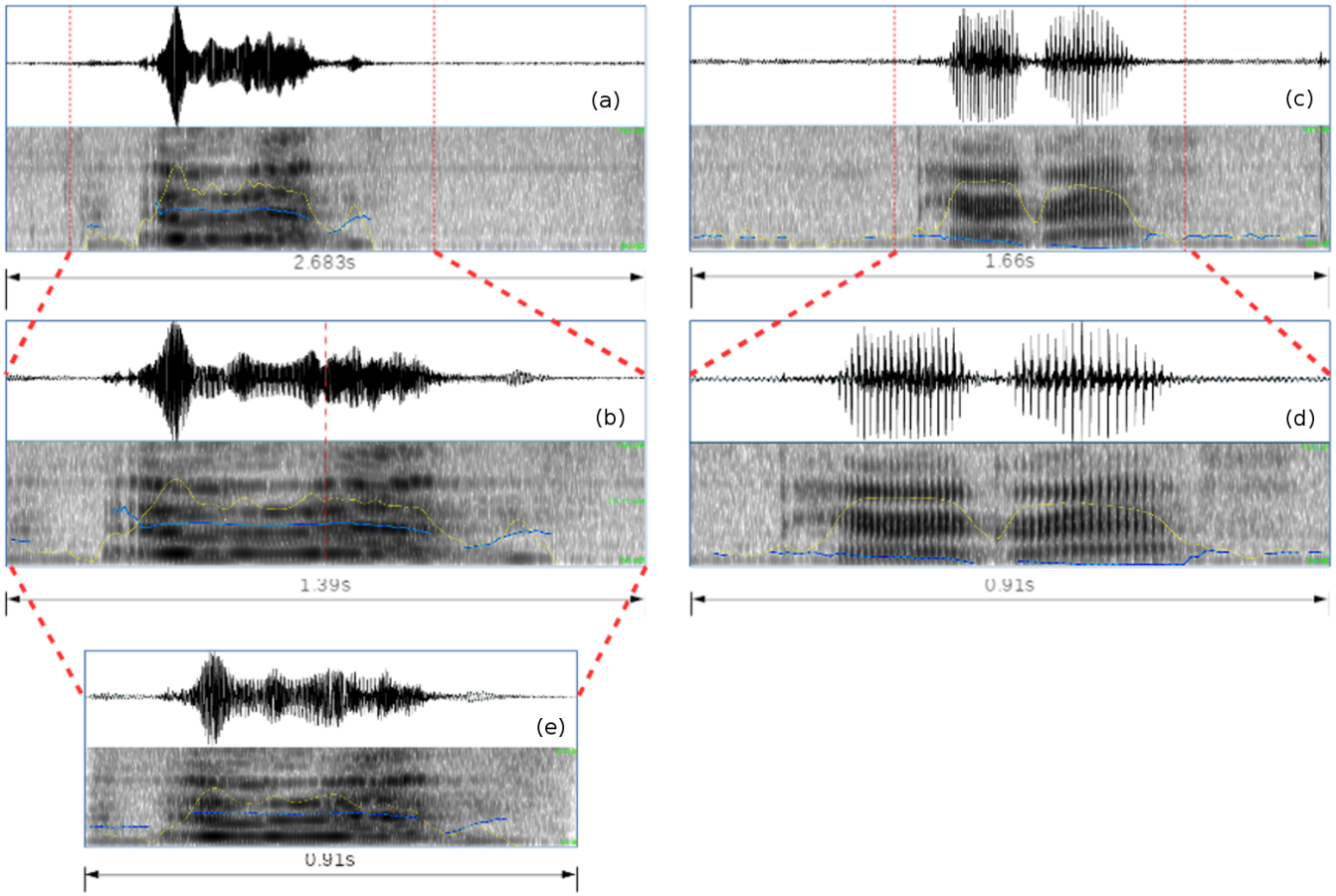
Use Deep Learning, namely, Denoising Auto Encoder (DAE).

TDNN - DAE (Denoising Auto Encoder)



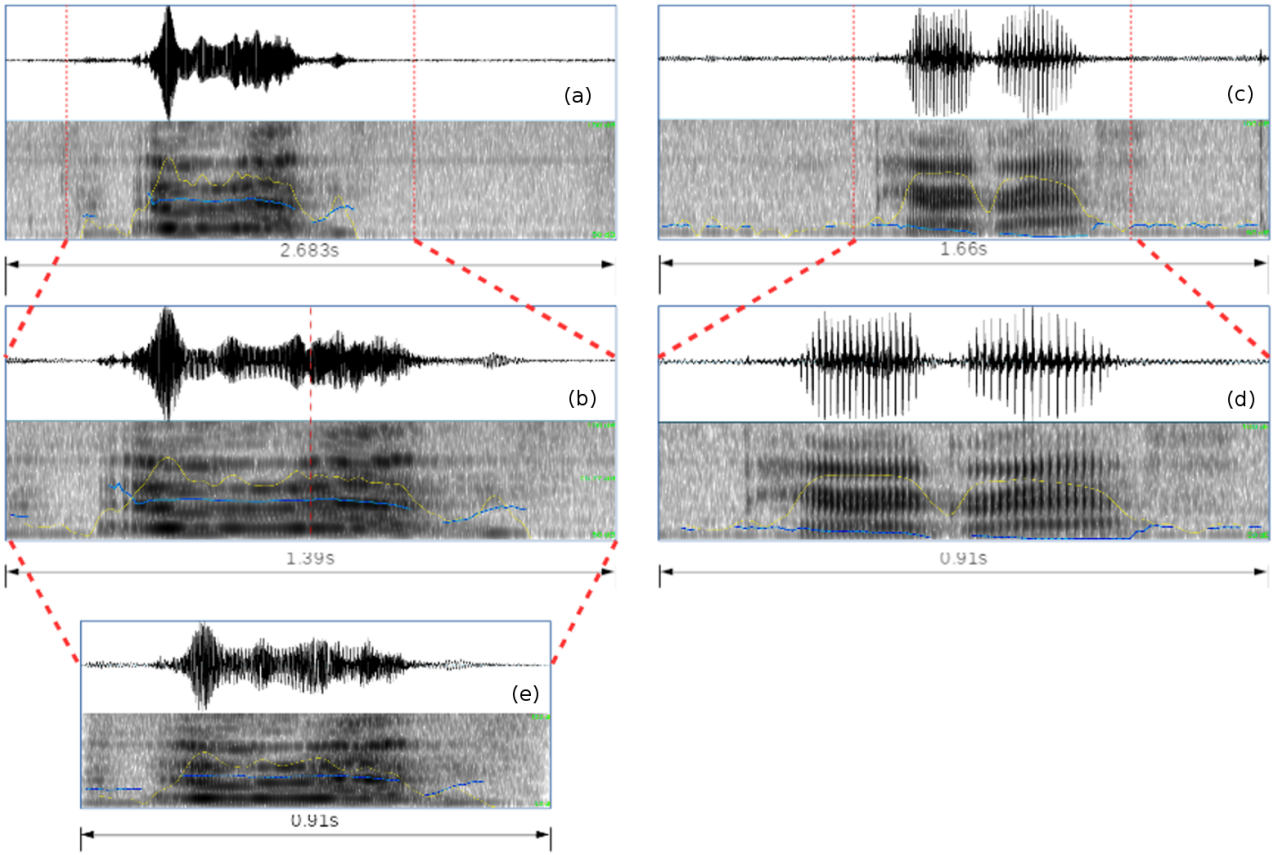
- Noisy speech is the Dysarthric speech
  - Usually depending on the severity level the length of Dysarthric speech is longer than normal speech!
- How does one use DAE?
  - matching of number of frames in dysarthric and control speech utterance!

Temporal Adaptation



- Speech part and 200 ms of silence on either sides of the speech was retained for both dysarthric and healthy control speech.
- Temporal adaptation of dysarthric speech utterance was done to match the healthy control counterpart using phase vocoder.

Temporal Adaptation



(a) Original dysarthric utterance (2.68s)
(b) Dysarthric utterance after end point silence removal (1.39s)
(c) Original healthy control utterance of duration (1.66s)
(d) Healthy Control utterance after end point silence removal (0.91s)
(e) Dysarthric utterance after tempo adaptation (0.91s) to match (d)

Data for TDNN-DAE

Data - Universal Access (UA) Speech Corpus

- UA dysarthric speech corpus comprises data from **13 healthy control (HC)** + **15 dysarthric (DYS)** speakers with cerebral palsy.
- 3 blocks of data B1, B2 and B3 were collected for each speaker
- In each block a speaker recorded 10 digits, 26 international radio alphabets, 19 **computer commands**, 100 common words and 100 uncommon words such that each speaker recorded $255(= 10 + 26 + 19 + 100 + 100)$ distinct words and a total of $765(= 255x3)$ isolated words.

Data Augmentation

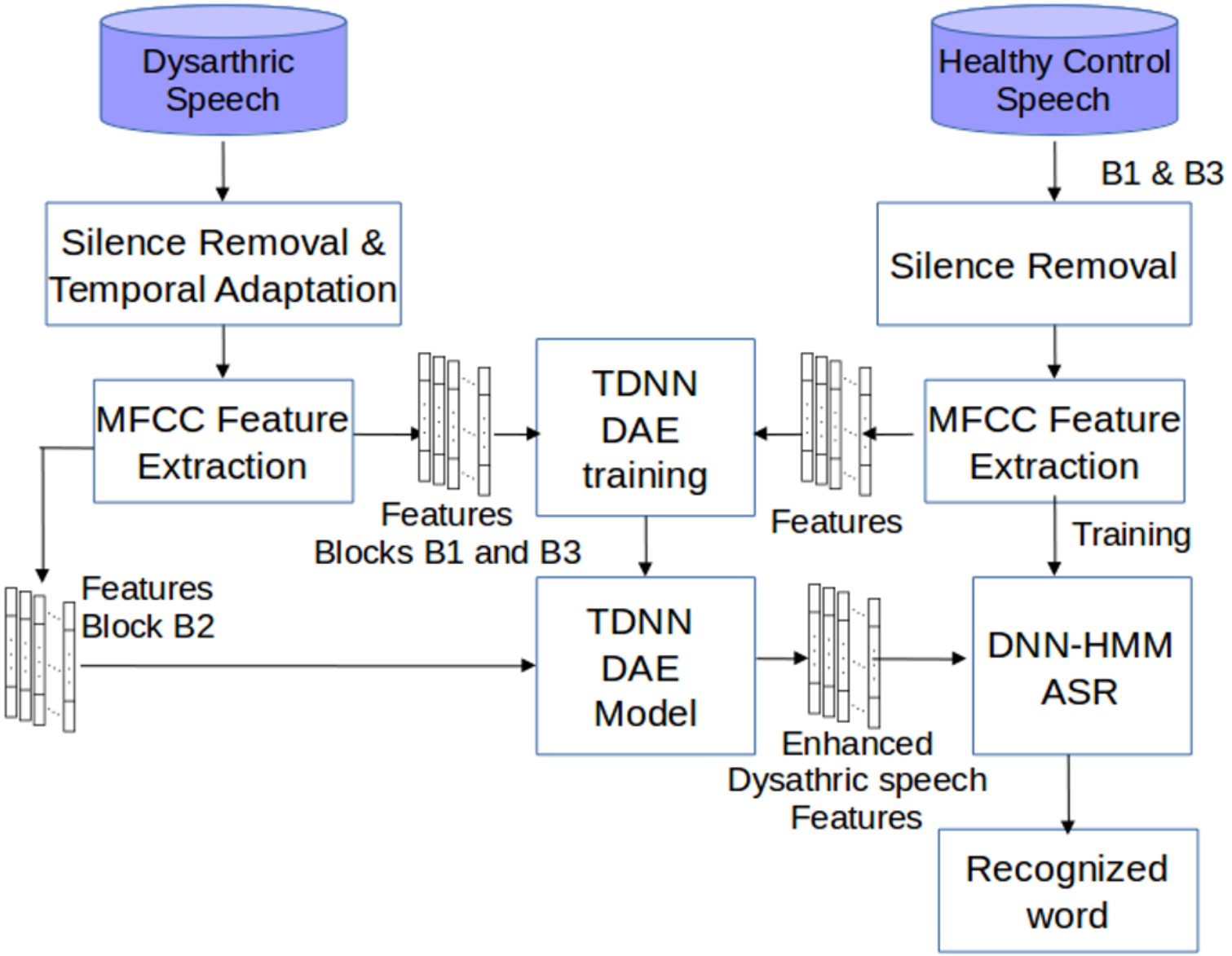
- 3511 dysarthric utterances were temporally adapted
- For every computer command (CC) $u_i \in i = 1:19$

$$^{ta}DYS_{u_i j} = f(DYS_j, HC_{u_i})$$

where DYS_j dysarthric utterance $j = 1:3511$, HC_{u_i} healthy control CC utterances with $i = 1:19$ and f is the temporal adaptation (TA) function.

- $^{ta}DYS_{u_i j}$ and HC_{u_i} used for training TDNN-DAE

Proposed System for Improved ASR



Brief Results

Training Configuration

System	Train Conf	Test Conf
S-1	HC-CC (B1,B3)	^{ta} DYS-CC (B2)
S-2	S-1 + DYS-CC(B1,B3)	
S-3	S-1 + ^{ta}DYS-CC(B1,B3)	

Brief Results

Training Configuration

System	Train Conf	Test Conf
S-1	HC-CC (B1,B3)	^{ta} DYS-CC (B2)
S-2	S-1 + DYS-CC(B1,B3)	
S-3	S-1 + ^{ta}DYS-CC(B1,B3)	

WER for TDNN-DDA with session mismatch

Train Conf	MFCC-TA		MFCC-TA +TDNN-DAE	
	SA	SI	SA	SI
S-1	-	37.86	-	24.73
S-2	21.44	33.67	60.8	29.7
S-3	82.69	72.47	18.54	34.39

Brief Results

Training Configuration

System	Train Conf	Test Conf
S-1	HC-CC (B1,B3)	^{ta} DYS-CC (B2)
S-2	S-1 + DYS-CC(B1,B3)	
S-3	S-1 + ^{ta}DYS-CC(B1,B3)	

Severity level analysis of WER

Severity	S-2 MFCC-TA-SA	S-3 TDNN-DAE-SA	Absolute Improvement
Very Low	5.71	1.35	4.4
Low	11.39	9.4	1.99
Medium	22.67	19.46	3.2
High	57	52.5	4.5

Conclusions

- TDNN-DAE used to enhance the dysarthric speech features (MFCC).
- Silence removal followed by temporal adaptation used to match the frame numbers of dysarthric utterances with corresponding healthy control utterances.
- An absolute improvement of 15% and 3% was observed in ASR performance for session and speaker mismatch respectively.

Reference

Bhavik Vachhani, Chitralekha Bhat, Biswajit Das and Sunil Kumar Kopparapu, "Deep Autoencoder based Speech Features for Improved Dysarthric Speech Recognition", INTERSPEECH 2017, Stockholm, Sweden.

go back to Main

ANN output error correction using ECC

Based On

Rupayan Chakraborty, Sunil Kopparapu, "ECC-ANN and DNN based Speech Emotion Recognition ", 2016 IEEE International Conference on Systems, Man, and Cybernetics, Budapest, Hungary, 2016.

ANN output error correction using ECC

Background

- Audio Emotion recognition system is viewed as noisy communication channel
  - Insufficiently learnt artificial neural network is the noisy communication channel, and gives erroneous emotion classification
- Can Error Correcting Codes (ECC) be used in multi-class speech emotion recognition
  - By encoding the emotion class using a Block Coder and correcting the output of ANN using error correction decoding

ANN output error correction using ECC

Problem formulation

- Emotions are considered in categorical space
  - Classes: Anger, Boredom, Disgust, Fear, Happy, Neutral, Sad
- Categorical emotions are encoded by binary bit-stream
  - 7 emotion class problem: use 7 bit binary encoded word

ANN output error correction using ECC

Problem formulation

- Emotions are considered in categorical space
  - Classes: Anger, Boredom, Disgust, Fear, Happy, Neutral, Sad
- Categorical emotions are encoded by binary bit-stream
  - 7 emotion class problem: use 7 bit binary encoded word

Emotions represented in terms of 7 bits

o_1	o_2	o_3	o_4	o_5	o_6	o_7	Emotion
1	0	0	0	0	0	0	anger
0	1	0	0	0	0	0	boredom
0	0	1	0	0	0	0	disgust
0	0	0	1	0	0	0	fear
0	0	0	0	1	0	0	happy
0	0	0	0	0	1	0	neutral
0	0	0	0	0	0	1	sad

Details

Formulation

- Discriminative classifier like ANN is considered as a communication channel
  - ANN takes audio features as input
  - Transmits class information through the network
  - Outputs binary word (bit stream)

Details

Formulation

- Discriminative classifier like ANN is considered as a communication channel
  - ANN takes audio features as input
  - Transmits class information through the network
  - Outputs binary word (bit stream)

What causes errors? Does not matter!

- ANN acts like a noisy communication channel and gives erroneous output

Details

Formulation

- Discriminative classifier like ANN is considered as a communication channel
  - ANN takes audio features as input
  - Transmits class information through the network
  - Outputs binary word (bit stream)

What causes errors? Does not matter!

- ANN acts like a noisy communication channel and gives erroneous output
  - Errors due to - ?
  - choice of input features-?, insufficient training samples-?, learning algo-?

Details

Formulation

- Discriminative classifier like ANN is considered as a communication channel
  - ANN takes audio features as input
  - Transmits class information through the network
  - Outputs binary word (bit stream)

What causes errors? Does not matter!

- ANN acts like a noisy communication channel and gives erroneous output
  - Errors due to - ?
  - choice of input features-?, insufficient training samples-?, learning algo-?

Solution!

- ECC is used to handle the channel errors

Emotion Recognition System

Approach

Like a conventional Pattern Recognition problem

- feature extraction at the front end
- Classification is done by ANN
- Error correction decoding to correct errors

ECC is used to handle the channel errors

Emotion Recognition System

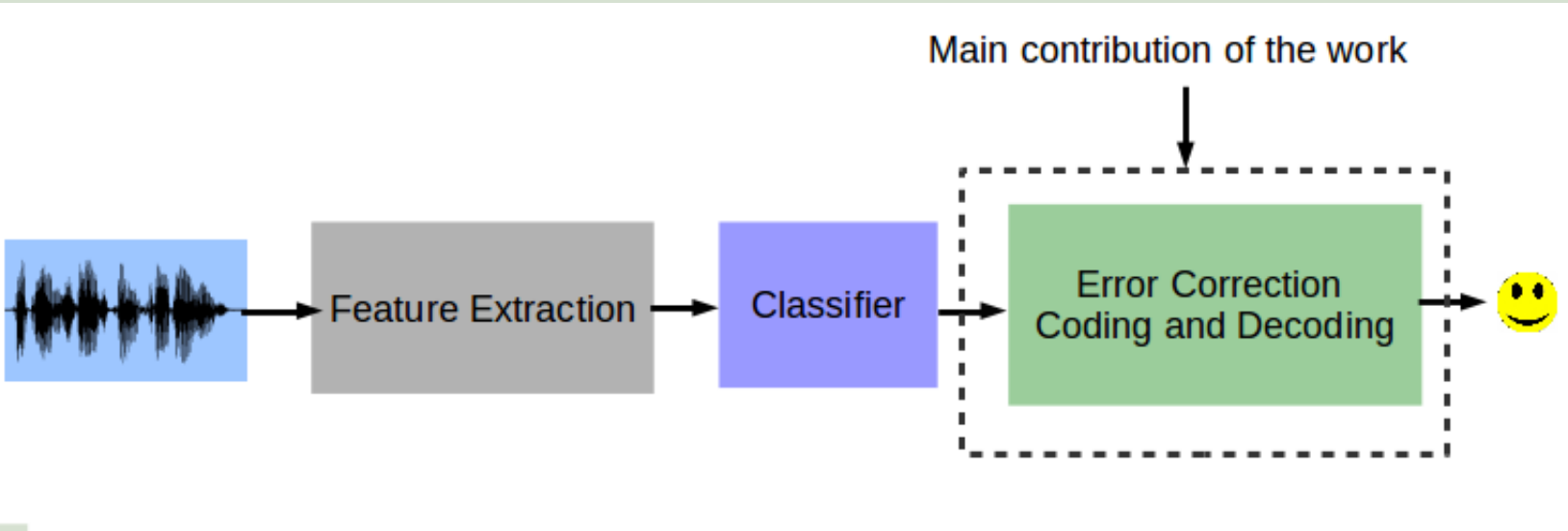
Approach

Like a conventional Pattern Recognition problem

- feature extraction at the front end
- Classification is done by ANN
- Error correction decoding to correct errors

ECC is used to handle the channel errors

Emotion recognition block



Emotion Recognition System

back to Main

Approach

- $\{f_{x_i}^j\}_{j=1}^J$ be a set of J audio features extracted from the audio clip $x_i(t)$
- 7 emotion classes are represented by k=7 bits, namely $o_1, o_2, o_3, o_4, o_5, o_6$, and o_7
- ANN has been trained using an annotated training set, namely

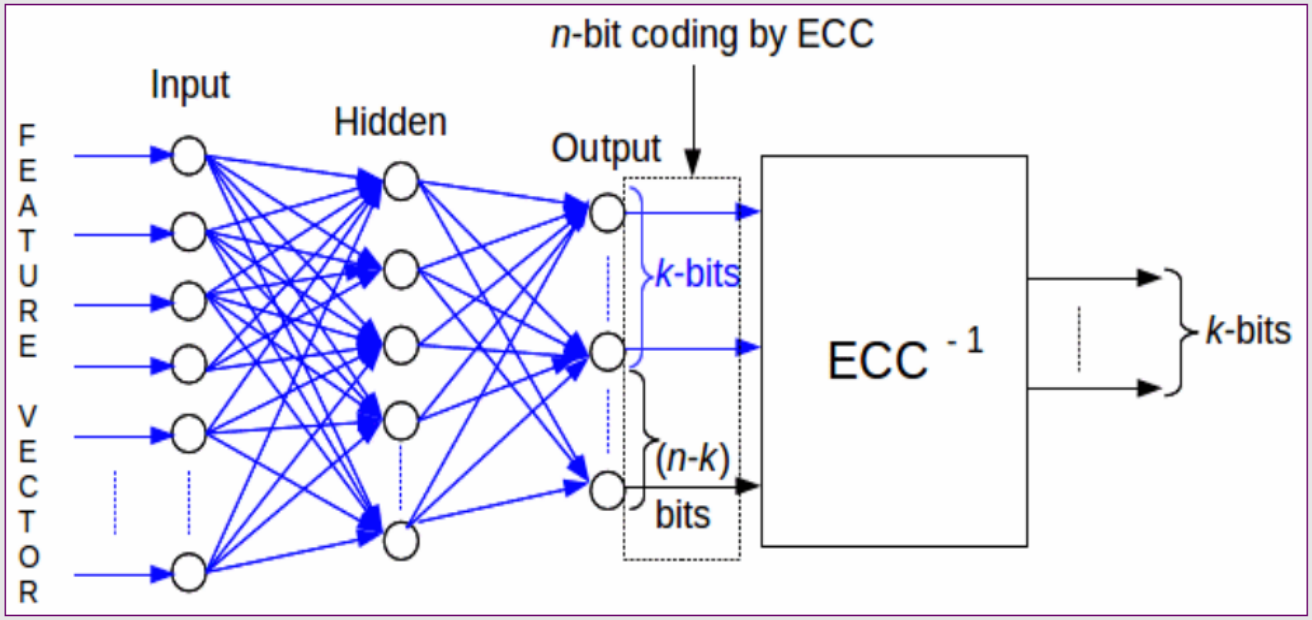
$$\{\{f_{x_i}^j\}_{j=1}^J, o_1^i, o_2^i, o_3^i, o_4^i, o_5^i, o_6^i, o_7^i\}_{i=1}^S$$

- Construct n bit binary string from 7 bit $o_1, o_2, o_3, o_4, o_5, o_6, o_7$ ($(n-7)$ are the parity bits)

Emotion Recognition System

back to Main

ECC-ANN block



Emotion Recognition System

back to Main

Some Results

On EmoDB dataset.

Architecture description		recognition accuracy
ANN (baseline)		63.75
ANN-1		57.5 (6.25 ↓)
DNN		77.5 (13.75 ↑)
	ECC Configuration	
	without error correction	with error correction
ANN + Cyclic(15,7)	58.3 (5.45 ↓)	77.5 (13.75 ↑)
ANN + Cyclic(31,7)	56.2 (7.55 ↓)	79.1 (15.35 ↑)
ANN + BCH(15,7)	59.1 (4.65 ↓)	80.3 (16.55 ↑)
ANN + BCH(31,7)	56.4 (7.35 ↓)	81.2 (17.45 ↑)
ANN + BCH*(31,7)	62.1 (1.65 ↓)	82.5 (18.75 ↑)

Emotion Recognition System

back to Main

A novel approach to make up for for the errors in ANN (can be due to lack of data)

Reference

Rupayan Chakraborty, Sunil Kopparapu, "ECC-ANN and DNN based Speech Emotion Recognition ", 2016 IEEE International Conference on Systems, Man, and Cybernetics, Budapest, Hungary, 2016.