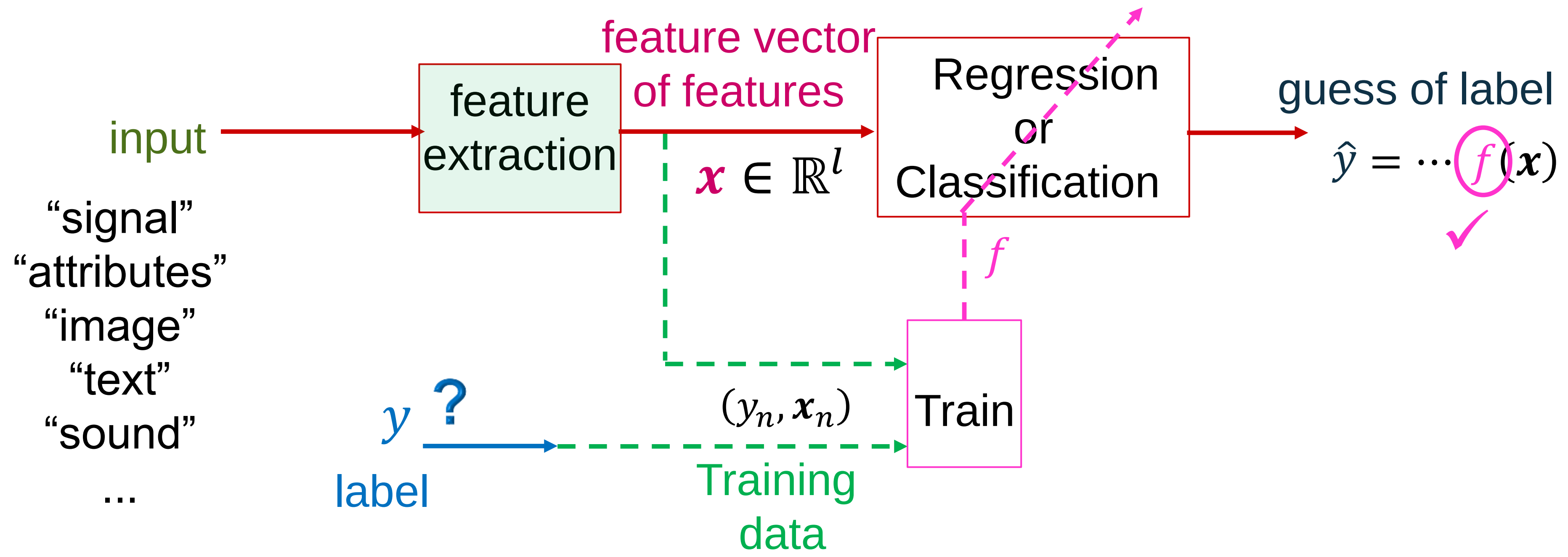
Manual feature extraction

No chapter in book

Recollect: Inference



Feature extraction methods

- Several steps for Feature extraction
- Operationally, Feature extraction = Dimensionality reduction
- Example: signal waveform (infinite dimensions) $\rightarrow$ 10-dimensional $\boldsymbol{x}$
- Two ways of doing Feature extraction ...

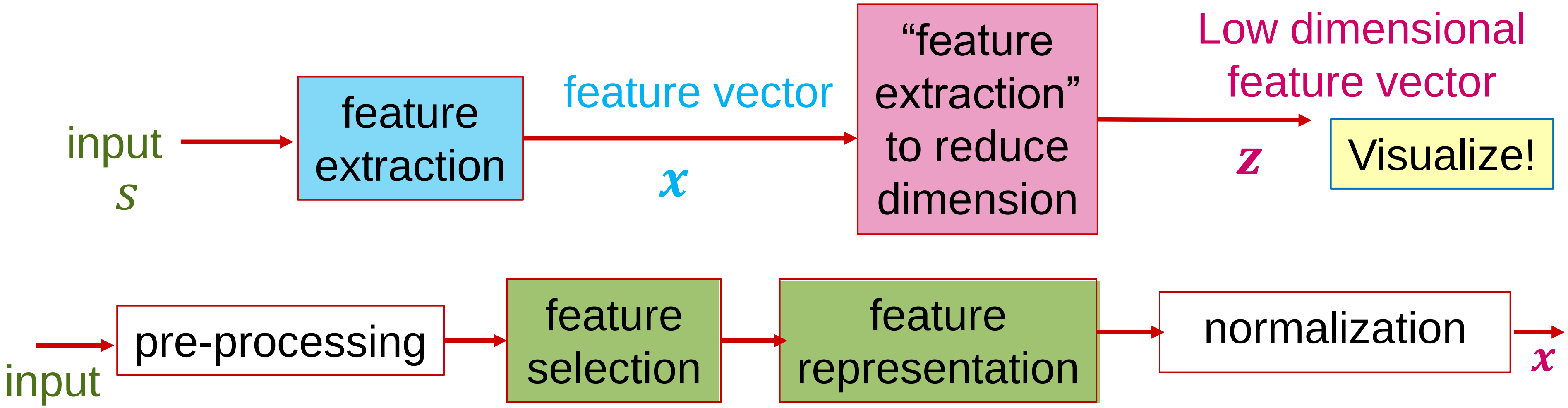
1. Using <u>Automatic</u> dimensionality reduction methods
   - Useful if no domain knowledge
   - Need lots of Training data to overcome lack of domain knowledge
   - Features from Deep learning networks popular nowadays

input → pre-processing → feature selection → feature representation → normalization → $\boldsymbol{x}$

Feature extraction methods

2. Using <u>Manual</u> methods
   - Highly domain-specific: Images versus Text versus Audio
   - Better features if we don't have lots of Training data

- Discuss mainly Feature selection and Feature Representation
- Feature extraction use: For <u>Classification/Regression</u>
- Feature extraction use: For <u>visualization</u> if $\dim(\boldsymbol{z}) \leq 3$



Manual feature extraction

- Discuss Manual feature extraction today
  - Domain-specific methods using expert knowledge
  - e.g., Weather prediction: Rainfall? Humidity matters, not Wind speed
- Manual methods for following applications:
  1. Text processing
  2. Image/Video processing
  3. Audio processing
  4. Time series

Course project in these applications must use at least one Manual feature extraction method

- `sklearn.feature_extraction` package of Python
- Caution: Lecture mainly to provide a flavor of such methods.

  Read research papers for latest ideas

Feature extraction:
Text applications

Text applications

- Examples:
  1. Predict next word in a sentence
  2. Classify document as belonging to a particular topic
  3. Mark email as spam or not-spam
  4. Classify sentence as ‘positive sentiment’ or ‘negative sentiment’
  5. ...

- `nltk` (Natural Language Toolkit) package of Python
  — Matlab also has a module, but not free to CMU

Preprocessing

> Carnegie Mellon University (CMU) is a private research university based in Pittsburgh, Pennsylvania. Founded in 1900 by Andrew Carnegie as the Carnegie Technical Schools, the university became the ...

Wikipedia

- Pre-processing for Text: (Not all steps always needed)

1. Split into sentences
2. Convert to lower case, Remove punctuation
3. Remove STOP words (‘a’, ‘the’, ‘is’, ‘in’, ...) ← e.g., Use “STOP words dictionary”
4. Tokenize: Sentence → distinct words
   `nltk.tokenize.sent_tokenize`
5. Stemming (Remove suffix): ‘running’ → ‘runn’
6. Lemmatize (Sophisticated alternative to stemming): ‘ran’ → ‘run’ (root word)
   `nltk.stem.wordnet.WordNetLemmatizer`

Reduces over-fitting (steps 5–6)

(document) input → **pre-processing** → feature selection → feature representation → normalization → x

Feature selection and representation

> Carnegie Mellon University (CMU) is a private research university based in Pittsburgh, Pennsylvania. Founded in 1900 by Andrew Carnegie as the Carnegie Technical Schools, the university became the …

Wikipedia

- <u>Feature selection and representation for Text</u>:

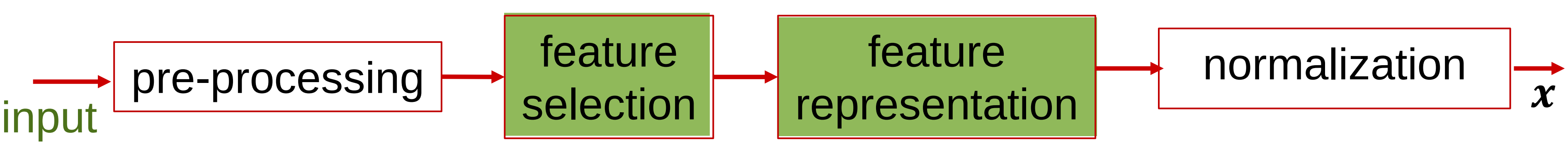
1. Bag-of-Words: (We used this in Spam SMS example)

   - Choose dictionary (l most common words in Training documents)
   - Feature vector $\boldsymbol{x}$ = Is dictionary word is in document?
   - Feature vector $\boldsymbol{x}$ = Number of occurrences of dictionary word in document

$$\boldsymbol{x} = \begin{bmatrix} 1 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} 3 \\ \vdots \\ 0 \\ 13 \end{bmatrix}$$

← merged is present
⋮
← tourism is absent

← merged occurs 3 times
⋮
← tourism is absent



Feature selection and representation

- <u>Feature selection and representation for Text</u>:

1. Bag-of-Words: Adjust for document length and word rarity
   - Term frequency $f(t, x)$ = Fraction of times word t appears in document x
   - Document frequency $N(t)$ = Fraction of documents containing word t
   - Term frequency–Inverse document frequency: Rare words emphasized

$$\text{tf-idf} = \log(1 + f(t, x)) \log \frac{1}{N(t)}$$

$$\boldsymbol{x} = \begin{bmatrix} {}^{3}/_{378} \\ \vdots \\ 0 \\ {}^{13}/_{378} \end{bmatrix}$$

← merged occurs 3 times in a 378 word document

$$\boldsymbol{x} = \begin{bmatrix} 3.14 \\ \vdots \\ 0 \\ 1.426 \end{bmatrix}$$

← merged has this tf-idf

input → pre-processing → feature selection → feature representation → normalization → $\boldsymbol{x}$

Feature selection and representation

Carnegie Mellon University (CMU) is a private research university based in Pittsburgh, Pennsylvania. Founded in 1900 by Andrew Carnegie as the Carnegie Technical Schools, the university became the …

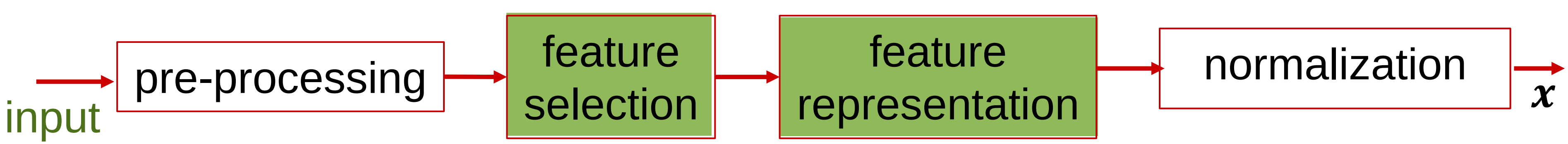
Wikipedia

- Feature selection and representation for Text:

2. Bi-grams: Bag-of-Words, but using Dictionary of Bi-grams
   - Bi-gram = Two consecutive words
   - Advantage: Bi-gram better captures context
   - Disadvantage: Dictionary very large. Over-fitting possible

private research
university became

3. N-grams: Triplet of consecutive words, etc.
   - Generally, 3-grams is maximum used. Else severe over-fitting



Feature selection and representation

Carnegie Mellon University (CMU) is a private research university based in Pittsburgh, Pennsylvania. Founded in 1900 by …

Wikipedia

- Feature selection and representation for Text:

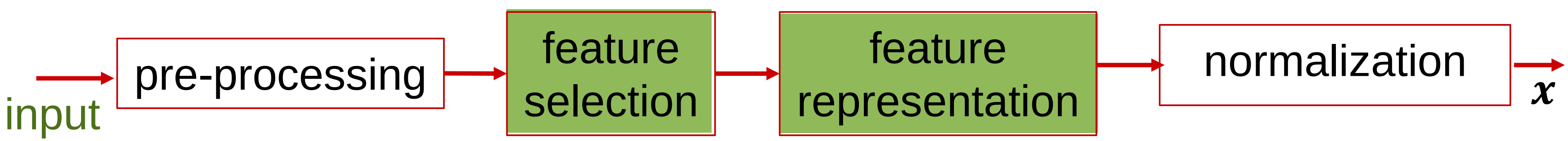
4. Embeddings: Using a pre-trained model, convert a word, into a 'vector embedding' $\boldsymbol{x}$

Automatic Feature extraction, not Manual !

- Embeddings often satisfy intuitive relationships

```
import spacy
nlp = spacy.load('en_core_web_sm')
```

research → word2vec embedding → $\boldsymbol{x} = \begin{bmatrix} 0.1412 \\ \vdots \\ -0.3521 \\ 0.0525 \end{bmatrix}$



Feature selection and representation

Carnegie Mellon University (CMU) is a private research university based in Pittsburgh, Pennsylvania. Founded in 1900 by …

Wikipedia

- <u>Feature selection and representation for Text</u>:

4. Embeddings: Using a <u>pre-trained model</u>, convert a ~~word~~, sentence or even a document into a 'vector embedding' $\boldsymbol{x}$

Carnegie Mellon University (CMU) is a private …

Automatic Feature extraction, not Manual !

Google universal sentence encoder

$$\boldsymbol{x} = \begin{bmatrix} 0.0342 \\ \vdots \\ -0.071 \\ 0.02125 \end{bmatrix}$$

input → pre-processing → feature selection → feature representation → normalization → $\boldsymbol{x}$

Example: Universal Sentence Encoder embedding

	Question1	Question2	DuplicateIndicator
4126	Why should I still vote for Hillary Clinton?	Why shouldn't I vote for Hillary Clinton?	0
4350	What are some of the strangest addictions ever?	What are some of the strangest addictions?	1
4493	What is metamorphic rock?	How are metamorphic rocks classified?	0
4745	What is the fastest land mammal?	What are the South American Land Mammal Ages?	0
5001	How can you tell if you are a narcissist?	How can I identify a narcissist?	1

- **Quora question pairs***
- Attribute Information:
  - Pairs of questions
  - Duplicate indicator: Question pair IS NOT versus IS duplicate
- Inference task: Classify question pair as being duplicate or not
- Popular classifiers usable if each Question i $\rightarrow$ feature vector $\boldsymbol{x}_i$

https://www.kaggle.com/c/quora-question-pairs

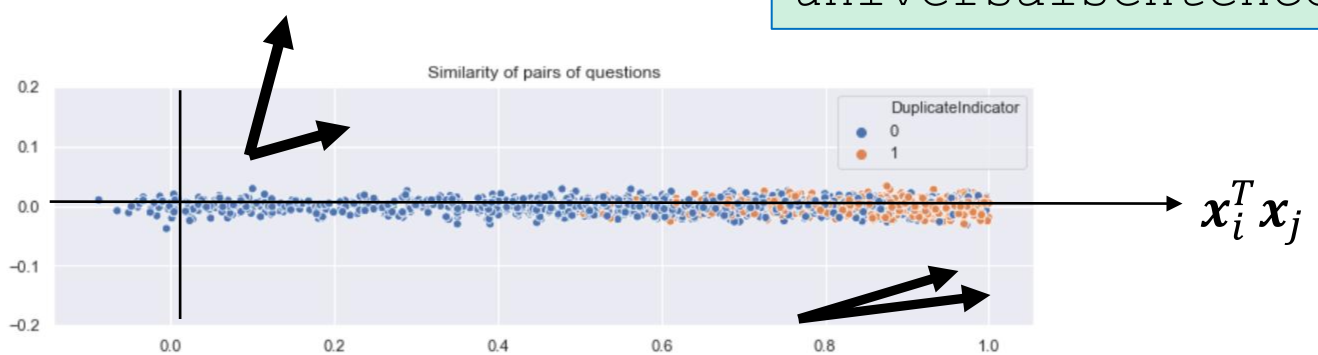
Example: Universal Sentence Encoder embedding

	Question1	Question2	DuplicateIndicator	Embedding1	Embedding2
4126	Why should I still vote for Hillary Clinton?	Why shouldn't I vote for Hillary Clinton?	0	[-0.0013395054, -0.099904045, 0.06714142, -0.0...	[-0.010914813, -0.09277783, 0.06034255, -0.037...
4350	What are some of the strangest addictions ever?	What are some of the strangest addictions?	1	[-0.019629745, 0.0375768, -0.028663237, 0.0502...	[-0.019145647, 0.038091455, -0.025344428, 0.05...
4493	What is metamorphic rock?	How are metamorphic rocks classified?	0	[-0.062192235, 0.0700693, 0.026325881, 0.06246...	[-0.041530643, 0.003250799, 0.027312022, 0.065...
4745	What is the fastest land mammal?	What are the South American Land Mammal Ages?	0	[-0.047259703, 0.025728384, -0.0036811058, 0.0...	[-0.046342198, -0.01937709, 0.030181423, -0.06...
5001	How can you tell if you are a narcissist?	How can I identify a narcissist?	1	[-0.08539912, 0.04583511, 0.11148303, -0.00119...	[-0.074138455, 0.032638546, 0.08159932, -0.032...

512 dimensions

- Feature vector $\boldsymbol{x}$ using Universal Sentence Encoder embedding
- Measure similarity of Questions i, j as inner product $\boldsymbol{x}_i^T \boldsymbol{x}_j$
- Embedding $\boldsymbol{x}$ gives reasonable question similarity measure
- $\Rightarrow$ $\boldsymbol{x}$ legitimate feature vector

`tensorflow` package
`tensorflow_hub` package
`universalsentenceencoder_model`



Feature extraction:
Image and Video applications

Image and Video applications

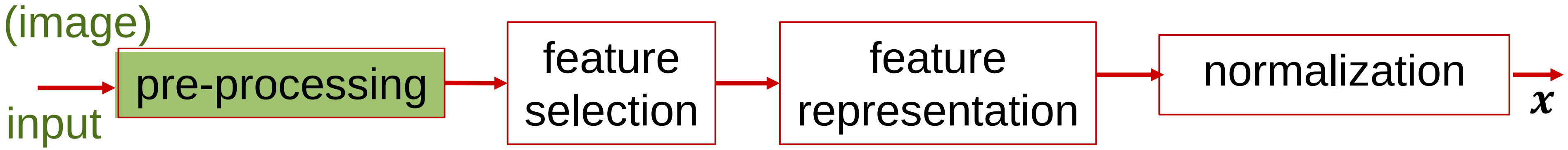
- Examples:
  1. Optical character recognition (OCR) in scanned text
  2. Face recognition in image
  3. Tracking target in a video
  4. ...

- OpenCV (`opencv-python`) package of Python
- Patented algorithms (like SIFT and SURF) in `opencv-contrib-python` package
  - (Free for non-commercial use)

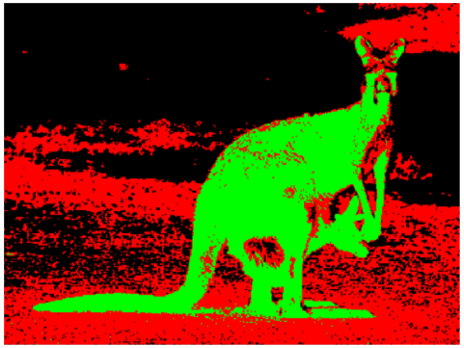
Preprocessing

- Pre-processing for Images: (Not all steps always needed)

1. Crop and re-sample to single size (e.g., 1000 × 800 pixels)
2. Down-sample to smaller size (for less computations)
3. Noise filtering (Gaussian blur)
4. Correct lighting (histogram normalization)
5. Color to Gray-scale (needed by some feature extractors)



Preprocessing



$k = 3$
clusters

- Pre-processing for Images:

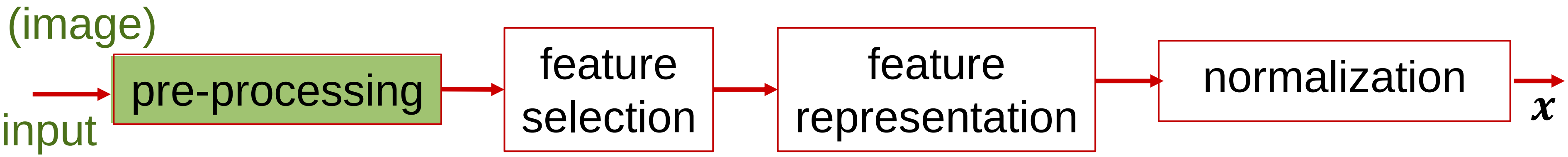
5. Segment image into patches. Approaches ...

   — Quantize intensity values

   — Plot Histogram of colors $\rightarrow$ Each peak gives one patch

   — Cluster pixels into k clusters (k −means or Gaussian clustering)

   — Detect edges $\rightarrow$ Join to get boundaries $\rightarrow$ Get patches

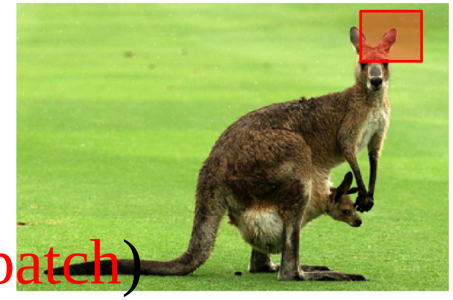


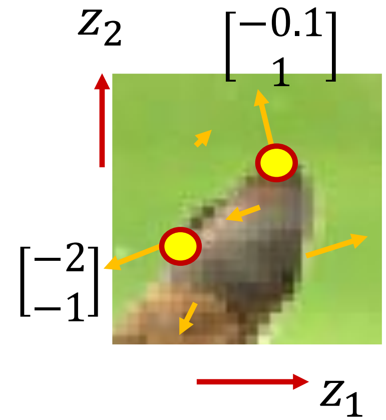
Feature selection

- <u>Feature selection for Images</u> :

1. Histogram of Gradients (HOG):

   (Intensity gradients in each image patch)





   — Quantize angles $0 - 360^o$ into 8 bins

   — Image patch: At each pixel $\boldsymbol{z} = (z_1, z_2)$, find gradient $G(z_1, z_2)$ of intensity I

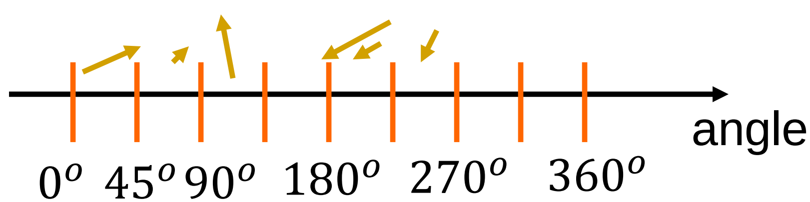
   — Bin gradient G by angle. Increase bin's weight by $|G(z_1, z_2)|$

   — Normalize $\boldsymbol{u}$ $\rightarrow$ $\boldsymbol{u}$ is called HOG vector of that image patch

   — $\Rightarrow$ Image features selected = List of HOG vectors

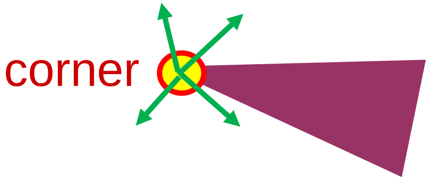
$$G(z_1, z_2) = \frac{\partial I}{\partial \boldsymbol{z}} = \begin{bmatrix}\frac{\partial I}{\partial z_1}\\ \frac{\partial I}{\partial z_2}\end{bmatrix}$$

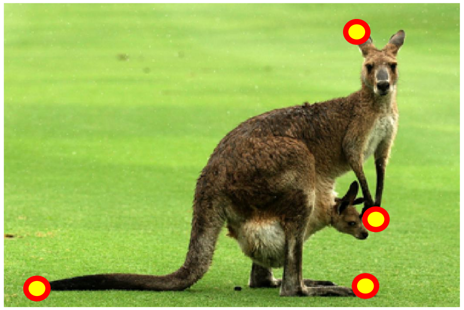
8-dim $\boldsymbol{u}$ = bin weights vector



input → pre-processing → feature selection → feature representation → normalization → $\boldsymbol{x}$

Feature selection





- <u>Feature selection for Images</u> (continued):

2. List of Keypoints: Keypoint = Corner in image

   — Harris Corner Detection: Corner ≐ Large change in intensity in all directions

   — Scale-Invariant Feature Transform (SIFT): Patented popular corner detector.

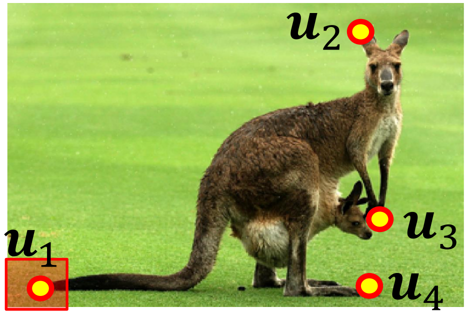
   - Uses Harris corner detection + Several image transforms (e.g., scaling)
   - Detect corners invariant to scale, intensity variation, and rotation of image

   — Speeded Up Robust Features (SURF): Also patented

   - Similar to SIFT, but fast calculations

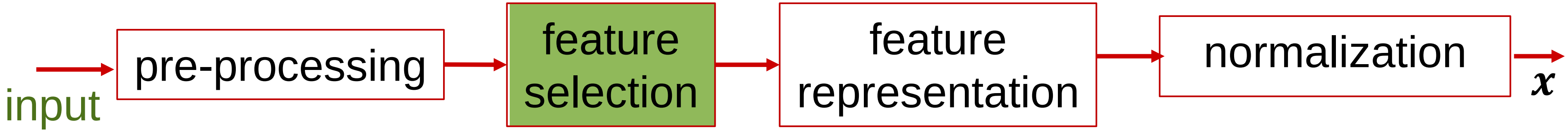
input → pre-processing → **feature selection** → feature representation → normalization → x

Feature selection

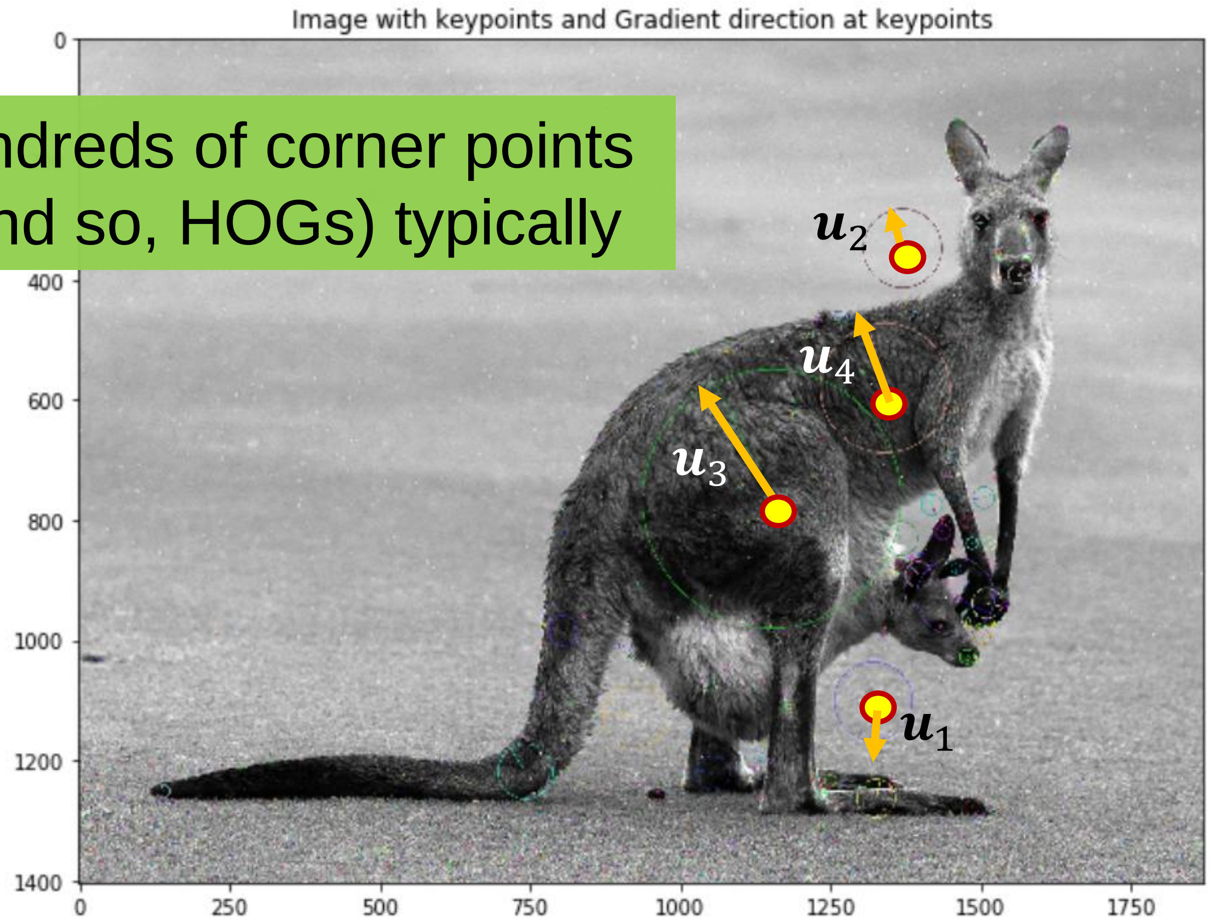


- Feature selection for Images (continued):

3. HOGs but only for Keypoints: The most popular method
   - Detect corners (Harris or SIFT or SURF)
   - Only consider image patch around corner points
   - For each such patch, calculate HOG vector $\boldsymbol{u}_i$
   - $\Rightarrow$ Image features selected = List of corner points and their HOGs $(\boldsymbol{u}_i)$



Example

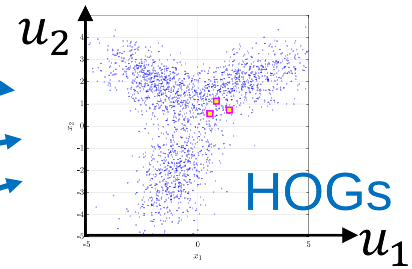


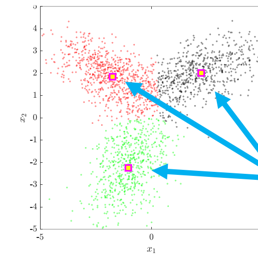
Hundreds of corner points (and so, HOGs) typically

input → pre-processing → feature selection → feature representation → normalization → x

Feature representation

. . .





3 clusters

3 cluster heads

- <u>Feature representation for Images</u> : Depends on application

1. Bag-of-<u>Visual</u>-Words: Bag of Words using HOGs

   Dictionary

   $$x = \begin{bmatrix} 3 \\ \vdots \\ 0 \\ 13 \end{bmatrix}$$

   ← cluster head (3 times in image)

   . . .

   ← cluster head (absent in image)

   — E.g., Useful for image classification

   — Each Training image → List of HOGs $\boldsymbol{u}_i$

   — Cluster the HOGs → Each Cluster-head = "Visual word"

   — Image: Assign each HOG in it to its cluster-head ("Visual word")

   — ⇒ Image = HOGs = Bag of Visual Words = $\boldsymbol{x}$ feature vector (captures shape)

(image) input → pre-processing → feature selection → feature representation → normalization → $\boldsymbol{x}$

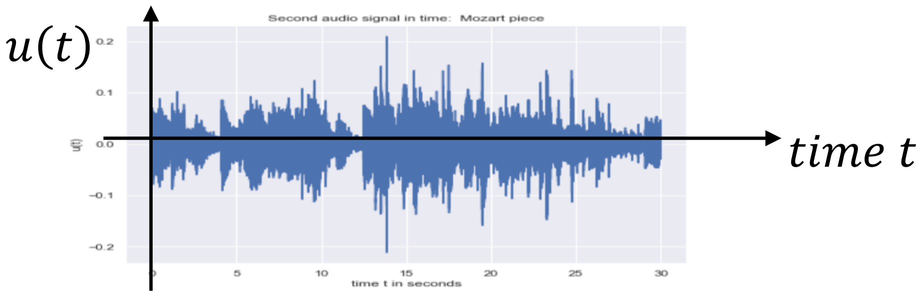
Feature extraction: Audio applications

Audio applications

- Examples:
  1. Recognizing musical compositions
  2. Voice recognition
  3. De-noising microphone signal
  4. ...

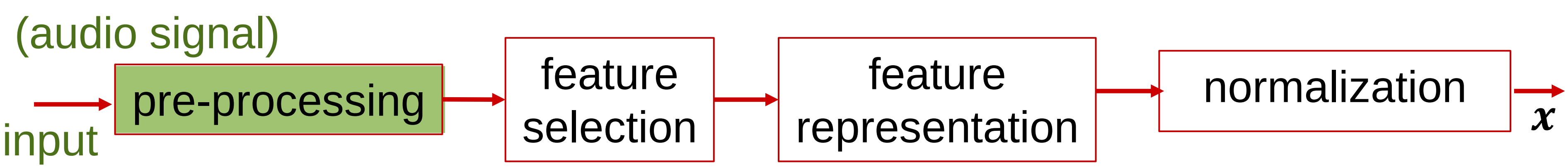
- `librosa` package in Python
  - Other audio packages for Python too

Preprocessing



- <u>Pre-processing for Audio</u>: Not all steps always needed

1. Resample signal to one sampling rate, say, 22 kHz.
2. Normalize signal to ± 1
3. Convert from stereo to mono (average the two channels)
4. Remove irrelevant intervals (e.g., pauses or noise)
5. …



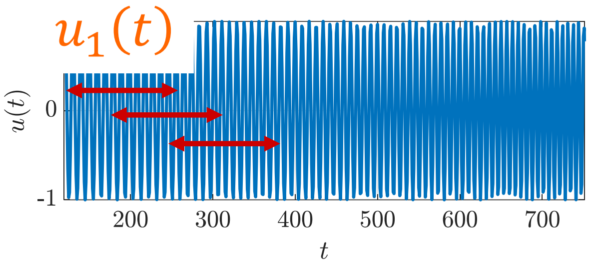
Feature selection and representation

- Let $\mathcal{F}$ be Fourier transform
- <u>Feature selection and representation</u>: Call audio signal as $u(t)$

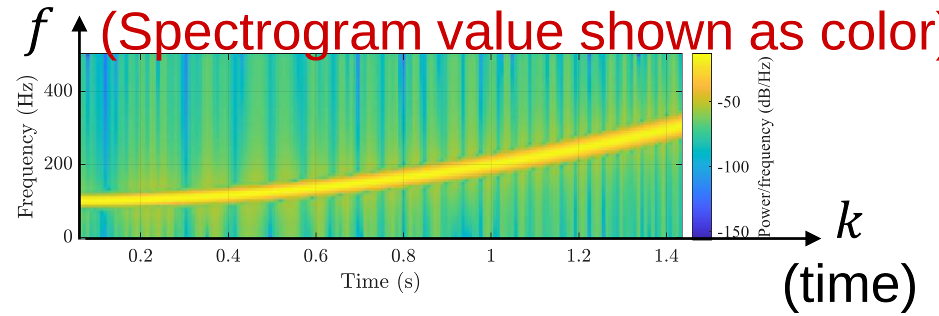
1. <u>Spectrum</u>: Measures frequency content of audio signal
   - Calculate $U(f) = |\mathcal{F}[u(t)]|^2$
   - Sample $U(f)$ at discrete frequencies $\rightarrow$ Finite dimensional feature $\boldsymbol{x}$

(audio signal)
input → pre-processing → feature selection → feature representation → normalization → $\boldsymbol{x}$

Feature selection and representation



chirp signal



2. <u>Spectrogram</u>: Better for <u>non-stationary</u> signals like music

- Break $u(t)$ into 25 *millisec* windowed segments $u_k(t) = u(t)\, w(t - kT)$
- (Consecutive Hamming/Hanning windows usually overlap)
- Calculate Spectrum of $u_k(t)$ (Short Term Fourier Transform – STFT)

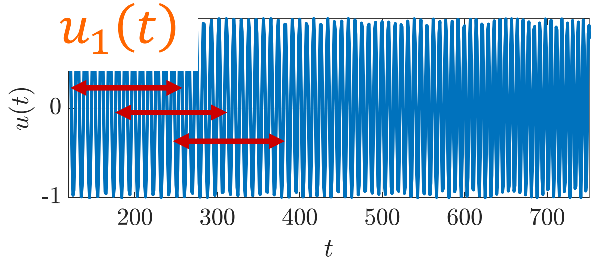
$$U_k(f) = |\mathcal{F}[u_k(t)]|^2$$

- Sample the frequency f $\rightarrow$ Finite dimensional feature $\boldsymbol{x}_k$
- More useful for <u>visualization</u> of audio

input → pre-processing → feature selection → feature representation → normalization → $\boldsymbol{x}$

Feature selection and representation

$u_1(t)$



3. <u>Cepstrum</u> C: As earlier, audio $u(t) \rightarrow$ windowed $u_k(t)$

$$C_k(t) \doteq |\mathcal{F}^{-1}[\ln|\mathcal{F}[u_k(\tau)]||]|^2$$

t is called 'quefrency' in milliseconds

frequency $\rightarrow$ "time" t

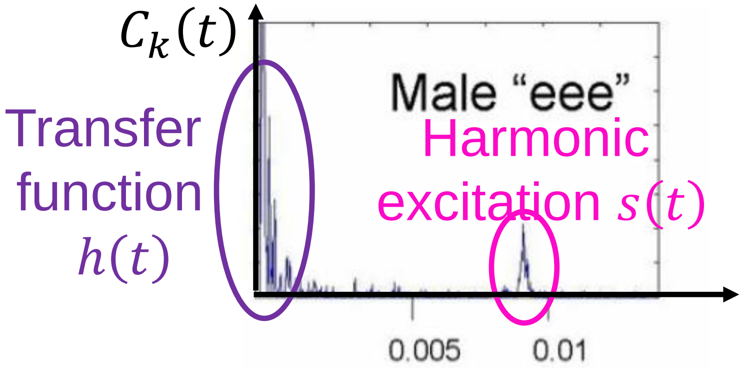
logarithm to compresses

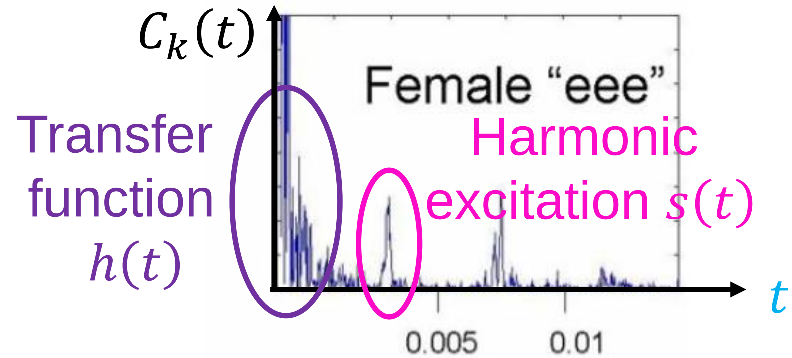
Time $\tau \rightarrow$ frequency (spectrum)

— Sample $C_k(t)$ waveform $\rightarrow$ Finite dimensional feature $\boldsymbol{x}$

input $\rightarrow$ pre-processing $\rightarrow$ feature selection $\rightarrow$ feature representation $\rightarrow$ normalization $\rightarrow$ $\boldsymbol{x}$

Feature selection and representation





- Cepstrum found useful in music, speech processing. Intuition?
- Separates Vocal tract transfer function from Harmonic excitation

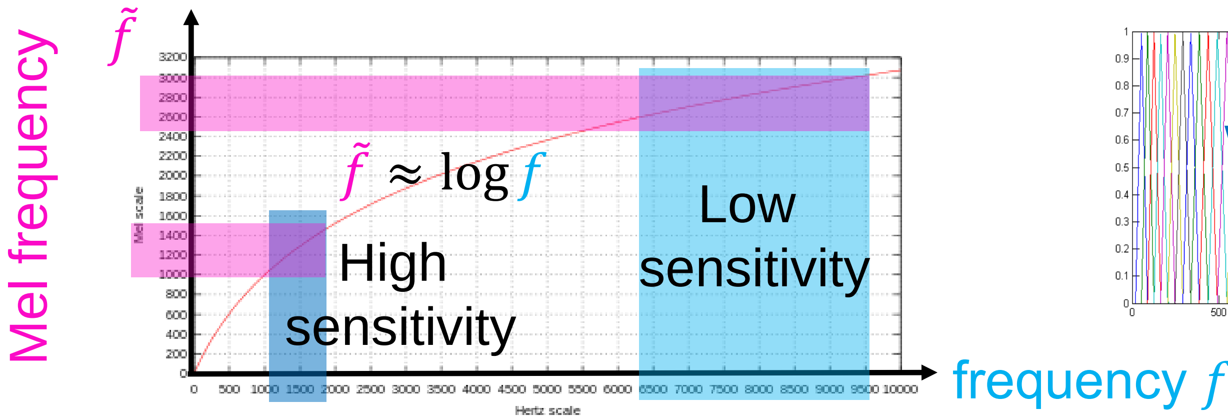
Speech model

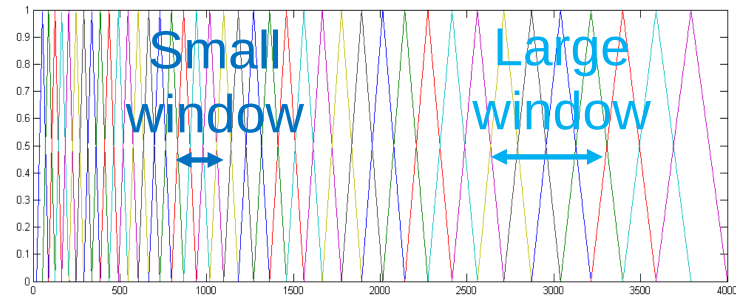
Harmonic excitation $s(t)$ → Vocal tract $h(t)$ → Speech $u(t) = h(t) * s(t)$

$$\mathcal{F}^{-1}\ln|\mathcal{F}[u(t)]| = \ln|\mathcal{F}[h(t) * s(t)]| = \ln|\mathcal{F}[h(t)]\mathcal{F}[s(t)]|$$

$$= \underbrace{\mathcal{F}^{-1}\ln|\mathcal{F}[h(t)]| + \mathcal{F}^{-1}\ln|\mathcal{F}[s(t)]|}_{h(t)\text{ separated from }s(t)}$$

Feature selection and representation





4. <u>Mel Frequency Cepstral Coefficients (MFCC)</u>: Cepstrum modified

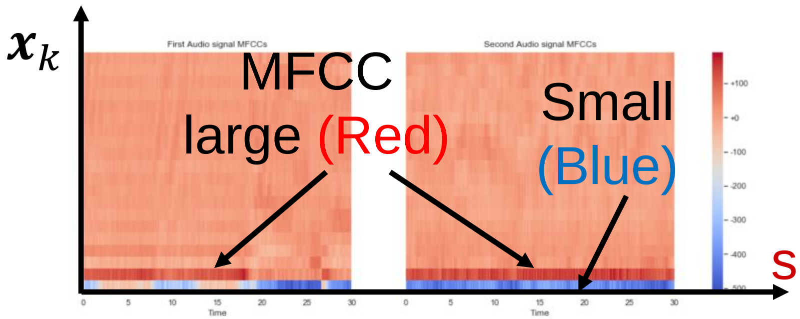
$$U(f) = \mathcal{F}[u_k(t)] \xrightarrow{\text{cepstrum}} C_k(t) = |\mathcal{F}^{-1}[\ln|U(\tilde{f})|]|^2$$

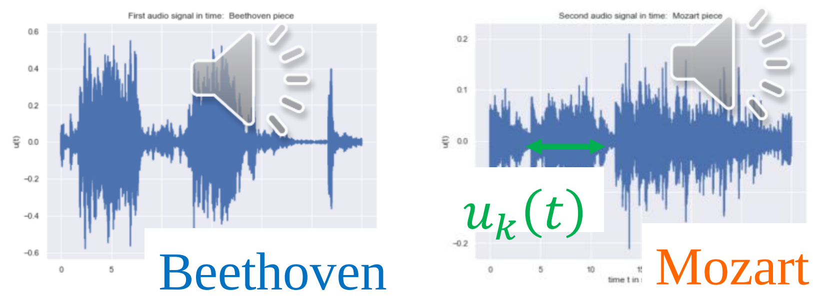
- Motivation: Ear is more sensitive to <u>low</u> frequencies. So, warp $f \rightarrow \tilde{f}$
- $f \rightarrow \tilde{f}$ practically: Integrate $|U(f)|^2$ with Triangular windows
- $\mathcal{F}^{-1}$ practically: Use Inverse Discrete Cosine Transform (IDCT)
- $\rightarrow$ Resulting $C_k(t)$ called MFCC

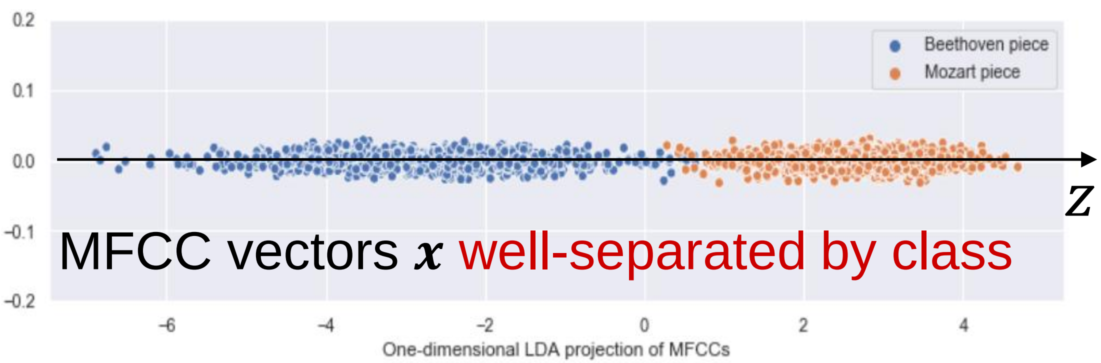
input → pre-processing → feature selection → feature representation → normalization → x

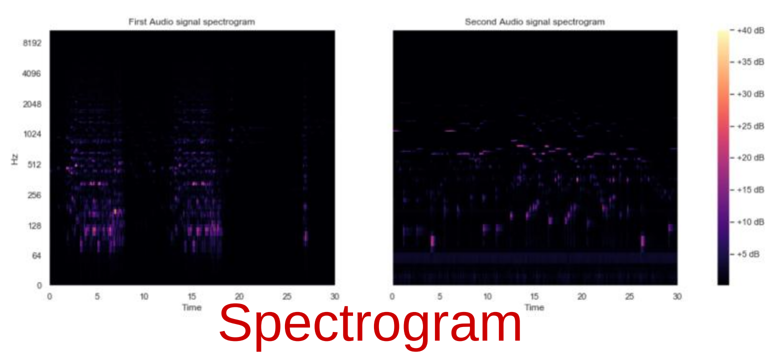
Example: MFCC

- Example: Two audio files – Beethoven piece and Mozart piece
  - Binary classification: Classify short segment $u_k(t)$ as Beethoven or Mozart
  - Time domain signal and Spectrograms not very informative of class here
  - So, each segment $u_k(t)$ $\rightarrow$ Compute 20 −dim MFCC $\boldsymbol{x}_k$ feature
  - Visualize all $\boldsymbol{x}_k$ by LDA-projecting into 1-dim space (z_n)





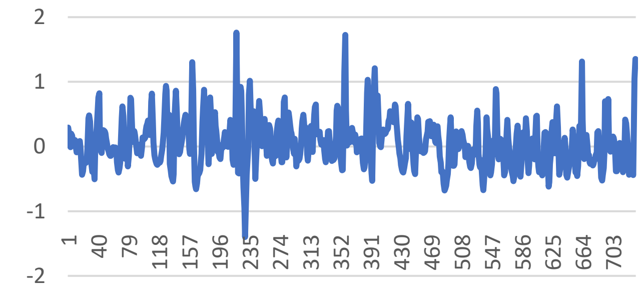


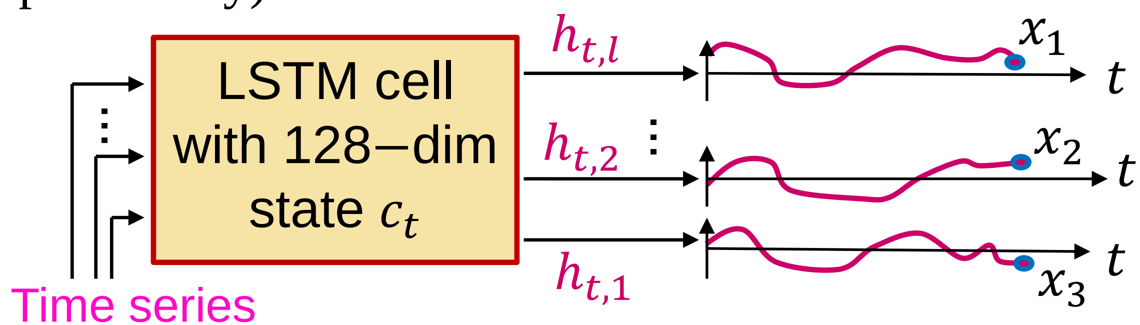


Feature extraction:
Time series applications

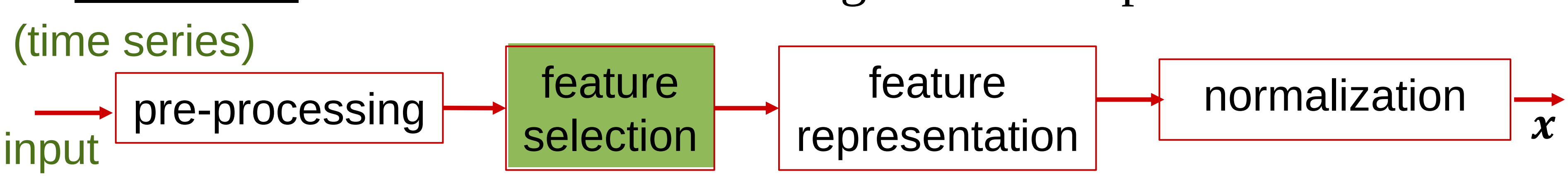
Time series application: Classification of series

- Classification of a stationary time series: Class of entire time series?
  - Example: Smartphone accelerometer time series → walking or sleeping person?
- Feature selection in time series: Calculate various metrics
  - Mean, variance, skew, kurtosis of series samples
  - Autocorrelation function at various lags
  - Centroid of FFT (measures periodicity)
  - Lempel-Ziv complexity
  - ...
  - See `tsfresh` package



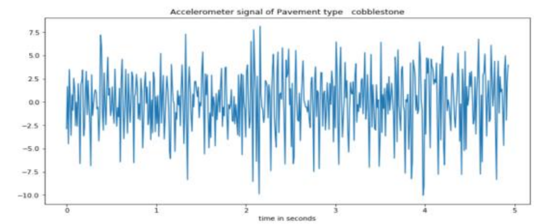


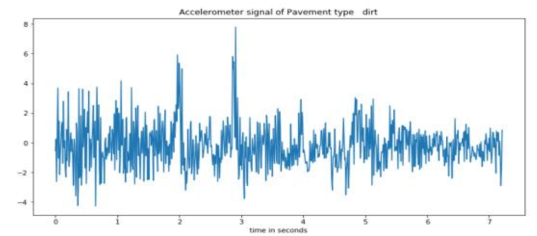
- Automatic feature extraction $\boldsymbol{x}$ using LSTM deep network

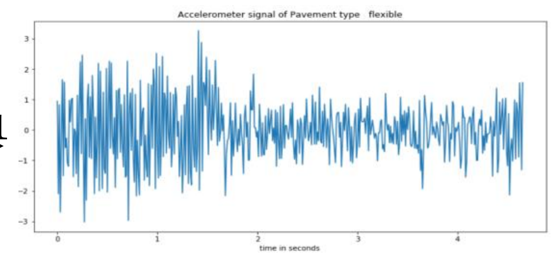


Example: Driving on different pavements

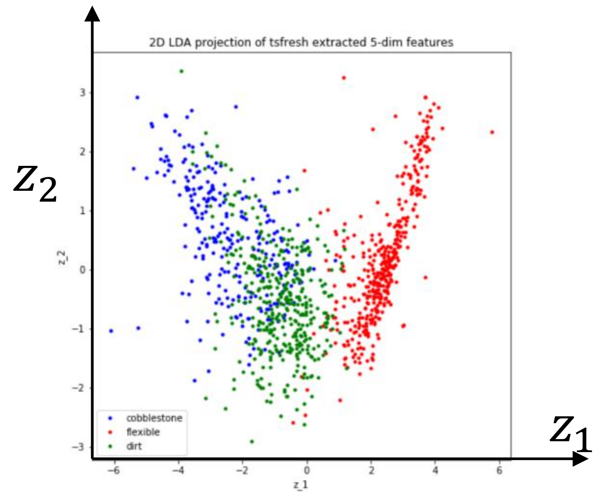
- **`Driving on different pavements dataset*`**
  - — Accelerometer time series of car driving on Cobblestone, Flexible, Dirt
  - — Goal: Classify time series to these 3 classes







- `tsfresh`: Each series $\rightarrow$ 5-dimensional $\boldsymbol{x}$
  - — Autocorrelation at lag = 1
  - — Centroid of FFT
  - — Kurtosis
  - — Variance
  - — LZ complexity



- Visualization: Project $\boldsymbol{x}$ into $\mathbb{R}^2$ using LDA
- $\rightarrow$ Good class separation $\Rightarrow$ $\boldsymbol{x}$ useful for classification

`http://www.timeseriesclassification.com/description.php?Dataset=AsphaltPavementType`