AUDIO FEATURES & MACHINE LEARNING

E.M. Bakker

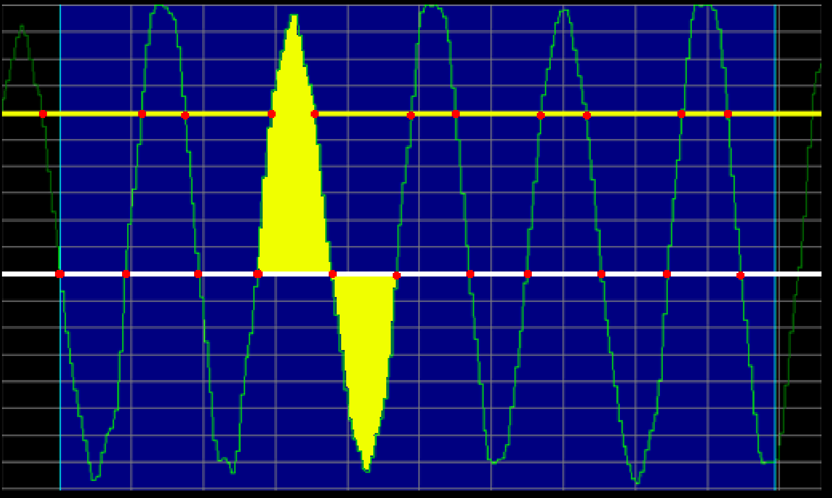
API2023

FEATURES FOR SPEECH RECOGNITION AND AUDIO INDEXING

- Parametric Representations
  - Short Time Energy
  - Zero Crossing Rates
  - Level Crossing Rates
  - Short Time Spectral Envelope
- Spectral Analysis
  - Filter Design
  - Filter Bank Spectral Analysis Model
  - Linear Predictive Coding (LPC)
  - MFCCs

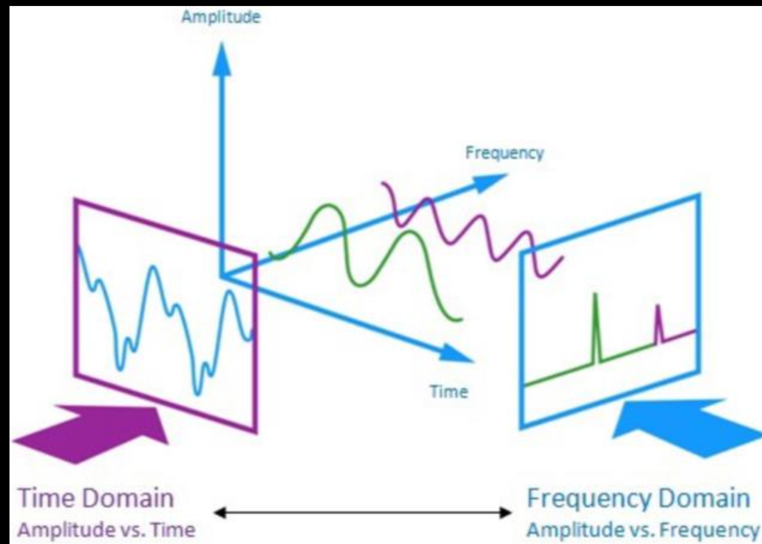
FEATURES FOR SPEECH RECOGNITION AND AUDIO INDEXING

- Parametric Representations
  - Short Time Energy
  - Zero Crossing Rates
  - Level Crossing Rates

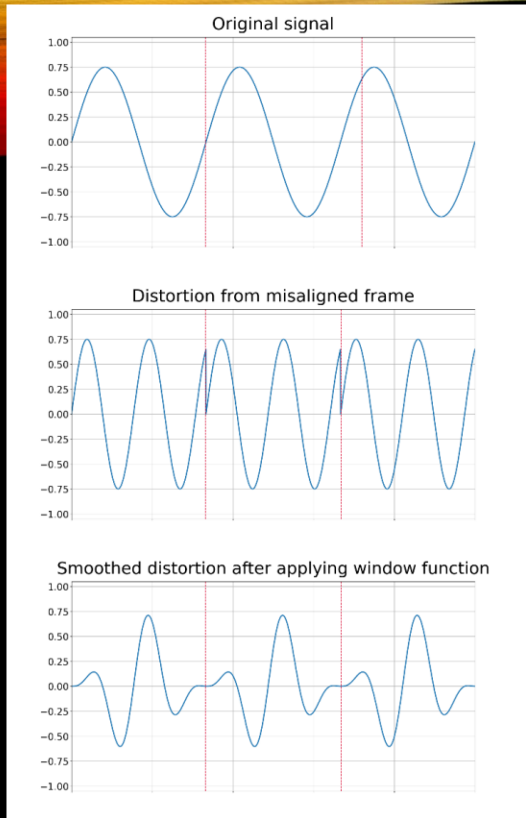


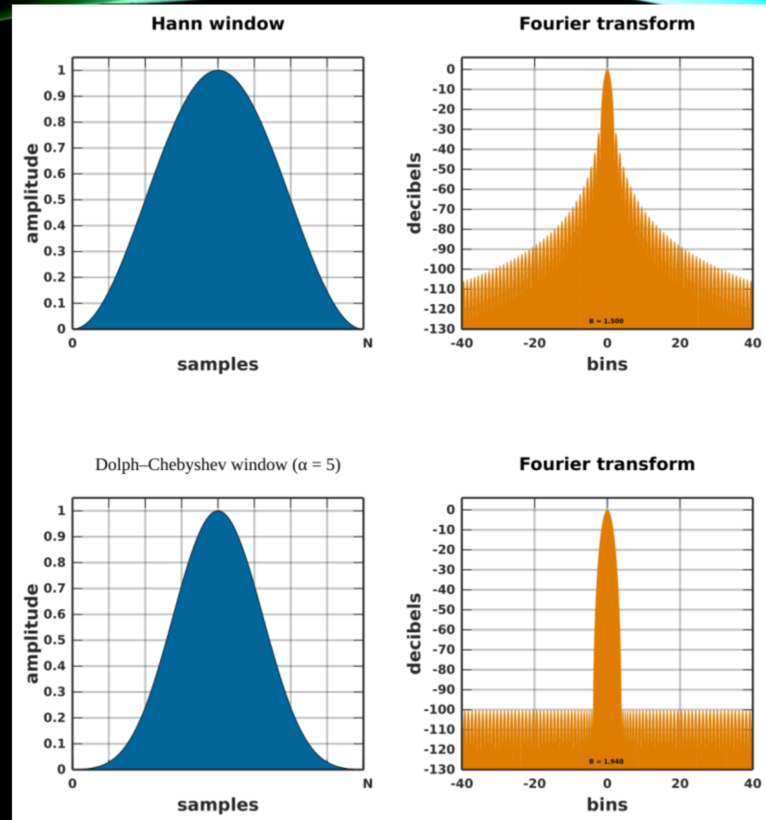
Example: Speech of length 0.01 sec.

FEATURES FOR SPEECH RECOGNITION AND AUDIO INDEXING



- Spectral Analysis
  - Fourier Transform
  - Filter Design
  - Filter Bank Spectral Analysis Model
  - Linear Predictive Coding (LPC)
    - Speech signal at time $n = s(n) \approx a_1\, s(n-1) + a_2\, s(n-2) + \ldots\, a_p\, s(n-p)$
    - Estimate $a_1 \ldots a_p$ by autocorrelation, or solving LPC analysis equations from a covariance matrix form.
  - MFCCs

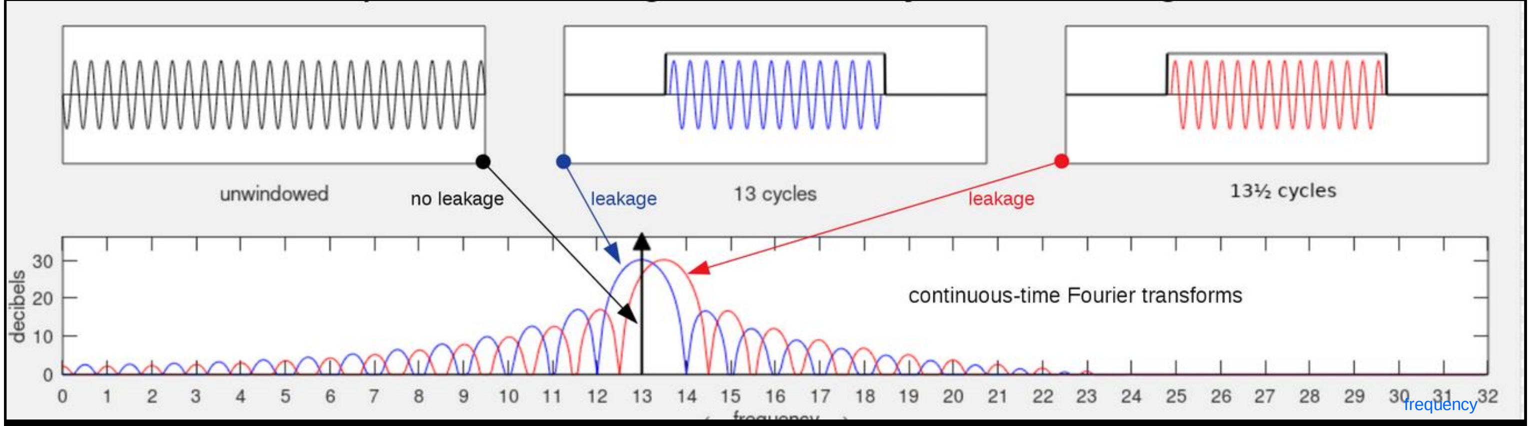




By L. de Jonckheere

- Spectral Analysis using Discrete Short Time Fourier Transform
  - Rectangular window => high resolution, low dynamic range (not good at distinguishing components of different amplitudes)
  - Hann or Hamming window => moderate

FEATURES FOR SPEECH RECOGNITION AND AUDIO INDEXING

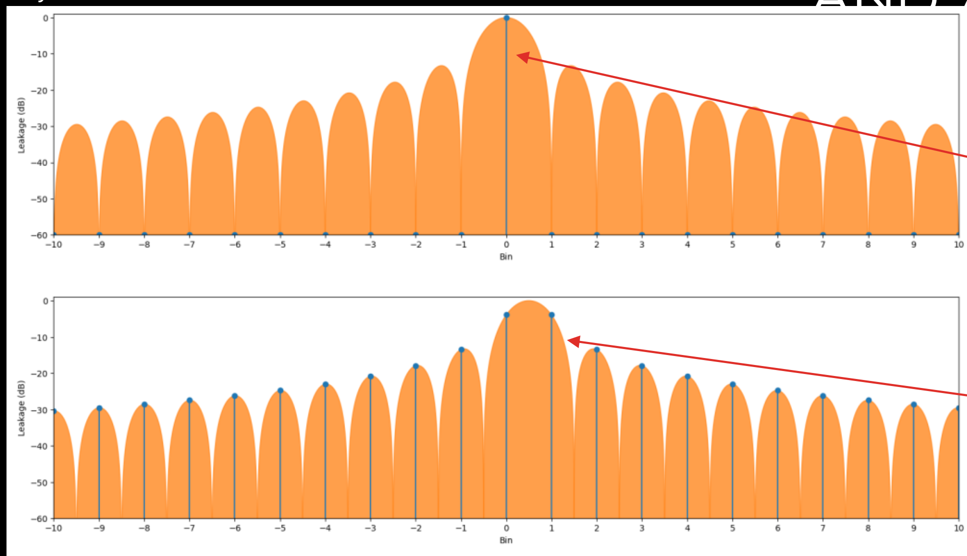


- Spectral Analysis using Discrete Short Time Fourier Transform
  - Frame of samples => frequency bins
  - Each bin corresponds to one frequency

  => Spectral leakage

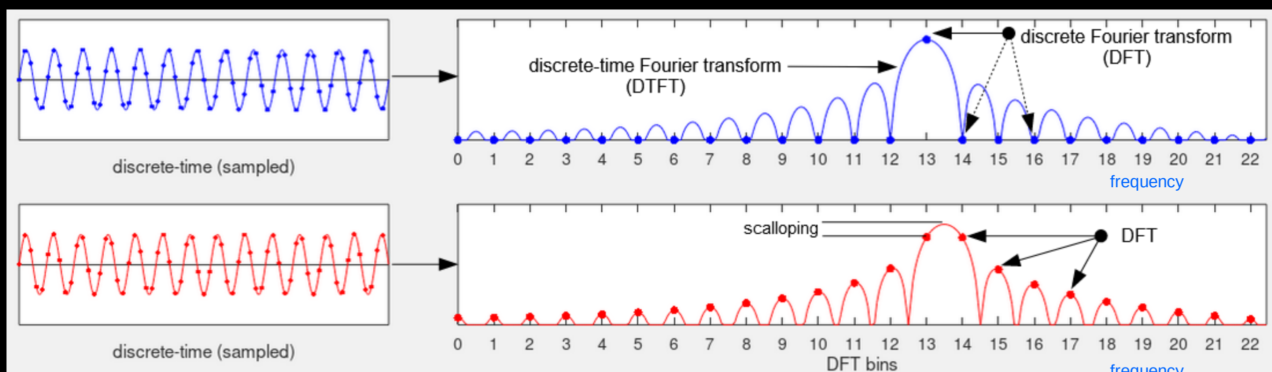
FEATURES FOR SPEECH RECOGNITION AND AUDIO INDEXING

By L. de Jonckheere



Depending on the Sampling: DFT shows the actual frequency of the signal, or shows the scalloping effect..

scalloping



SHORT TIME FOURIER TRANSFORM SHORT HAMMING WINDOW: 50 SAMPLES (=5MSEC)

Voiced Speech

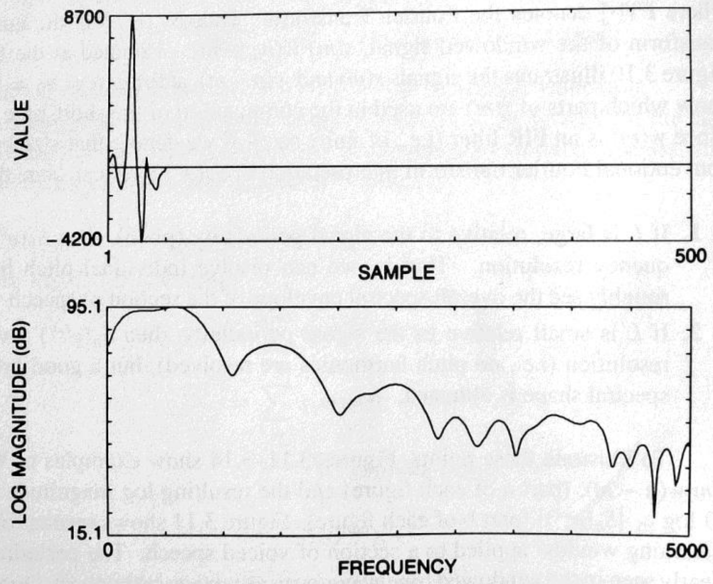


Figure 3.12 Short-time Fourier transform using a short (50 points or 5 msec) Hamming window on a section of voiced speech.

From: Rabiner et al.

Short Window

- Poor frequency resolution
- No resolved harmonics
- Good estimate of the overall spectral shape

SHORT TIME FOURIER TRANSFORM LONG HAMMING WINDOW: 500 SAMPLES (=50MSEC)

Voiced Speech

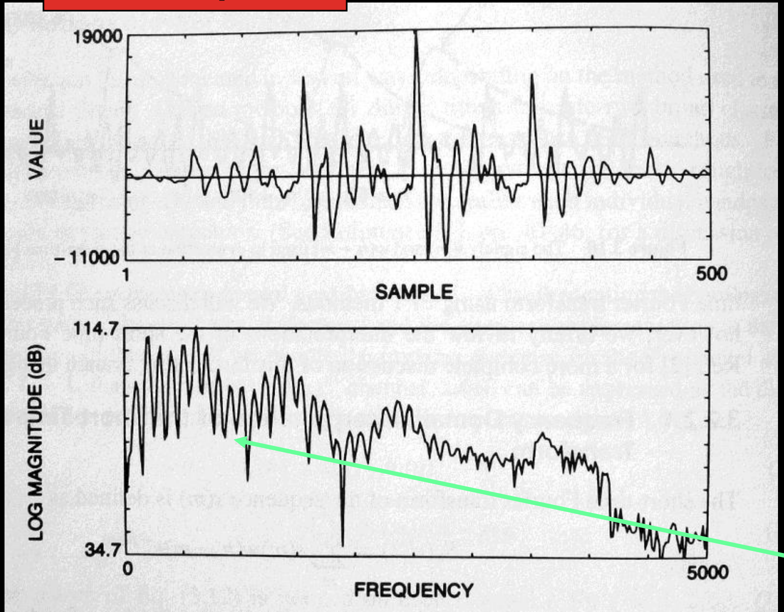


Figure 3.11 Short-time Fourier transform using a long (500 points or 50 msec) Hamming window on a section of voiced speech.

From: Rabiner et al.

Long Window

- Good frequency resolution
- Resolved harmonics
- Rough estimate of the overall spectral shape

Lower frequencies

SHORT TIME FOURIER TRANSFORM SHORT HAMMING WINDOW: 50 SAMPLES (=5MSEC)

Unvoiced Speech

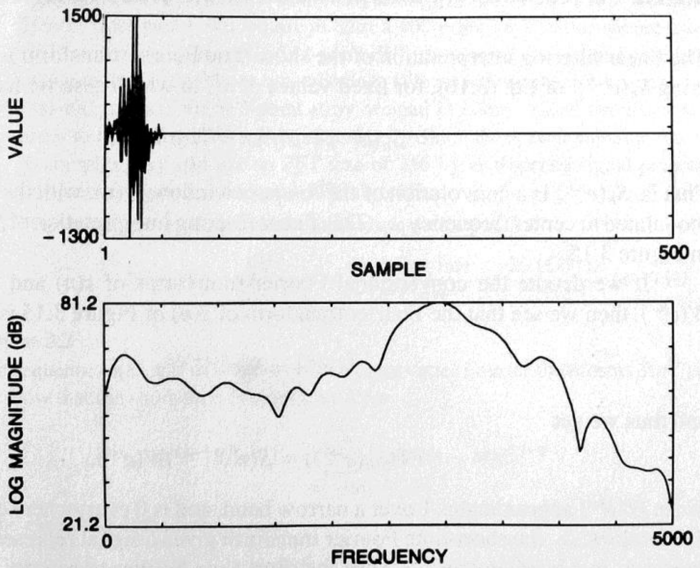


Figure 3.14 Short-time Fourier transform using a short (50 points or 5 msec) Hamming window on a section of unvoiced speech.

From: Rabiner et al.

Short Window

- Poor frequency resolution
- No resolved harmonics
- Good estimate of the overall spectral shape

SHORT TIME FOURIER TRANSFORM LONG HAMMING WINDOW: 500 SAMPLES (=50MSEC)

Unvoiced Speech

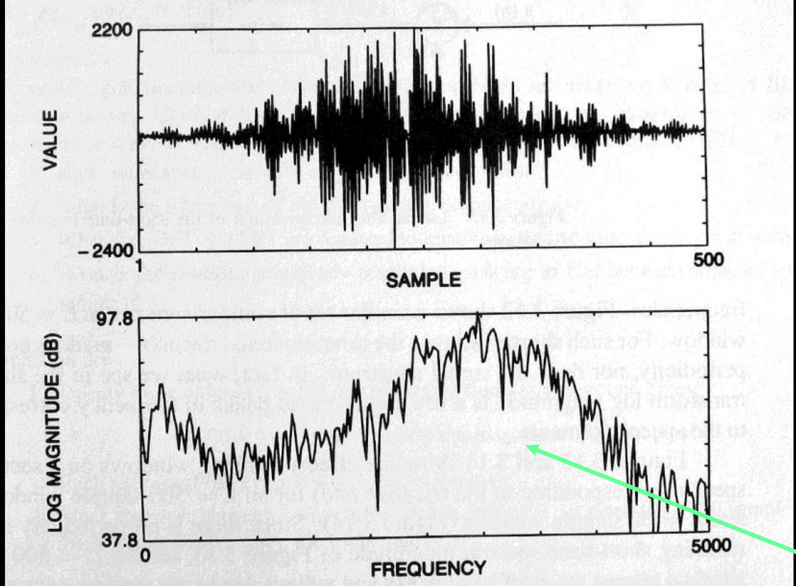


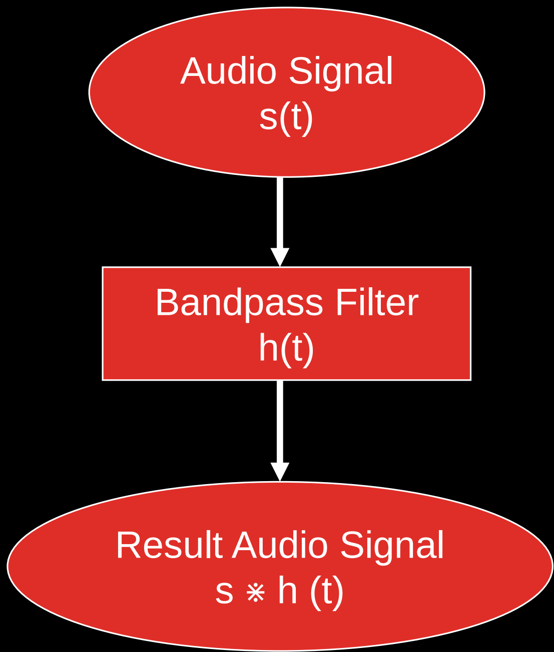
Figure 3.13 Short-time Fourier transform using a long (500 points or 50 msec) Hamming window on a section of unvoiced speech.

From: Rabiner et al.

Long Window

- Good frequency resolution
- Resolved harmonics
- Rough estimate of the overall spectral shape

Higher frequencies



BAND PASS FILTER

Note that the band pass filter can be defined as:

- a *convolution* with a filter response function h(t) in the time domain
- a *multiplication* with a filter response H(f) function in the frequency domain

$$s \ast h\ (t) = \int_{-\infty}^{\infty} s(\tau)h(t-\tau)d\tau \leftrightarrow S(f) \cdot H(f)$$

$$s \ast h\ (t) = \sum_{\tau} s(\tau)h(t-\tau) \leftrightarrow S(f) \cdot H(f)\ \text{(discrete)}$$

BANK OF FILTERS ANALYSIS MODEL

Index	Bark Scale		Mel Scale	
	Center Freq. (Hz)	BW (Hz)	Center Freq. (Hz)	BW (Hz)
1	50	100	100	100
2	150	100	200	100
3	250	100	300	100
4	350	100	400	100
5	450	110	500	100
6	570	120	600	100
7	700	140	700	100
8	840	150	800	100
9	1000	160	900	100
10	1170	190	1000	124
11	1370	210	1149	160
12	1600	240	1320	184
13	1850	280	1516	211
14	2150	320	1741	242
15	2500	380	2000	278
16	2900	450	2297	320
17	3400	550	2639	367
18	4000	700	3031	422
19	4800	900	3482	484
20	5800	1100	4000	556
21	7000	1300	4595	639
22	8500	1800	5278	734
23	10500	2500	6063	843
24	13500	3500	6964	969

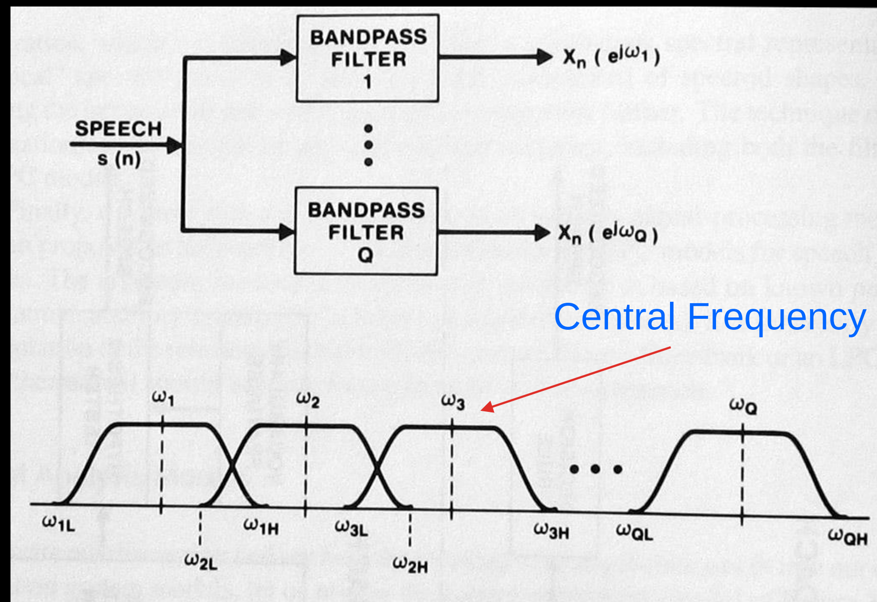


Figure 3.2 Bank-of-filters analysis model.

MEL-CEPSTRUM [4]

Auditory characteristics

- Mel-scaled filter banks

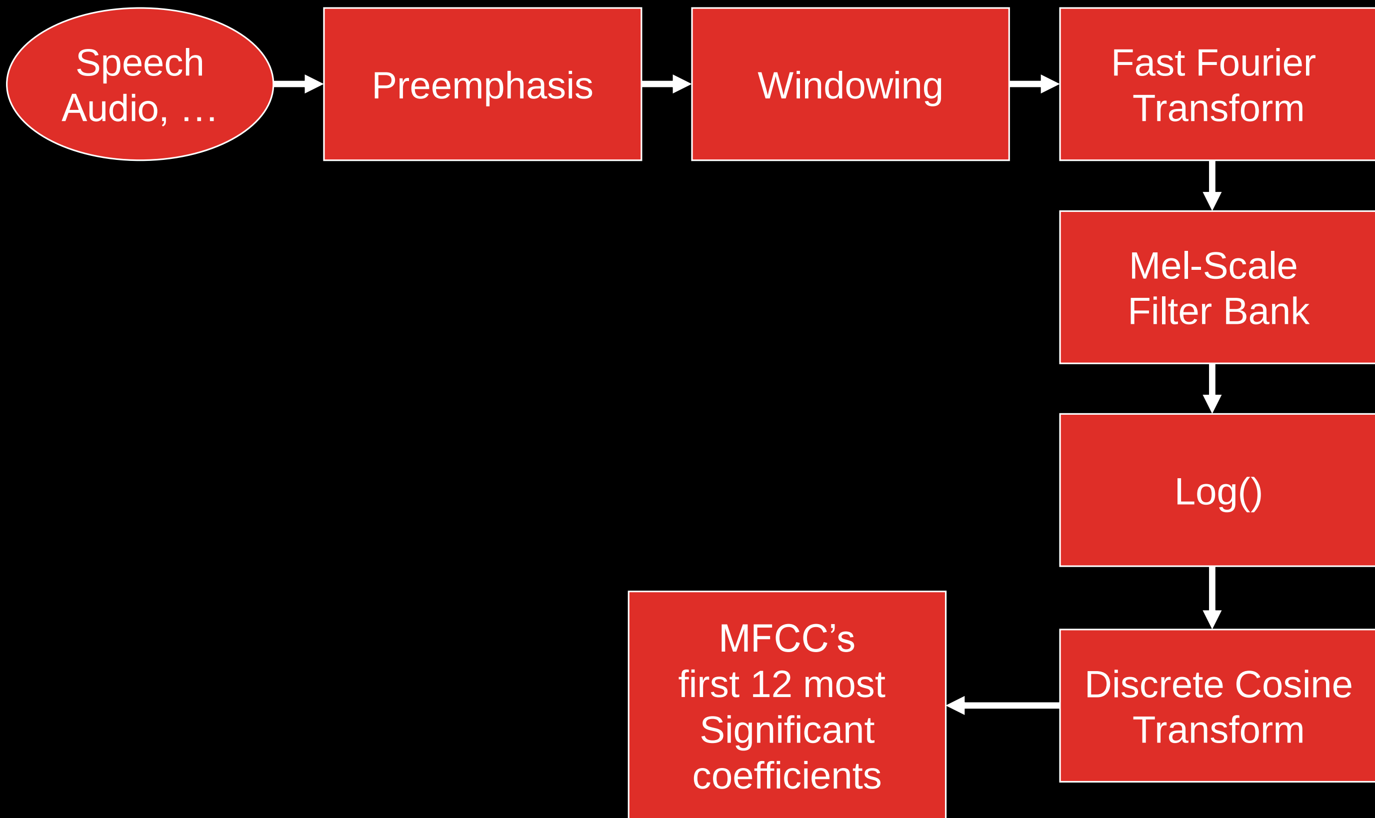
De-correlating properties

- by applying a discrete cosine transform (which is close to a Karhunen-Loeve transform) a de-correlation of the mel-scale filter log-energies results
- => probabilistic modeling on these de-correlated coefficients will be more effective.

One of the most successful features for speech recognition, speaker recognition, and other speech related recognition tasks.

[1, pp 712-717]

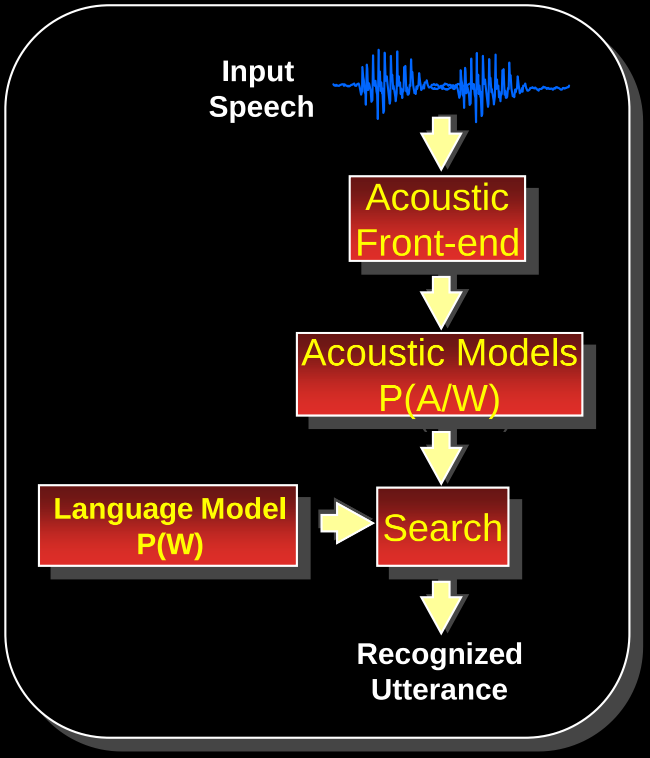
MFCCS



Speech Audio, …
Preemphasis
Windowing
Fast Fourier Transform
Mel-Scale Filter Bank
Log()
Discrete Cosine Transform
MFCC's first 12 most Significant coefficients

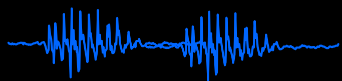
Automatic Speech Recognition Architectures

Incorporating Multiple Knowledge Sources



- The signal is converted to a sequence of feature vectors (spectral and temporal).
- Acoustic models represent sub-word units, such as phonemes: finite-state machine models spectral structure and temporal structure.
- The language model predicts the next set of words, and controls which models are hypothesized. (N-grams)
- Search to find the most probable word sequence.

Acoustic Modeling
Feature Extraction



Input Speech

Fourier Transform

Cepstral Analysis

Perceptual Weighting

Time Derivative

Time Derivative

Energy
+
Mel-Spaced Cepstrum

Delta Energy
+
Delta Cepstrum

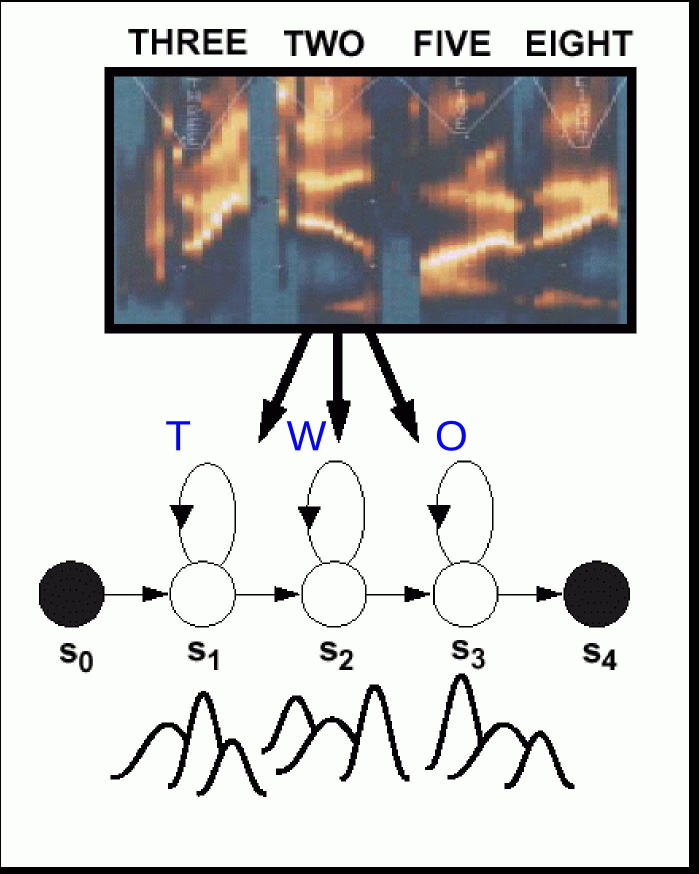
Delta-Delta Energy
+
Delta-Delta Cepstrum

- Incorporate knowledge of the nature of speech sounds in measurement of the features.
- Utilize rudimentary models of human perception.

- Typically: 512 samples (16kHz sampling rate) =>
- Use a ~30 msec window for frequency domain analysis.
- Include absolute energy and 12 spectral measurements.
- Time derivatives to model spectral change.

Acoustic Modeling
Hidden Markov Models

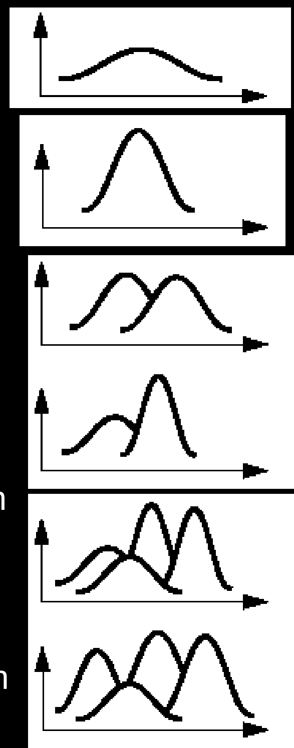
- Acoustic models: temporal evolution of the features (spectrum).
- Gaussian mixture distributions for variations in speaker, accent, and pronunciation.
- Phonetic model topologies are simple left-to-right structures.
- Skip states (time-warping) and multiple paths (alternate pronunciations).
- Sharing model parameters to reduce complexity.



Acoustic Modeling
Parameter Estimation

- Initialization
- Single Gaussian Estimation
- 2-Way Split
- Mixture Distribution Reestimation
- 4-Way Split
- Reestimation

•••



- Word level transcription
- Supervises a closed-loop data-driven modeling
- Initial parameter estimation
- The expectation/maximization (EM) algorithm is used to improve our parameter estimates.
- Computationally efficient training algorithms (Forward-Backward) are crucial.
- Batch mode parameter updates are typically preferred.
- Decision trees and the use of additional linguistic knowledge are used to optimize parameter-sharing, and system complexity,.

MACHINE LEARNING METHODS

- k Nearest Neighbors
- Decision Trees
- Random Forests (weighted neighborhoods scheme)
- Gradient Boosting Machines (e.g. boosting of prediction model ensembles)
- Vector Quantization
  - Finite code book of spectral shapes
  - The code book codes for ‘typical’ spectral shape
  - Method for all spectral representations (e.g. Filter Banks, LPC, ZCR, etc. …)
- Support Vector Machines
- Markov Models
- Hidden Markov Models
- Neural Networks Etc.

VECTOR QUANTIZATION

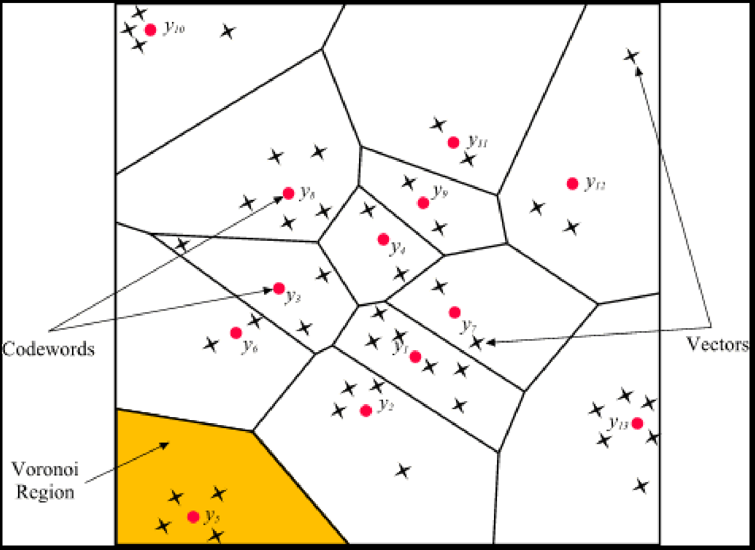
- Data represented as feature vectors.
- Vector Quantization (VQ) Training set => determine a set of code words that constitute a code book.
- Code words are centroids using a similarity or distance measure d.
- Code words together with measure d divide the space into Voronoi regions.
- A query vector falls into a Voronoi region and will be represented by the respective code word.

[2, pp. 466 – 467]

VECTOR QUANTIZATION

Distance measures d(x,y):

- Euclidean distance
- Taxi cab distance
- Hamming distance
- etc.



VECTOR QUANTIZATION

Let a training set of L vectors be given for a certain class of objects.
Assume a codebook of M code words is wanted for this class.

Initialize:

- choose M arbitrary vectors of the L vectors of the training set.
- This is the initial code book.

Nearest Neighbor Search:

- for each training vector v, find the code word w in the current code book that is closest and assign v to the corresponding cell of w.

Centroid Update:

- For each cell with code word w determine the centroid c of the training vectors that are assigned to the cell of w.
- Update the code word w with the new vector c.

Iteration:

- repeat the steps **Nearest Neighbor Search** and **Centroid Update** until the average distance between the new and previous code words falls below a preset threshold.

VQ FOR CLASSIFICATION

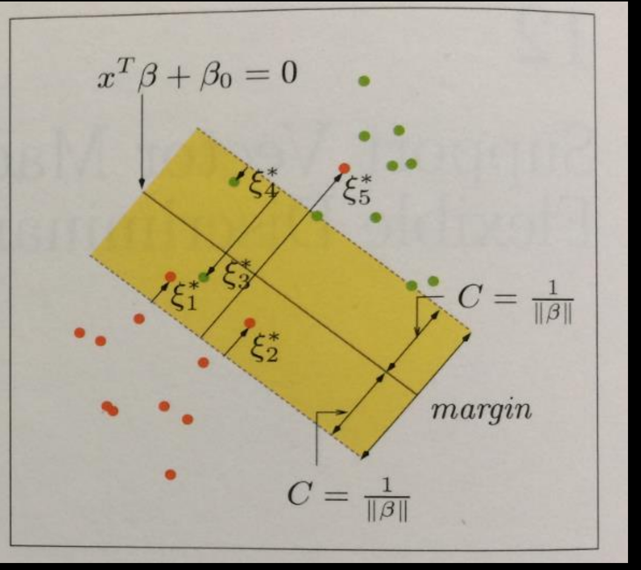
A code book $CB_k = \{y^k_i \mid 1 \leq i \leq M\}$, can be used to define a class C_k.

Example Audio Classification:

- Classes 'crowd', 'car', 'silence', 'scream', 'explosion', etc.
- Determine by using VQ code books CB_k for each of the respective classes C_k.
- VQ is very often used as a baseline method for classification problems.

SUPPORT VECTOR MACHINES

- A generalization of linear decision boundaries for classification.
- Necessary when classes overlap when using linear decision boundaries (non separable classes).



From: [2]

Find hyper plane P: $x^T\beta + \beta_0 = 0$, such that

$\boxed{\|\beta\| \text{ is minimized}}$ over $\begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \varepsilon_i \ \forall i \\ \varepsilon_i \geq 0, \quad \sum \varepsilon_i \leq constant \end{cases}$

$\boxed{\Rightarrow \text{Margin } C = \frac{1}{\|\beta\|} \text{ is maximized.}}$

Where $(x_1,y_1), \ldots (x_N,y_N)$ are our training pairs, with $x_i \in \mathbb{R}^p$ and $y_i \in \{-1,1\}$,

$\varepsilon = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_N)$ are the slack variables, i.e.,

ε_i = the amount that x_i is on the wrong side of the margin $C = \frac{1}{\|\beta\|}$ from the hyper plane P.

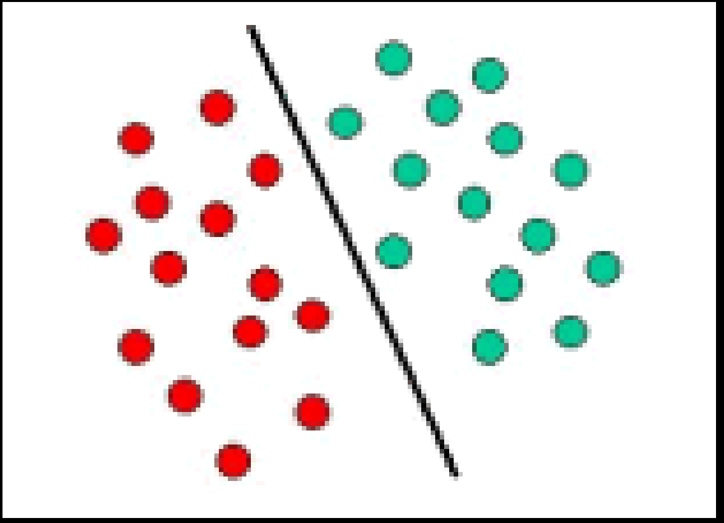
i.e. C is maximized.

=> Problem is quadratic with linear inequalities constraint. [2, pp 377-389]

SUPPORT VECTOR MACHINE (SVM)

In this method so called support vectors define decision boundaries for classification and regression.

An example where a straight line separates the two Classes: a linear classifier



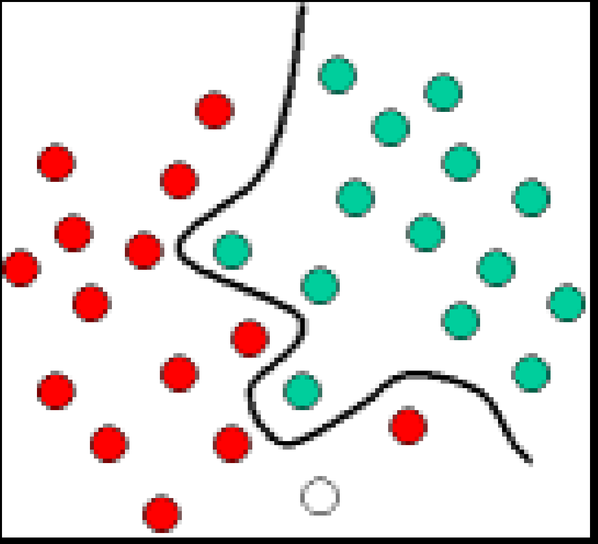
Images from: www.statsoft.com.

SUPPORT VECTOR MACHINE (SVM)

In general classification is not that simple.

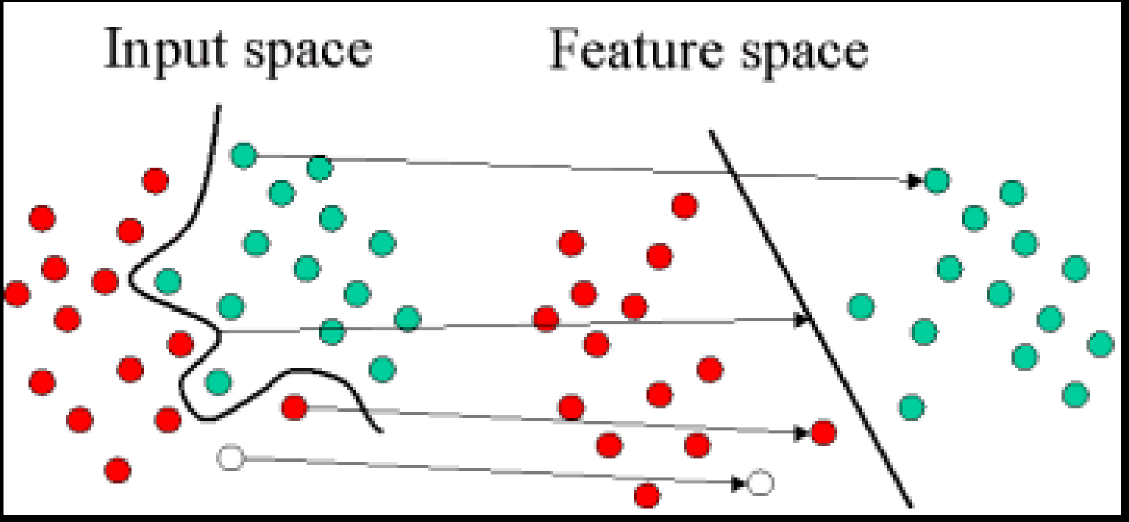
SVM is a method that can handle the more complex cases where the decision boundary requires a curve.

SVM uses a set of mapping functions (kernels) to map the feature space into a transformed space so that hyperplanes can be used for the classification.



SUPPORT VECTOR MACHINE (SVM)

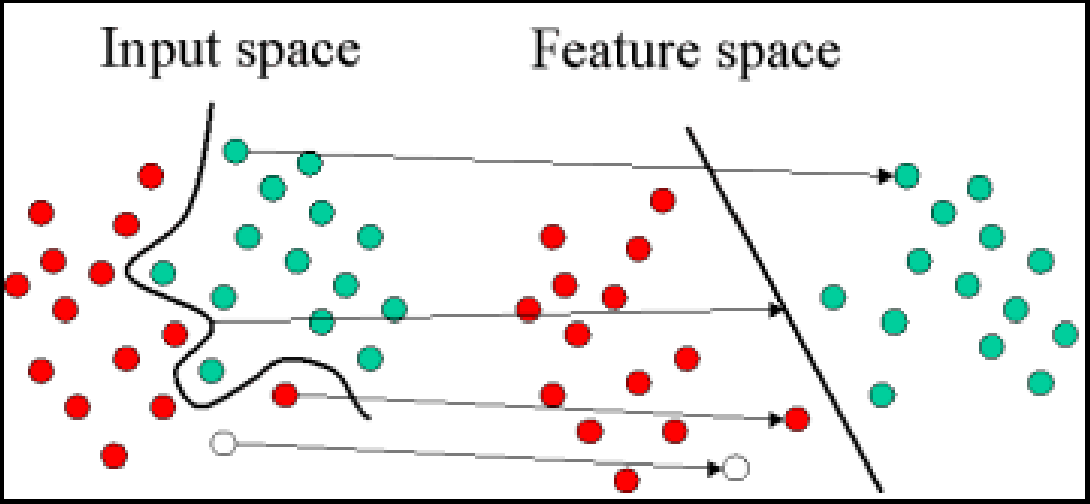
SVM uses a set of mapping functions (kernels) to map the feature space into a transformed space so that hyperplanes can be used for the classification.



SUPPORT VECTOR MACHINE (SVM)

Training of an SVM is an iterative process:

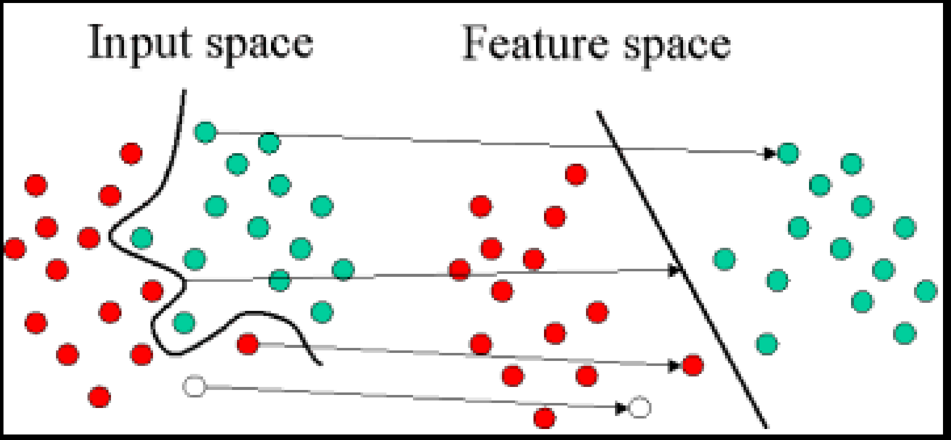
- optimize the mapping function while minimizing an error function
- The error function should capture the penalties for misclassified, i.e., non separable data points.



SUPPORT VECTOR MACHINE (SVM)

SVM uses kernels that define the mapping function used in the method. Kernels can be:

- Linear
- Polynomial
- RBF
- Sigmoid
- Etc.



- RBF (radial basis function) is the most popular kernel, again with different possible base functions.
- NB The final choice depends on characteristics of the classification task.

AUDIO CLASSIFICATION USING NEURAL NETWORKS

An example by Rishi Sidhu:

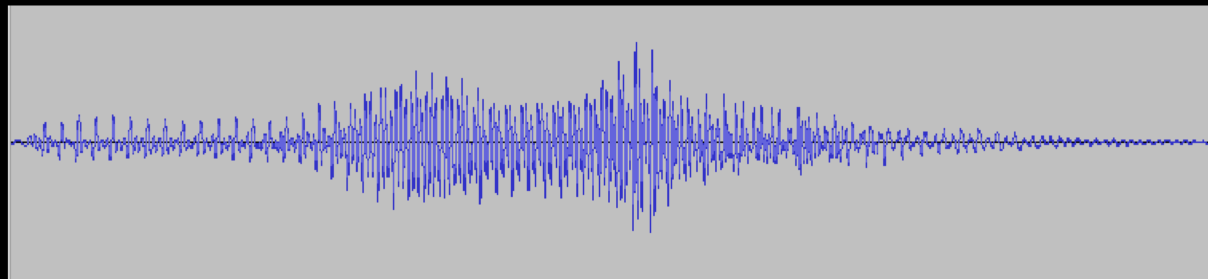
https://medium.com/x8-the-ai-community/audio-classification-using-cnn-coding-example-f9cbd272269e

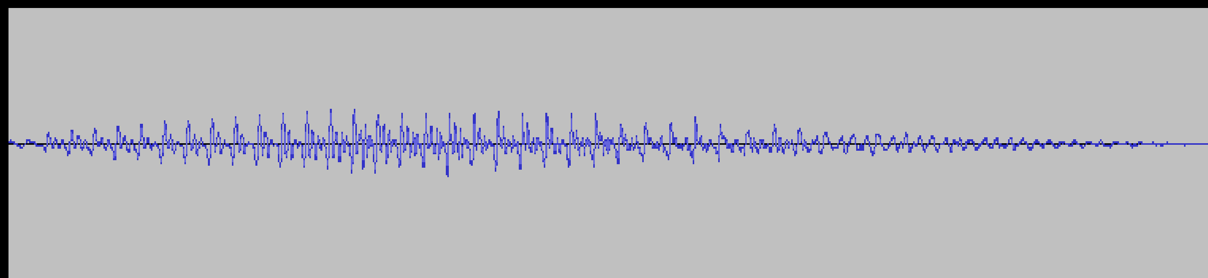
Using data from the **Spoken Digit Dataset** by Zohar Jackson:

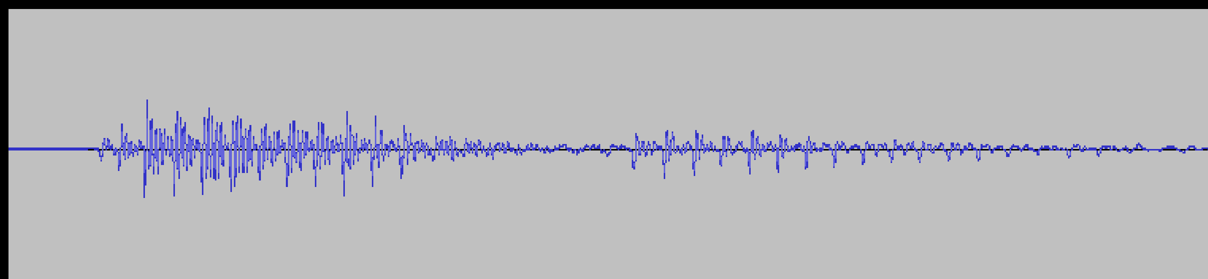
Https://github.com/Jakobovski/free-spoken-digit-dataset

Using Convolutional Neural Networks on Spectrograms.

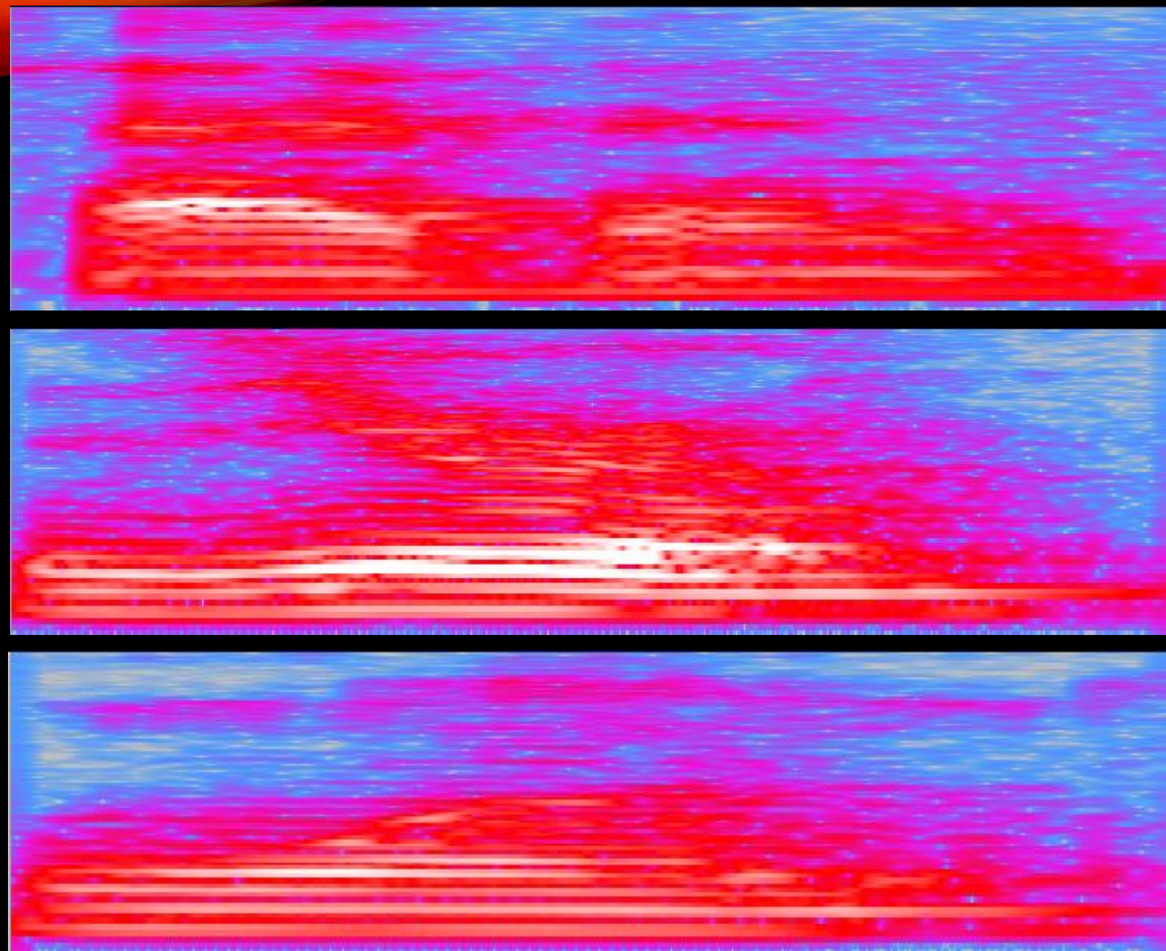
DIGITS



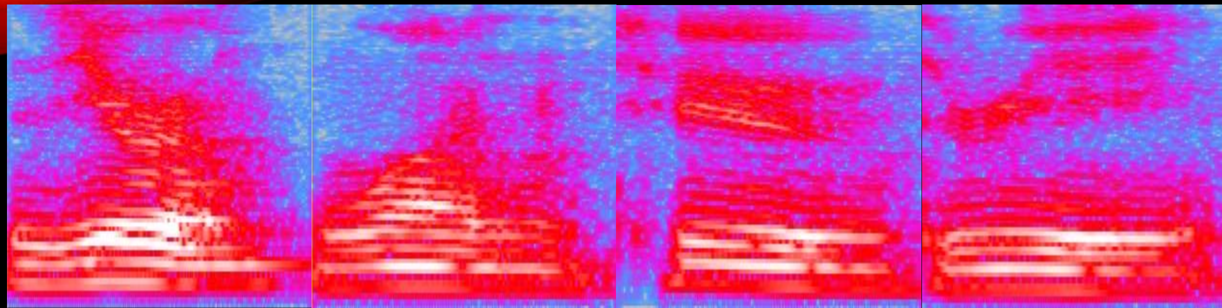


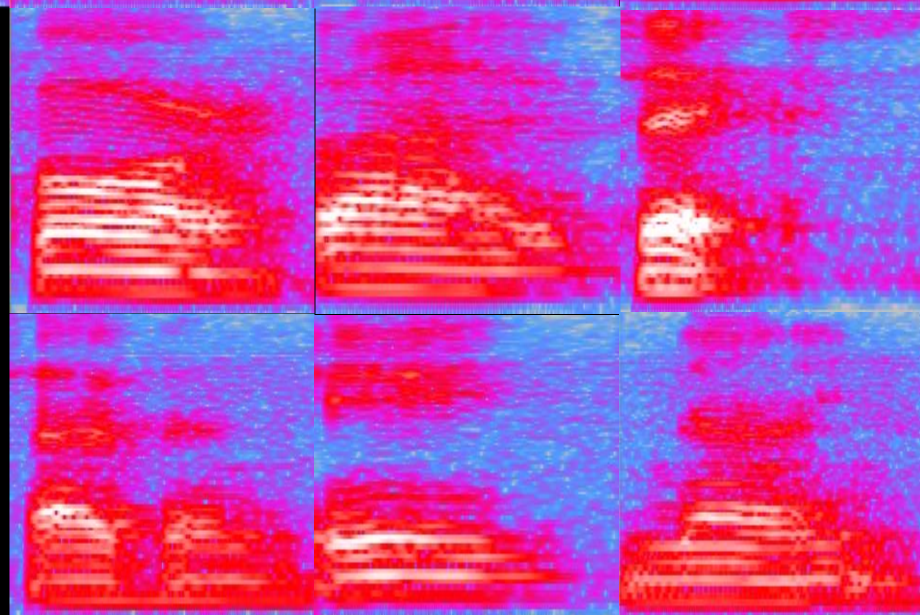


API

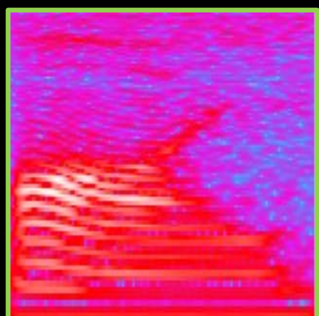


API

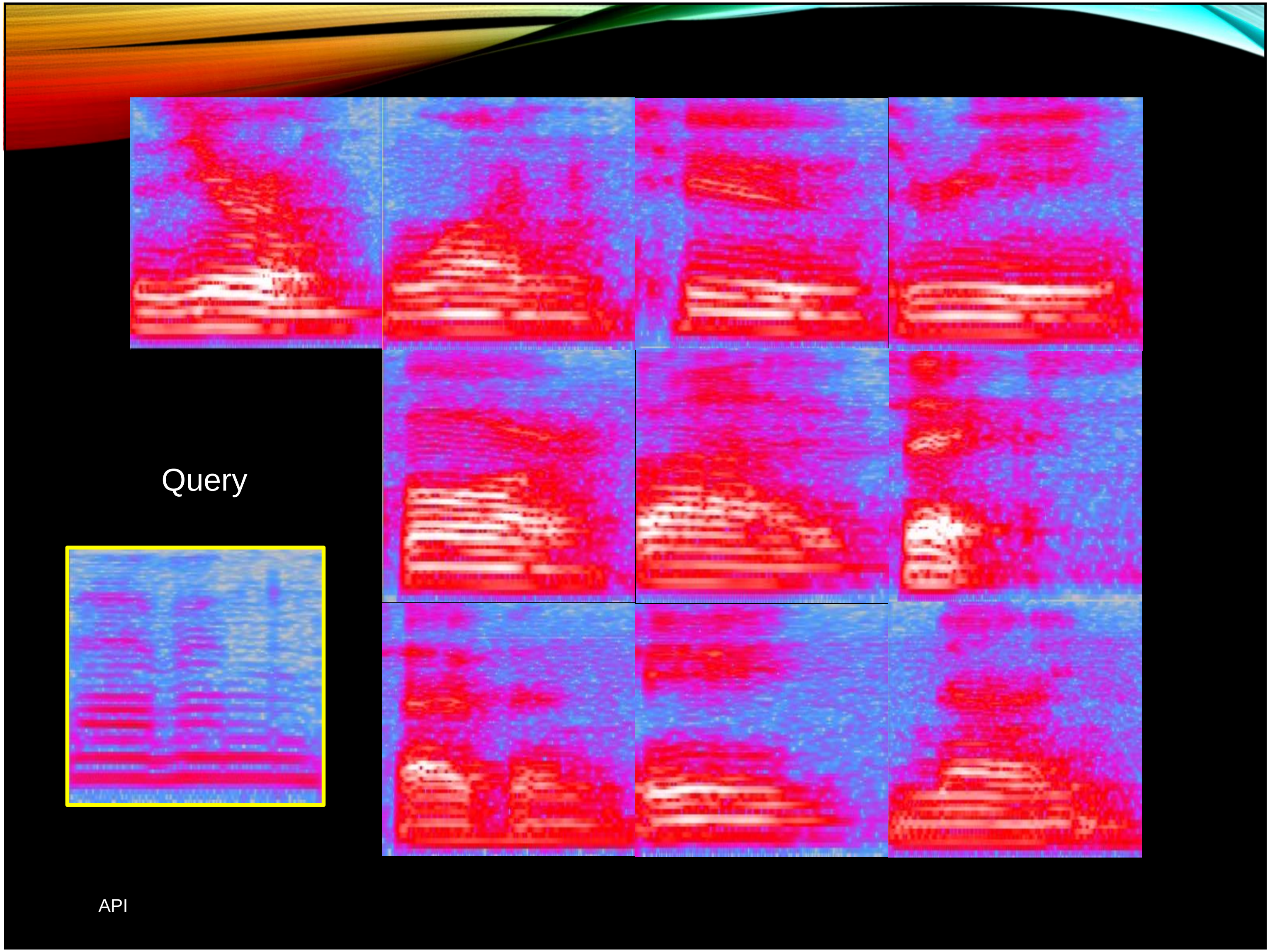




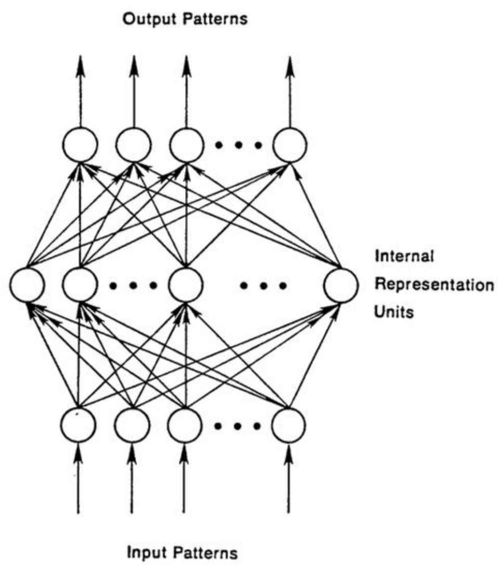
Query



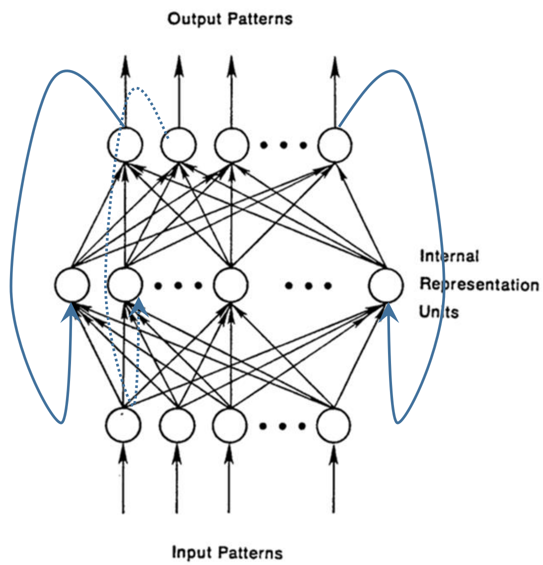
API


Query
API

Some Neural Networks



Feed Forward Neural Network



Recurrent Neural Network

DNN: AlexNet, VGG16, ResNet, etc.

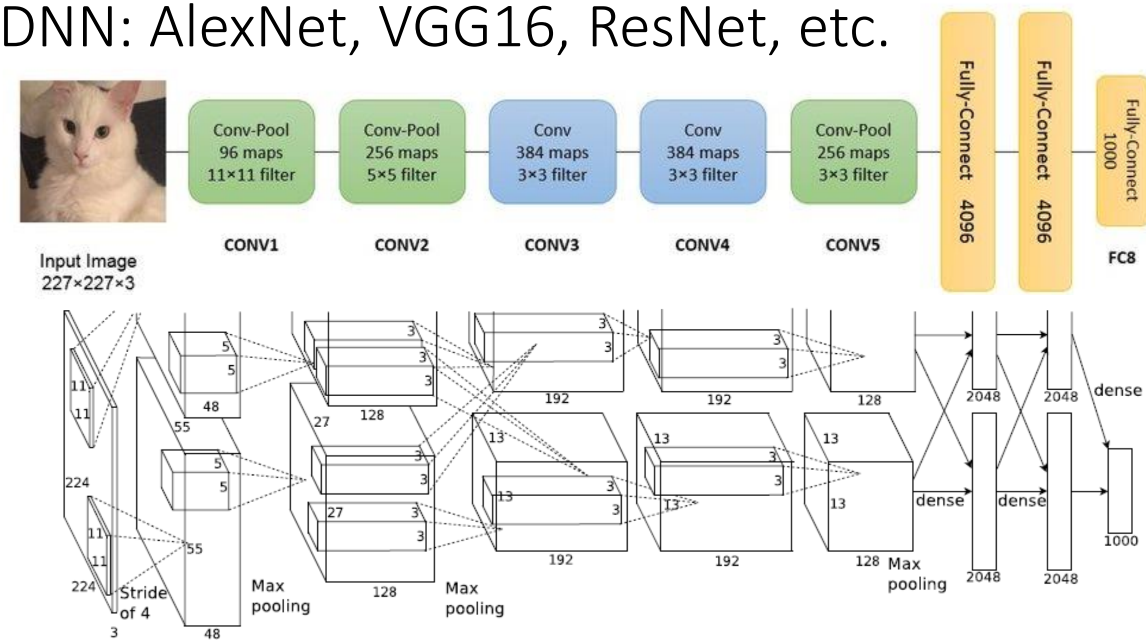


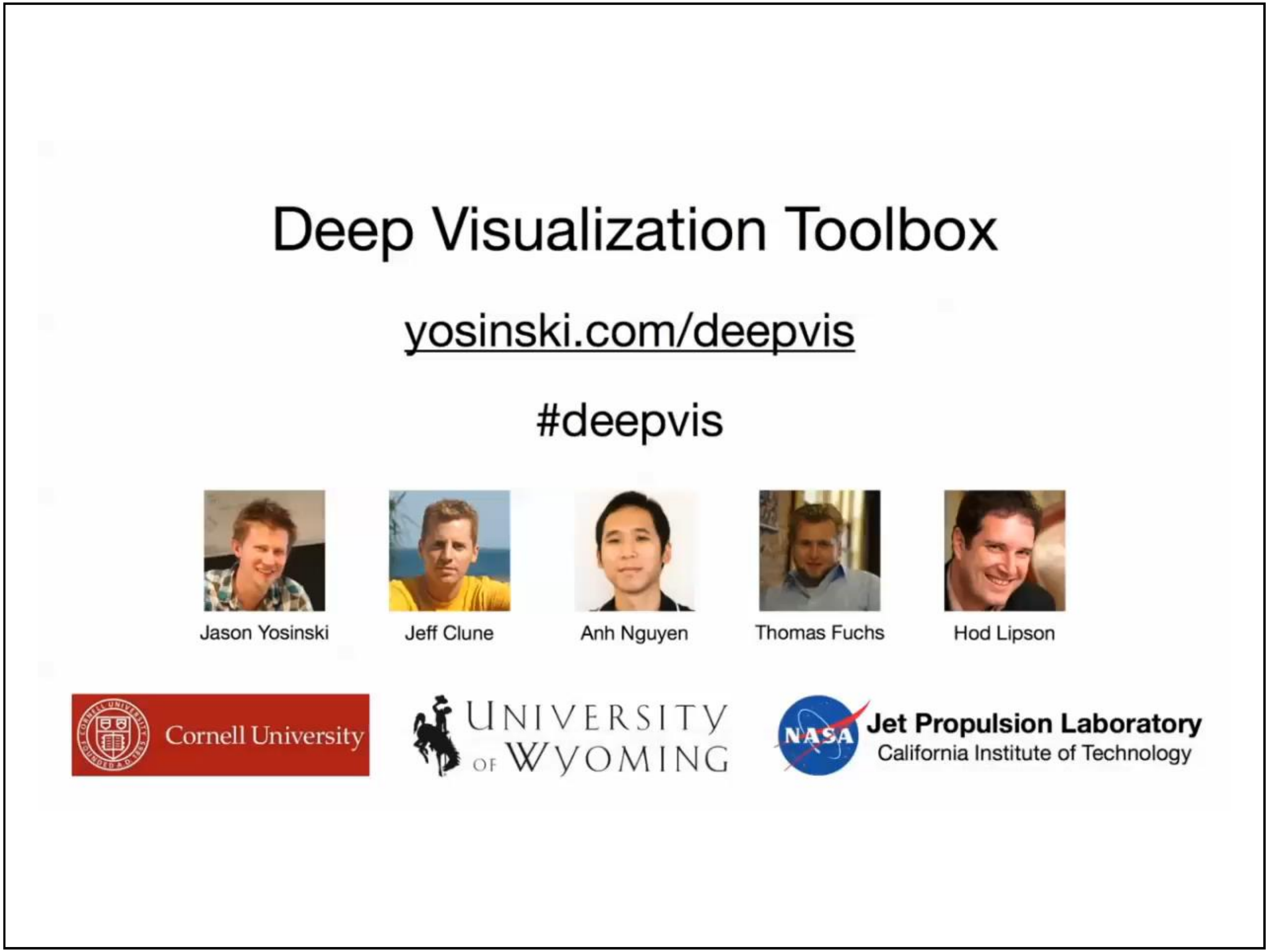
Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffrey E. "ImageNet classification with deep convolutional neural networks" Communications of the ACM. 60 (6): 84–90.

ImageNet

- AlexNet (~2011; 2015 58.9 %)
- VGG-16 (2015, 74.4%)
- ResNet-152 (2015, 78.57%)
- EfficientNetV2B0 (2021, 83.9%)

https://paperswithcode.com/sota/image-classification-on-imagenet


Deep Visualization Toolbox
yosinski.com/deepvis
#deepvis
Jason Yosinski
Jeff Clune
Anh Nguyen
Thomas Fuchs
Hod Lipson
Cornell University
University of Wyoming
NASA
Jet Propulsion Laboratory
California Institute of Technology

Cats and Dogs

Kaggle Dataset (https://www.kaggle.com/c/dogs-vs-cats/data)

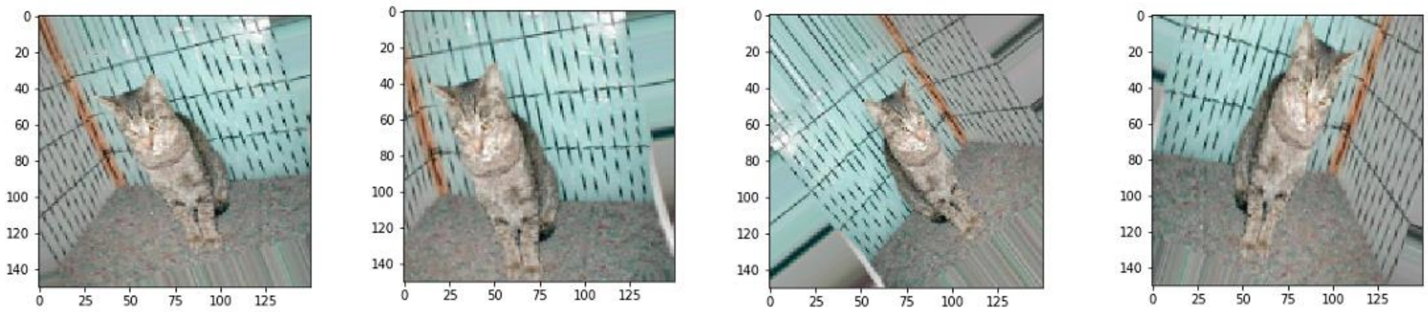
- 2000 images of cats
- 2000 images of dogs

- Given an image: is it a cat or a dog?

Divide into:

- Training set (2000 images)
- Validation set (1000 images)
- Test set (1000 images)

Cats and Dogs

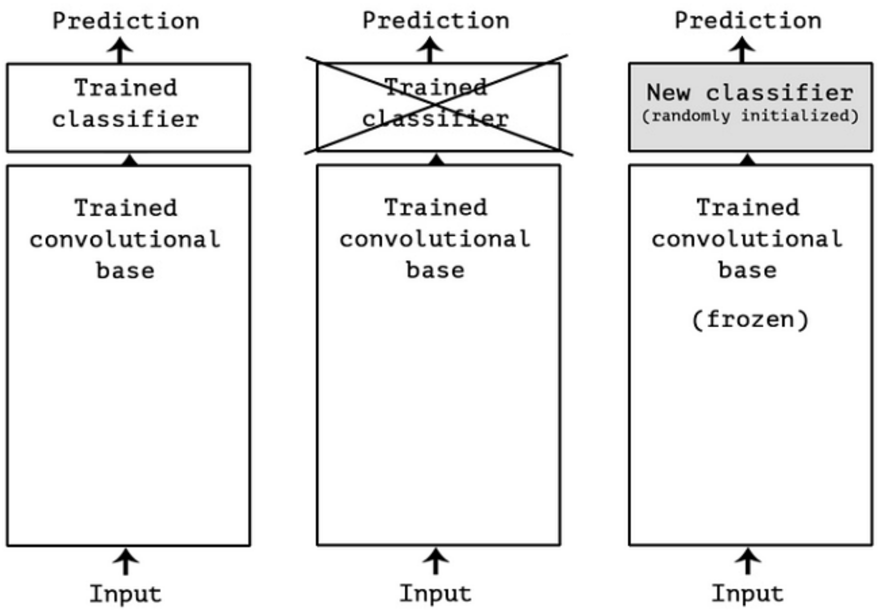


Convolutional Neural Network

- Without any regularization: ~71% accuracy
- With data augmentation: ~82% accuracy
- Feature extraction using a pre-trained NN: ~90% accuracy
- Fine tuning a pre-trained NN: ~95% accuracy

These are examples of Deep Learning with Small Datasets.

Cats and Dogs



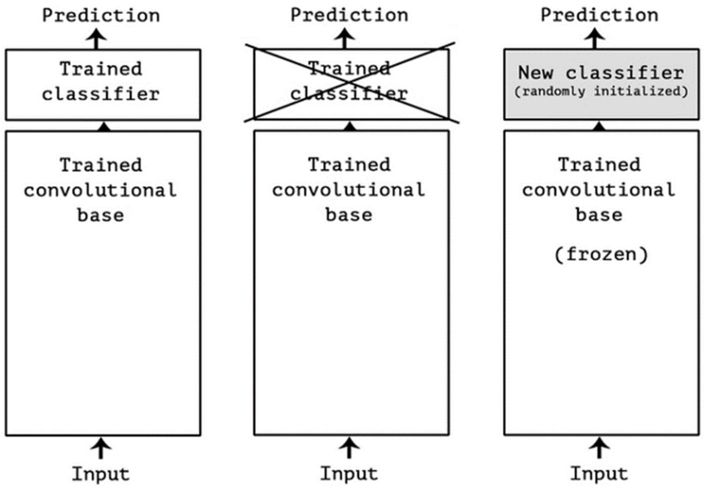
VGG16 (pre packed with Keras)

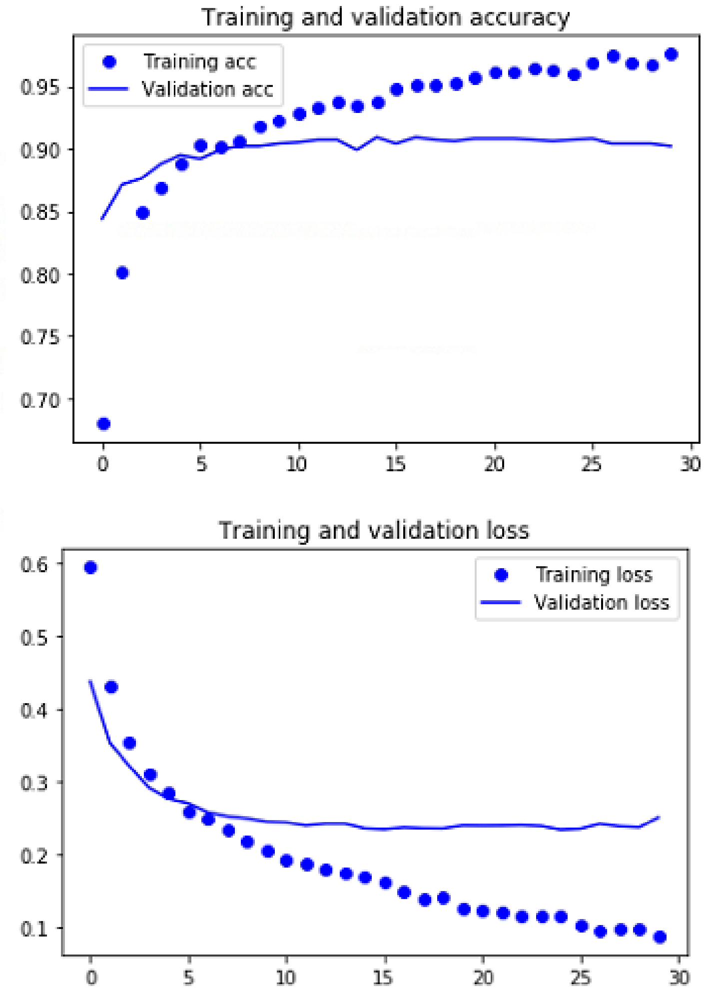
Convolutional Neural Network

- Without any regularization: ~71% accuracy
- With data augmentation: ~82% accuracy
- Feature extraction using a pre-trained NN: ~90% accuracy
- Fine tuning a pre-trained NN: ~95% accuracy

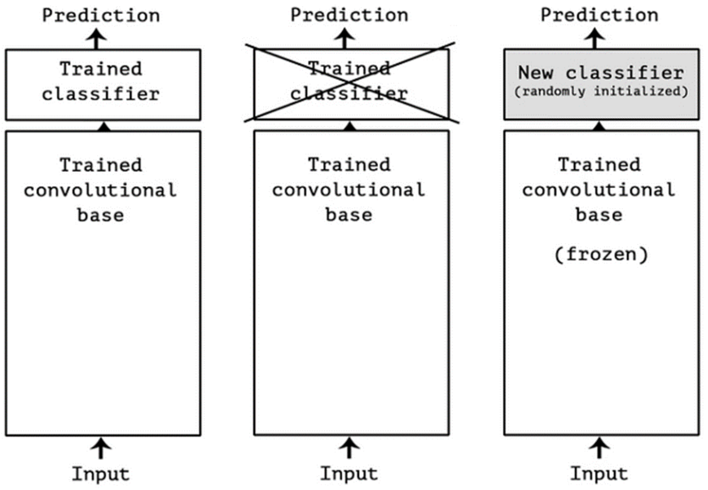
These are examples of Deep Learning with Small Datasets.

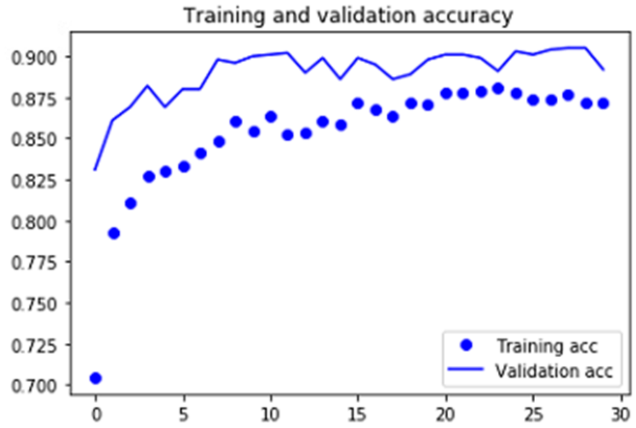
VGG16
Feature Extraction

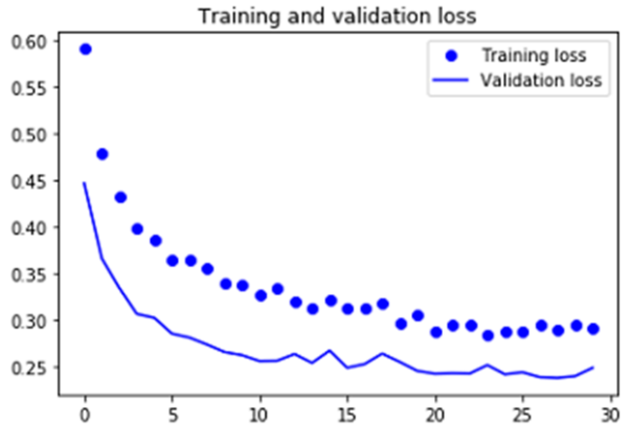




VGG16
Feature Extraction + Data Augmentation

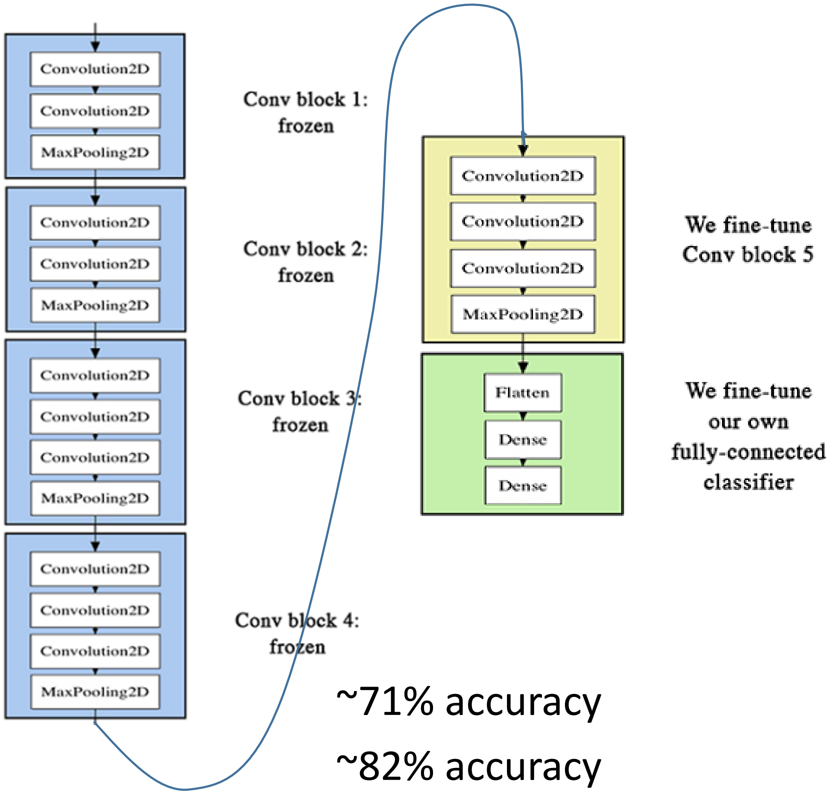






Cats and Dogs

VGG16 (pre packed with Keras)

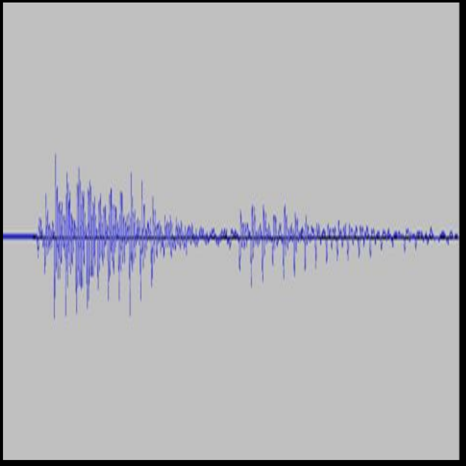


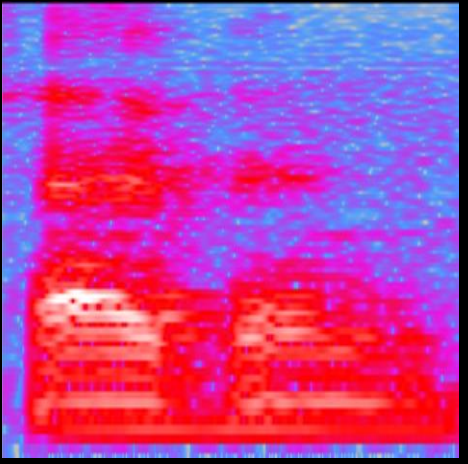
Convolutional Neural Network

- Without any regularization: ~71% accuracy
- With data augmentation: ~82% accuracy
- Feature extraction using a pre-trained NN: ~90% accuracy
- Fine tuning a pre-trained NN: ~95% accuracy

These are examples of Deep Learning with Small Datasets.

CNN'S FOR AUDIO CLASSIFICATION





- Both images can be used to recognize the spoken digit.
- The spectrogram yields better accuracy for the tests.
- How would you perform data augmentation?

API

CNN ARCHITECTURE

Input Layer

Convolutional layer with kernel size 3x3

Convolutional layer with kernel size 3x3

Max Pooling layer with pool size 2x2

Dropout layer

Flattening layer

Dense layer 1

Dense layer 1

API

CNN DEFINED IN TF.KERAS

```
#Define Model
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=input_shape))
model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
#Compile
model.compile(loss=keras.losses.categorical_crossentropy,
optimizer=keras.optimizers.adam(), metrics=['accuracy'])
print(model.summary())
#Train and Test The Model
model.fit(x_train, y_train, batch_size=4, epochs=10, verbose=1, validation_data=(x_test,
y_test))
```

API

TRAINING, TEST AND VALIDATION DATASETS

Training Data

- 1800 Images of Spectrograms: 34x50 pixels
- Each image is labeled with the correct digit

Validation Data

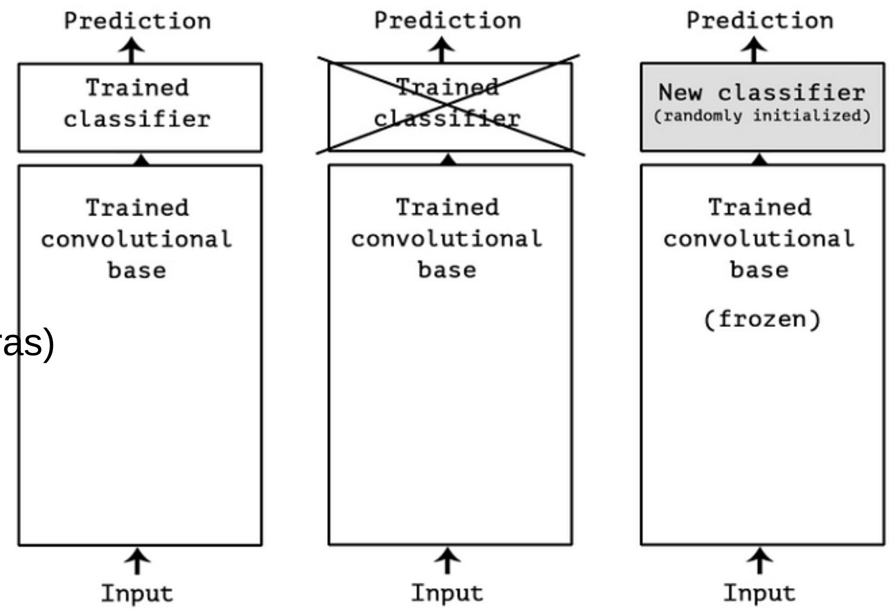
- 200 Images of Spectrograms: 34x50 pixels
- Each image is labeled with the correct digit
- Exclusive speaker(s)

Test Data

- 200 Images of Spectrograms: 34x50 pixels
- Each image is labeled with the correct digit
- Exclusive speaker(s)

Digits

VGG16 (pre packed with TF Keras)



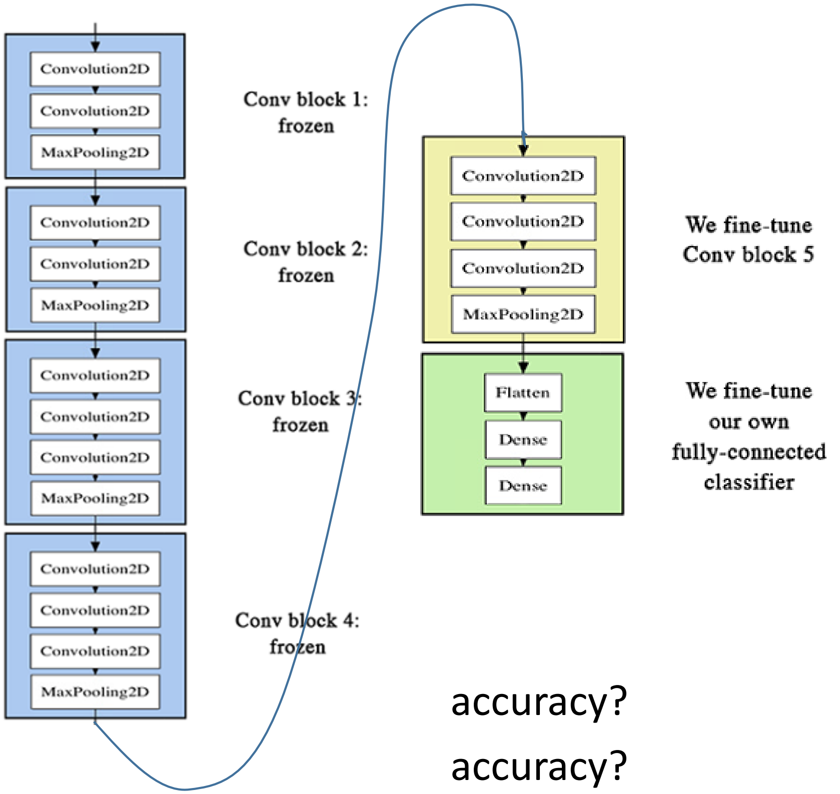
Convolutional Neural Network

- Without any regularization: accuracy ?
- With data augmentation: accuracy ?
- Feature extraction using a pre-trained NN: accuracy ?
- Fine tuning a pre-trained NN: accuracy ?

These are examples of Deep Learning with Small Datasets.

Digits

VGG16 (pre packed with TF Keras)

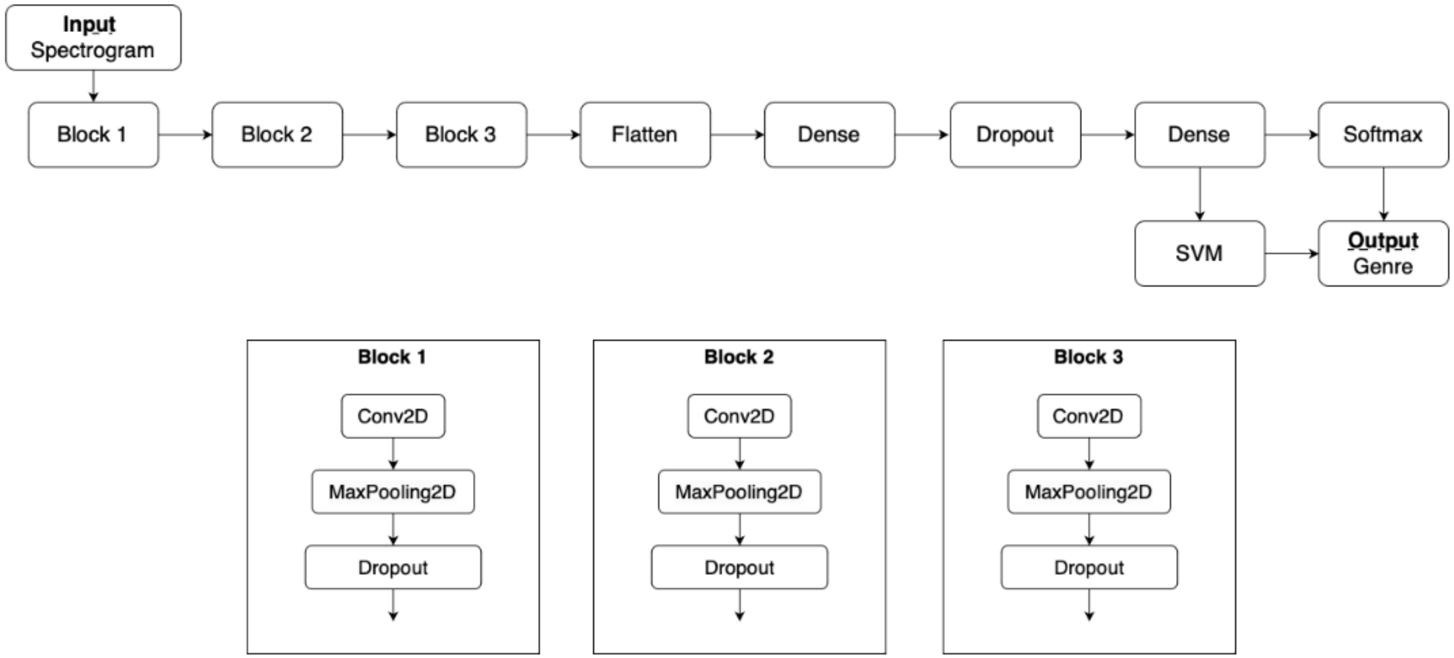


Convolutional Neural Network

- Without any regularization: accuracy?
- With data augmentation: accuracy?
- Feature extraction using a pre-trained NN: accuracy?
- Fine tuning a pre-trained NN: accuracy?

These are examples of Deep Learning with Small Datasets.

Genre Classification: MusicRecNet (Elbir et al., 2020)



Visualization of the MusicRecNet architecture. Output genres are either defined by using softmax probability scores or the SVM classifier.

Genre Classification Benchmarks
GTZAN and FMA

Dataset	GTZAN	FMA_8	FMA_14	FMA medium
Number of songs per genre	100	1000	100	21-7103
Total number of songs	1000	8000	1400	25000

Model	*GTZAN Accuracy*
Zhang et al. [11]	87.4%
Liu et al. [12]	93.9%
Elbir, A & Aydin, N. [1]	81.8%
Elbir, A & Aydin, N. with SVM [1]	**97.6%**
Our Baseline Implementation	81.0%
Our Baseline Implementation + SVM	81.6%

Genre Classification Benchmarks GTZAN and FMA

Dataset	GTZAN	FMA_8	FMA_14	FMA medium
Number of songs per genre	100	1000	100	21-7103
Total number of songs	1000	8000	1400	25000

Dataset	GTZAN	GTZAN 224	FMA_8	FMA_8 224	FMA_14
Method					
Baseline	81.0		68.6		42.0
Baseline-SVM-Output	81.5		70.7		42.0
Baseline-SVM-D128	81.6		72.1		42.6
VGG	73.1		53.4		53.6
VGG-SVM-Output	73.0		53.9		53.6
VGG-SVM-D128	76.5		54.4		54.8
VGG-FT	81.6		60.7		57.3
VGG-SVM-Output-FT	81.6		61.0		57.3
VGG-SVM-D128-FT	83.0		61.2		56.9
EfficientNet	80.0	82.1	59.6	62.0	56.8
EfficientNet-SVM-Output	80.6	82.5	60.5	63.0	56.5
EfficientNet-SVM-D128	83.0	87.5	61.4	63.1	60.8
EfficientNet-FT	90.0	90.5	76.9	73.8	60.4
EfficientNet-SVM-Output-FT	89.8	90.5	76.8	73.7	60.4
EfficientNet-SVM-D128-FT	**90.3**	**90.8**	**77.4**	**73.9**	**61.1**

C. Wu et al. Transformer-based Acoustic Modeling for Streaming Speech Synthesis, INTERSPEECH 2021

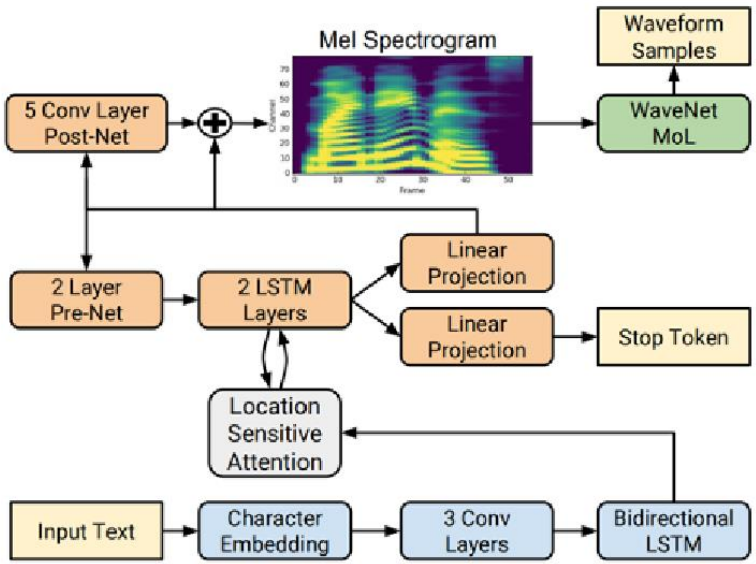
https://transformer-tts-accoustic-model.github.io/samples/

Tacotron2 uses Bi-directional Long Short-term Memory (BLSTM) recurrent networks.

- cannot effectively model long-term dependencies
- a poor quality on long speech.

FastSpeech state-of-the-art

- in modeling speech prosody and spectral features, but
- computation is parallel over the full utterance context.



Tacotron2

C. Wu et al. Transformer-based Acoustic Modeling for Streaming Speech Synthesis, INTERSPEECH 2021

TTS systems usually consist of two stages:

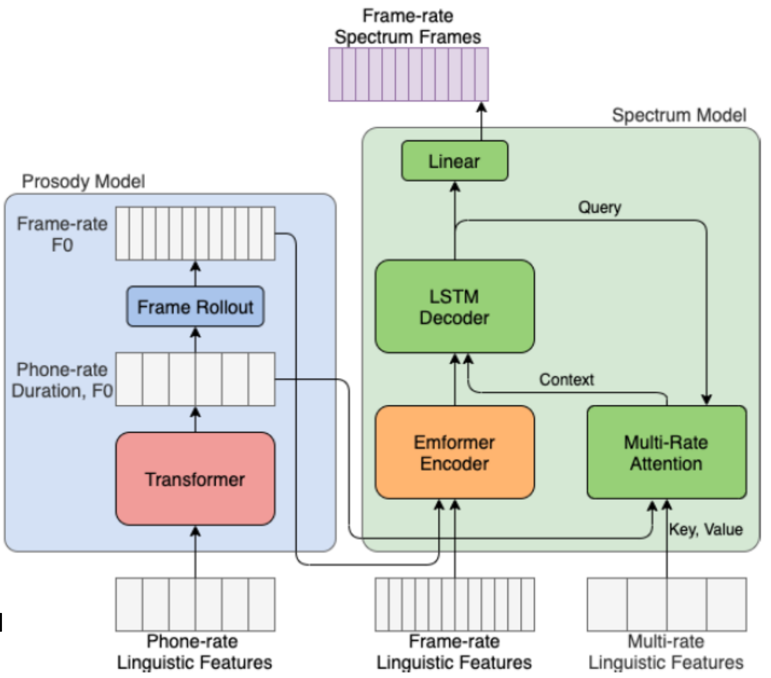
- acoustic model that predicts the prosody and spectral features
- followed by a neural vocoder that generates the audio
- waveform.

Tranformer models:

- model long-term dependencies
- Complexity grows quadratically

This work

- Effcient constant speed implementation: for streaming speech synthesis
- uses a transformer network that predicts the prosody features at phone rate
- an Emformer network to predict the frame-rate spectral features (streaming)
- WaveRNN Vocoder used



https://transformer-tts-accoustic-model.github.io/samples/

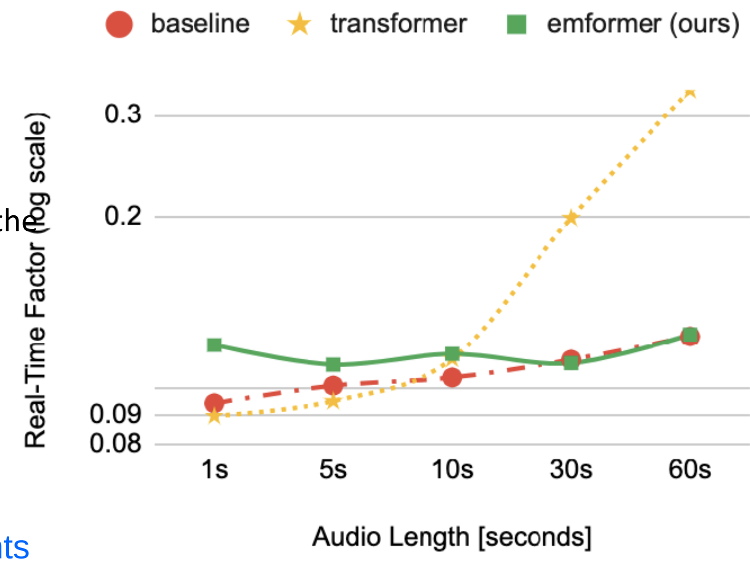
C. Wu et al. Transformer-based Acoustic Modeling for Streaming Speech Synthesis, INTERSPEECH 2021

TTS systems usually consist of two stages:

- acoustic model that predicts the prosody and spectral features
- followed by a neural vocoder that generates the audio
- waveform.

Tranformer models:

- model long-term dependencies
- Complexity grows quadratically



Mean Opinion Scores (1-5) from 400 participants

System	Prosody	Spectrum	Normal	Long
Groundtruth	–	–	4.307 ± 0.037	4.360 ± 0.044
Baseline [11]	BLSTM with self-attention [26]	Multi-rate attention [11]	4.173 ± 0.042	4.019 ± 0.055
Ours-1	Transformer	Multi-rate attention	4.174 ± 0.042	4.107 ± 0.052
Ours-2	BLSTM with self-attention	Emformer with multi-rate attention	4.192 ± 0.041	4.034 ± 0.053
Ours-3 (best)	Transformer	Emformer with multi-rate attention	**4.213 ± 0.042**	**4.201 ± 0.048**

https://transformer-tts-accoustic-model.github.io/samples/

J. Li, **Recent Advances in End-to-End Automatic Speech Recognition.**
APSIPA TranS. on Sig. & Inf. Processing, 2022.

- Hybrid ASR Systems
  - traditional architecture with DNN’s replacing Gaussian modelling.
- End-to-End (E2E) ASR System
  - One single network from input speech to a token sequence
  - uses one single objective function for optimizing the whole model
  - More simple ASR Pipeline
  - More compact models
- E2e Achieve state-of-the-art results on most benchmarks, but:
  - Hybrid models still used in large portion of commercial ASR Systems
  - Practical factors:
    - Streaming
    - Latency
    - Speaker and Language domain adaption (current main research focus)
    - Etc.
  - These challenges are being addressed in current E2E ASR systems research

End-to-End ASR Architectures

- Connectionist Temporal Classification
- Attention Based Encoder-Decoder (TRANSFORMERS)
- Recurrent Neural Network Transducer (RNN-T)
  - Streaming, High accuracy, low latency
  - Good candidate for industrial applications

REFERENCES

1. T.F. Quatieri, Discrete-Time Speech Signal Processing, Principles and Practice, Prentice-Hall, Inc. 2002.
2. T. Hastc, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Data Mining, Inference, and Prediction, Springer, 2001.
3. W.H. Press, S.A.Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipies in C++, The Art of Scientific Computing, 2nd Edition, Cambridge University Press, 2002.
4. S.B. Davies, P. Mermelstein, Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences, IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-28, no.4, pp. 357-366, Aug. 1980.

REFERENCES

5. P. Kenny, "Joint Factor Analysis of Speaker and Session Variability: Theory and Algorithms, Tech. Report CRIM-06/08-13," 2005.

Available: http://www.crim.ca/perso/patrick.kenny

6. N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Frontend factor analysis for speaker verification," IEEE Trans. Audio, Speech, Lang. Process., vol. 19, no. 4, pp. 788–798, May 2011.

7. François Chollet, Deep Learning with Python, Manning Publications, November 2017.

Name	Family	Session	Title
Amber	Zitman	1	F. Lieb et al. Audio inpainting - Evaluation of time-frequency representations and structured sparsity approaches. Signal Processing, 2018.
Anca	Matei	1	H. Rump et al. AUTOREGRESSIVE MFCC MODELS FOR GENRE CLASSIFICATION IMPROVED BY HARMONIC-PERCUSSION SEPARATION. ISMIR 2010.
Andrei	Mestani	1	S. Rau et al. VISUALIZATION FOR AI-ASSISTED COMPOSING, ISMIR 2022.
Anna	Perry	1	F. Nadeem, LEARNING FROM MUSICAL FEEDBACK WITH SONIC THE HEDGEHOG. SMIR2021.
Anthonie	Schaap	1	F. Foscarin et al. CONCEPT-BASED TECHNIQUES FOR "MUSICOLOGIST-FRIENDLY" EXPLANATIONS IN A DEEP MUSIC CLASSIFIER. ISMIR 2022
Bingxin	Wang	1	M. Won et al. SEMI-SUPERVISED MUSIC TAGGING TRANSFORMER. ISMIR 2021.
Carla	Staicu	1	C. Hawthorne et al. SEQUENCE-TO-SEQUENCE PIANO TRANSCRIPTION WITH TRANSFORMERS. ISMIR 2021.
Chenyu	Shi	1	C.-S. Ahn et al. Recurrent multi-head attention fusion network for combining audio and text for speech emotion recognition. Interspeech 2022.
David	Lin	2	C. Hawthorne et al. SEQUENCE-TO-SEQUENCE PIANO TRANSCRIPTION WITH TRANSFORMERS. ISMIR 2021.
Don	SHI	2	G. Liu et al. Speech emotion recognition based on emotion perception. EURASIP 2023.
George	Doukeris	2	A. Akman et al. Evaluating the COVID-19 Identification ResNet on the INTERSPEECH COVID-19 From Audio Challenges. Frontiers in Digital Health 2022.
Guilem Ca	Ruesga	2	A. Jansson et al. Singing voice separation with deep U-Net Convolutional Betworks. ISMIR 2017.
Kaiteng	Jiang	2	Y. Ozer et al. SOURCE SEPARATION OF PIANO CONCERTOS WITH TEST-TIME ADAPTATION. ISMIR 2022.
Lieuwe	Rooijakkers	2	T. de Reuse et al. A TRANSFORMER-BASED "SPELLCHECKER" FOR DETECTING ERRORS IN OMR OUTPUT. ISMIR 2022.
Lilly	Kientz	2	D. Steele et al. A perceptual study of sound annoyance. Audio Mostly 2007.
Luc	Schreurs	2	S. Garg et al. Mouth2Audio: intelligible audio synthesis from videos with distinctive vowel articulation. Int. Journal of Speech Technology, 2023.
Lucas	Allison	3	J. Miller et al. POLAR MANHATTAN DISPLACEMENT: MEASURING TONAL DISTANCES BETWEEN CHORDS BASED ON INTERVALLIC CONTENT. ISMIR 2023.
Matthijs	Zeeuw de	3	M. Giver et al. Score-Informed Source Separation of Choral Music. ISMIR 2020.
Nathalia	Morales Rojas	3	O. Lesota et al. TRACES OF GLOBALIZATION IN ONLINE MUSIC CONSUMPTION PATTERNS AND RESULTS OF RECOMMENDATION ALGORITHMS. ISMIR 2022.
Óscar	Nebreda Bernal	3	M. Acosta et al. AN EXPLORATION OF GENERATING SHEET MUSIC IMAGES. ISMIR 2022.
Parthipan	Ramakrishnan	3	Y. Zhang et al. INTERPRETING SONG LYRICS WITH AN AUDIO-INFORMED PRE-TRAINED LANGUAGE MODEL. ISMIR 2022.
Peli	Evrenoglou	3	C.K.A. Reddy et al. MusicNet: Compact Convolutional Neural Network for Real-time Background Music Detection. Interspeech 2022.
Pim	Bax	3	C. Donahue et al. MELODY TRANSCRIPTION VIA GENERATIVE PRE-TRAINING. ISMIR 2022.
Priya	Prabhakar	3	Y. Zhang et al. INTERPRETING SONG LYRICS WITH AN AUDIO-INFORMED PRE-TRAINED LANGUAGE MODEL. ISMIR 2022.
Rajiv	Jethoe	4	C. Hawthorne et al. SEQUENCE-TO-SEQUENCE PIANO TRANSCRIPTION WITH TRANSFORMERS. ISMIR 2021.
Rob	Mourits	4	D. Regnier et al. IDENTIFICATION OF RHYTHM GUITAR SECTIONS IN SYMBOLIC TABLATURES. ISMIR 2021.
Romme	Knol	4	S. Grimm et al. Wind noise reduction for a closely spaced microphone array in a car environment. EURASIP 2018.
Roos	Wensveen	4	H. Schweiger et al. DOES TRACK SEQUENCE IN USER-GENERATED PLAYLISTS MATTER? ISMIR 2021.
ROUQIONC	CUI	4	C. Hawthorne et al. SEQUENCE-TO-SEQUENCE PIANO TRANSCRIPTION WITH TRANSFORMERS. ISMIR 2021.
Sarah	Howes	4	M. Ryynanen et al. QUERY BY HUMMING OF MIDI AND AUDIO USING LOCALITY SENSITIVE HASHING. Xxxx
Setki	Fejsko	4	R. Castellon et al. CODIFIED AUDIO LANGUAGE MODELING LEARNS USEFUL REPRESENTATIONS FOR MUSIC INFORMATION RETRIEVAL. ISMIR 2021.
Shuang	Fan	4	C.-C. Chiu et al. Self-Supervised Learning with Random-Projection Quantizer for Speech Recognition. PMLR 2022.
Shupei	Lin	5	J. Kim et al. Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech. PMLR 2021.
Siwen	Tu	5	L. Pretet et al. IS THERE A "LANGUAGE OF MUSIC-VIDEO CLIPS" ? A QUALITATIVE AND QUANTITATIVE STUDY. ISMIR 2021.
Sjors	Holtrop	5	X. Liu et al. Speaker-Aware Anti-Spoofing. Interspeech 2023.
Swati	Soni	5	C.-H. Chen at al. LANGUAGE TRANSFER OF AUDIO WORD2VEC: LEARNING AUDIO SEGMENT REPRESENTATIONS WITHOUT TARGET LANGUAGE DATA. Xxxx
Tian	Xia	5	J. Shriram et al. SONUS TEXERE! AUTOMATED DENSE SOUNDTRACK CONSTRUCTION FOR BOOKS USING MOVIE ADAPTATIONS. ISMIR 2022.
Wenhu	Li	5	R.M. Bittner et al. A LIGHTWEIGHT INSTRUMENT-AGNOSTIC MODEL FOR POLYPHONIC NOTE TRANSCRIPTION AND MULTIPITCH ESTIMATION. ICASSP 2022.
Wouter	Ebing	5	Y. Getman et al. wav2vec2-based Speech Rating System for Children with Speech Sound Disorder. Interspeech 2022.
Xiang	He	5	A. C. Mendes da Silva et al. HETEROGENEOUS GRAPH NEURAL NETWORK FOR MUSIC EMOTION RECOGNITION. ISMIR 2022.
Xiaolin	Gu	5	G. Shibata et al. MUSIC STRUCTURE ANALYSIS BASED ON AN LSTM-HSMM HYBRID MODEL. ISMIR 2020.
Chris	Tsirogiannis	5	A. Raford et al. Robust Speech Recognition via Large-Scale Weak Supervision. Xxxx
Abed	Alrahman Hettini	5	A. Badi et al. SKYE: More than a conversational AI. Interspeech 2022.