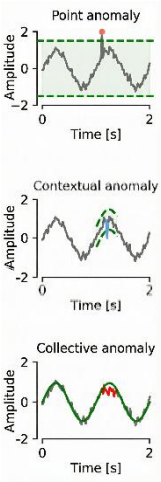
DISCLAIMER

This document is a personal compilation of lecture notes and does not constitute an official or verbatim transcript. While every effort has been made to ensure accuracy, the content may contain errors, omissions, or misinterpretations. Sections marked with "(*credo*)" indicate points where the lecturer's statements were not entirely clear and reflect my own understanding. These notes are intended solely for personal study and educational purposes and are not a substitute for the official course materials.

LECTURE 13 (ANOMALY DETECTION)

Taxonomy of Anomaly Detection



Point Anomalies An observation that deviates from trend.

- e.g. packet losses not expected, credit card transaction.

Contextual Anomalies Violation of a seasonal or contextual trend.

- e.g. 100mm/year rainfall is strange for London, not for Sahara; large credit card expenses on Christmas period.

Collective Anomalies Conditional/contextual anomalies w.r.t to the full dataset but not w.r.t. each other (deviation from a pattern).

- e.g. heart skip a beat.

Signal Decomposition

Signals or time-series (observed) can be decomposed into several parts:

- Trend
- Seasonal components (repeat it additionally components, repeat it on time [kind of periodic])
- Noise (the parts that you don't want, you want to get rid of that)

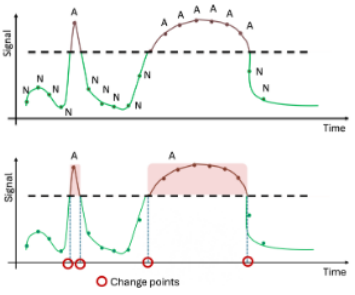
More advanced decomposition:

You add components, then you separate the additional component in order to extract that's different elements. You can have multiplicative components or cyclical components.

- Multiplicative decomposition
- Cyclical components (depends on the signal, e.g., PRNU)

The idea is to extract trends and repetitive components getting rid of noise.

AD Evaluation



Anomalies can be mainly divided into:

- **Point anomalies:** *outlier detection* (binary classification where you identify all the outliers given the sets of samples) (example packet dropping – detect you over the threshold or not).
- **Collective anomalies** (consecutive sequence of anomaly points): *change point detection*.(you want to detect an interval of possible anomalies -change point: you select the instant of the sample where the signal from being nominal start to becomes anomalous, and then you select the points where the signal comes back in a normal state.)

Metrics:

- Binary metrics ← *outlier detection*
- Detection time (or point #)
- Window-based detection ← *change point detection*

Binary metrics for change detection applied in window-based way; check if a predicted change point falls into the detection window OR compare between predicted and true windows.

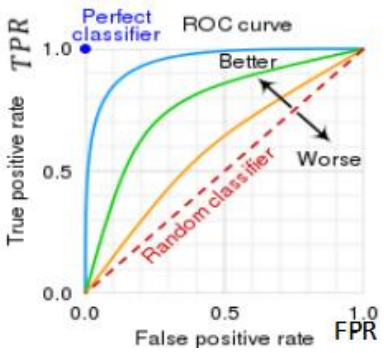
AD EVALUATION METRICS

- **Binary metrics:** number of occurrences with the same label. (how many nominal/anomalous points are correctly identify)
    - Examples: FDR/TPR/Recall/Sensitivity, FAR/FPR, MAR (), Specificity, Precision, Accuracy, G-mean, F-measure, ROC on AUC or PRC, MCC.
- **Detection time (or point #):** I can detected the difference between ground truth time and the real one. (I can compare the real interval, which is detected by my anomaly detection)
    - Examples: ADD/MAE/Annotation Error, MSD, MSE/RMSE/NRMSE, Hausdorff.
- **Window-based detection:** match predicted change point with the detection window around the actual change point in a way different from binary classification.
    - Examples: NAB scoring index, RandIndex.

AD Binary Metrics

Assuming that the decision is binary ("outlier or anomaly" A or "nominal" N), we can have four possible conditions in a **Confusion Matrix**:

The Confusion Matrix is a simple matrix where in the raw we have the real cases and in the column we have what is detected.



- Sample anomalous is classified as anomaly: TP
- Sample anomalous is classified as nominal: FN
- Nominal sample is classified as anomalous: FP
- Nominal sample is classified as nominal: TP

Real \ Estimated	A	N
A	True positive (TP)	False negative (FN)
N	False positive (FP)	True negative (TN)

Key Performance Indicators:

- **Precision** $\frac{n_{TP}}{n_{TP}+n_{FP}} = \frac{\text{detected true anomalies}}{\text{samples classified as anomalies}}$ (also known as Positive Predicted Value - PPV): where n_{TP} is the number of True Positive. The higher it is the more you can relay on the output of the classification.
- **Recall** $\frac{n_{TP}}{n_{TP}+n_{FN}} = \frac{\text{detected true anomalies}}{\text{total anomalies}}$ (also known as True Positive Rate - TPR or Sensitivity): this metric measure how much you are able to detect anomaly that come in your network), typically you want this very high.
- **Specificity** $\frac{n_{FP}}{n_{TN}+n_{FP}} = \frac{\text{detected false anomalies}}{\text{total nominals}}$ (also known as False Positive Rate - FPR): same thing to Recall but refer to nominal point. You want this small as possible.
- **Accuracy** (of the classifier): $\frac{n_{TP}+n_{TN}}{n_{total}}$, that's not specify whether your attack are all classified or not.
- **F1 Score**: $2 \cdot \frac{precision \cdot TPR}{Precision+TPR} = \frac{2TP}{2TP+FP+FN}$, Compares arithmetic and algebraic mean.
- **AUC (Area Under Curve)**: referred to ROC curve. Your detector is tuned/control by its threshold, this threshold can change, that can increase the number of false positive or decrease the possibility

of false negative. Changing this threshold generates a curve, this curve is obtain by plotting in the diagram the false positive rate and the true positive rate. This curves should be the more as possible, because this means that I have very low number of FP rates and very high TP rates. For representing this curve with one simple metric we use AUC. Sometimes we add another metric, the Equal Error Rate or Equal Accuracy Rate (ETR), this gives you the point where the number of TP is equal to 1-number of FP.

- **G-mean**: $\sqrt{\text{Sensitivity} \cdot \text{Specificity}}$,
- **MCC (Matthews Correlation Coefficient)**: $\frac{(n_{TP} \cdot n_{TN}) - (n_{FP} \cdot n_{FN})}{\sqrt{(n_{TP}+n_{FP}) \cdot (n_{TP}+n_{FN}) \cdot (n_{TN}+n_{FP}) \cdot (n_{TN}+n_{FN})}}$

measures quality of binary classification

Technique	Recall	FPR	Accuracy	F-1	MCC	Run Time
k-NN	85.71	0.38	97.39	85.71	85.32	≤1
LOF	78.57	0.58	97.38	78.57	77.98	≤1
COF	57.14	1.16	97.35	57.14	55.97	≤1
aLOCI	85.71	0.38	97.39	85.71	85.32	≤69
LoOP	42.85	1.55	97.33	42.85	41.29	≤1
INFLO	57.14	1.16	97.35	57.14	55.97	≤1
CBLOF	92.85	0.19	97.40	92.85	92.66	≤1
LDCOF	85.71	0.38	97.39	85.71	85.32	≤1
CMGOS	57.14	1.16	97.35	57.14	55.97	≤1
HBOS	28.57	1.94	97.32	28.57	26.62	≤1
LIBSVM	85.71	0.38	97.39	85.71	85.32	≤1

Is possible to see that even if you have a similar value of Accuracy the FPR (false positive rate) changes a lot.

In binary metrics, detected change points are labelled as positive if they correspond to ground truth points (detected or missed).

In anomaly detection, labels are nominal or anomaly.

Change Point Metrics

- **ADD** (Average Detection Delay) **= MAE** (Mean Absolute Error) **= Annotation Error:** difference between the detected time and next time for change points.

$$\text{MAE} = \frac{\sum_{i=1}^{\#CP} |\text{predicted}(CP) - \text{actual}(CP)|}{\#CP}$$

- **MSD (Mean Signed Difference)**:

$$\text{MSD} = \frac{\sum_{i=1}^{\#CP} (\text{predicted}(CP) - \text{actual}(CP))}{\#CP}$$

- **MSE (Mean Square Error):**

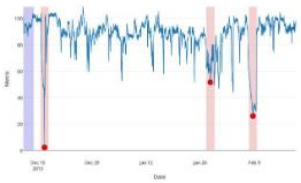
$$\text{MSE} = \frac{\sum_{i=1}^{\#CP} |\text{predicted}(CP) - \text{actual}(CP)|^2}{\#CP}$$

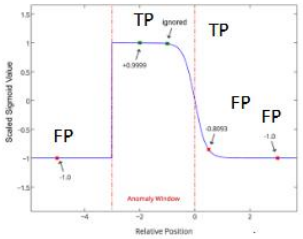
  - **RMSE (root MSE):** $\text{RMSE} = \sqrt{\text{MSE}}$
  - **NRMSE (normalize RMSE):** $\text{NRMSE} = \frac{\text{RMSE}}{\text{maxLength}(\text{actual}(CP)) - \text{minLength}(\text{actual}(CP))}$

- **ADD**: at a particular alarm probability
- **Hausdorff:** Biggest interval between detected CP and its actual value.

Window-Based Metrics

Binary methods do not encorporate time and do not reward early detection: not useful for real-time socring -> window-based scoring, Evaluate if anomaly falls within a window. What's the meaning of "not reward early detection"? That's mean that if I'm being able to detect an anomaly earlier (without processing full sequence) than you will basically evaluate*(credo)* a value. So you have to include this evaluation, for this reason we have NAB.





- **NAB scoring algorithm (Numenta Anomaly Benchmark):** Used for real-time anomaly detection; rewards early and correct detection and penalizes for false positives and false negatives.

  NAB Formula: $\sigma^A(y) = (A_{TP} - A_{FP}) \cdot \frac{1}{1+e^{5y}} - 1$
- **RandIndex:** Indicates that two (predicted and ground truth) segmentations either agree or do not agree on pairs of points. (declined for images or multidimensional signal, in this case we use segmentation metrics).

Performance Tuning

How to tune the performance? Depending on which are the choices that you want for your threshold:

- Automatize choice but leave some configuration capabilities.
- Avoid "false alarm black hole" (too many alarms).
- Allow false negatives to reduce the false positives.

Learning Conditions

How do you train this algorithm? In this way:

Supervised Learning

- Here we have a dataset where the sample are Labeled (some are anomalous some no), then I give this dataset to the algorithm and it distinguish by itself.
- **Problems:**
  - Classes are unbalanced (if data are high imbalance the bigger classes "going to dominate").
  - recall is more important than accuracy (catching all anomalies is more important than mislabeling nominals).
  - The classifier is very stick to the specific dataset that it was used to train the data, (if you have different condition from the dataset it not working anymore) so we want update but it's difficult. (this is continual learning – what we do? We retrain on the new data, but this brings another problem: the catastrophic forgetting, it forget about the olds one)

Unsupervised Learning

- Both training and testing have both classes, no labels. So you give to the classifier the data (mixed), the classifier looks for correlation between groups of data but in the end you get two clusters of information but you don't kwon which are the anomaly and which are not (in reality you know). Clustering separates data in different group.
- **Implicit assumption:** Nominals are "clustered", anomalies are not.
- **Problems:**
  - Normal objects may not share strong patterns, but outliers might in a small area.
  - overclustering, high false positive rate and actual outliers undetected.

Semi-supervised Learning

- Labeled examples and proximate unlabeled objects used to train a model.
- **Clean data:** Training set consists only of nominal points; test set is contaminated with anomalies. Clean data cannot be seen as unsupervised (*credo*).
- **Learning from positive and unlabeled examples (LPUE):** (mix between supervised and clean data) Two-class problems but only data from one class (positive) is available with unlabeled data. (so we have nominal data + unlabeled data).
- **Problems:** only some labeled outliers are available, a small number of labeled outliers may not cover the possible range of anomalies. (that's mean that this is not will cover all the possible attacks or anomaly)

MACHINE LEARNING AND DEEP LEARNING

Traditional Learning Strategies:

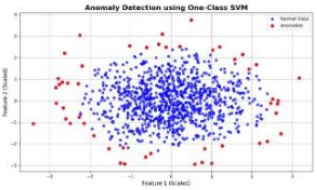
- K-means clustering, Spectral clustering.
- Decision trees (Random Forest).
- Support Vector Machine (One-Class SVM).
- Self-Organizing Map (SOM), Growing Neural Gases (GNG).

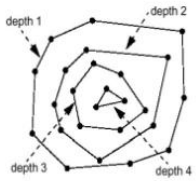
Deep Learning:

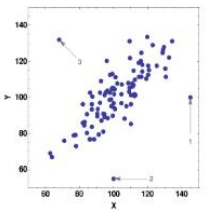
- Fully-Connected networks.
- Convolutional Neural Networks (CNN).
- Recurrent Neural Networks (RNN) - Long-Short Term Memory (LSTM).
- Generative Adversarial Networks (GAN).
- One-Shot or Few-Shot Learning architectures.
- Transformer networks.

CLASSES FOR CLASSICAL AD ALGORITHMS

- **Statistical/Model-based methods:** Normal data follows a statistical model (to be learned); identify objects in low probability regions. (gives a probability to an anomaly, low probability value are basically anomaly) (we can see in the image on the side that the blue dots are nominal data and the red ones are anomalous data – they go far from the center)
- **Density-based methods:** object is then considered an outlier if its neighborhood does not have enough other points. (where a point is isolated we can say that is an anomaly)
- **Deviation-based:** Variance changes when points are removed.
- **Neighbor/Cluster-based methods:** Anomalies are points far away; different pattern.
- **Projection-based methods:** Project points to reduce dimensionality.
- **Bayesian Networks:** Exploit relational statistics.
- **Neural Networks:** Estimate complex patterns.





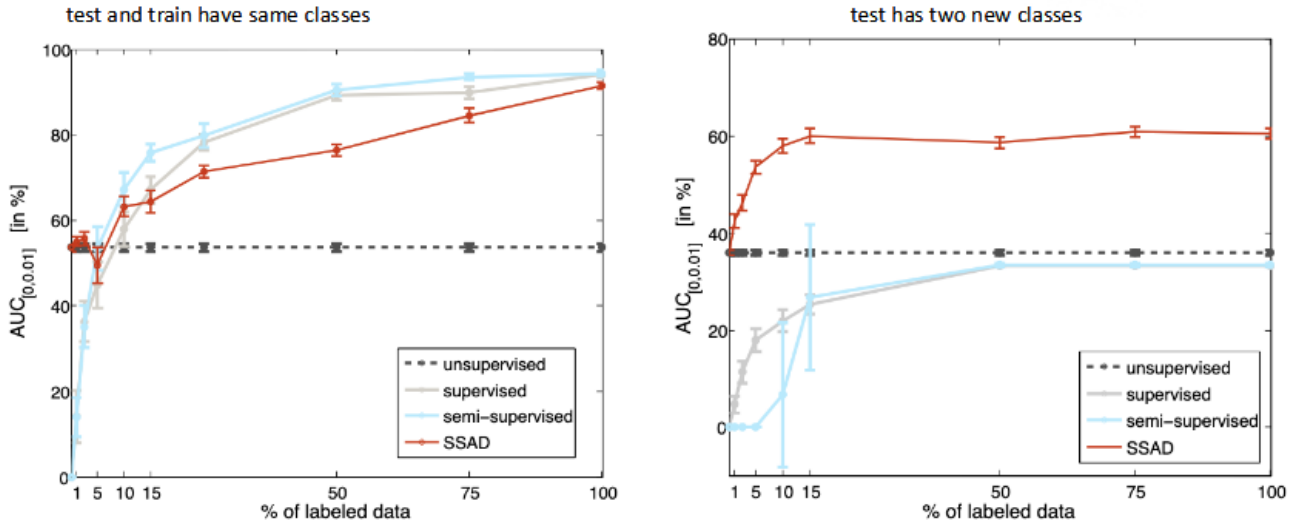


SUPERVISED LEARNING FOR ANOMALY DETECTION

- Set of data (x, y) where label is "normal" or "anomalous".
- Data collected from history to predict future problems (e.g., traffic ok vs network under attack).
- Nominal state of networks, equipment, … can be predicted from measurements, model, simulations, data.
- Data acts as a proxy to train a predictor model (regression).
- **Processing methods:**
  - Logistic/linear regression,
  - SVM,
  - Classification trees,
  - Deep Neural Networks,
  - Gaussian processes.

A quick comparison of methods

- Supervised and semi-supervised have better performances
- It depends on whether data are available
- On the left (below) we can see that the supervised and the semi-supervised are basically the same.
- On the right side instead we can see that semi-supervised and supervised still perform better even though the number of sample is lower.
- Supposed now that the test set has two new classes, that nobody as seen before. The unsupervised keeps here instead the supervised and semi-supervised collapse, for this reason we introduce SSAD (more robust in case of new element).



Images from [Go13]

UNSUPERVISED LEARNING FOR ANOMALY DETECTION

Find structures in data when no labeling is available.

- Avoid the need for labelling (expensive) (this in case of lots of data)
- Define patterns in data → outliers = samples that do not fit to such patterns
- **General idea:** Build a compressed representation of data; classify anomalies based on reconstruction errors (noise).

- **Techniques:** Clustering (k-means), PCA, SVM, Deep autoencoder, Filtering.

Theoretical Aspects of unsupervised AD:

- Let be $\boldsymbol{\alpha}$ the fraction of anomalous samples. (that leaves you with different condition)
- If α is large ($\boldsymbol{\alpha} > 5\%$):
  - Fit a 2-component mixture model
  - requires WDAD assumption
  - mixture models must be well-separated and identifiable
- If α is small (e.g. $\boldsymbol{\alpha}$ = 1, 0.1, 0.001 … %):
  - Use outlier detection
  - does not require WDAD (because WDAD implies that you are able to estimate the statistics of the nominal data, but if you have only a few nominal data you can't estimate statistics in a reliable way)
  - the idea is to find out anomalies as outliers.
  - fails if anomalies ≠ outliers (e.g. anomalies overlap nominal density, outliers are tightly clustered = another cluster) or if nominal distribution has heavy tails.

In any case, most of the AD solutions are related to density estimation.
Performances are dependent on how clustering performs: sometimes clustering errors happen.

SLIDING WINDOW CLUSTERING

You take your signals and dived it in intervals and then you cluster each and defined some prototype. Your signal is basically composed of those specifics "alphabets". Then you basically classify each segment, and when you have that a segment is not very well map by the dataset then you classify as anomaly (*credo*).

1) Divide signal into interval using a sliding window
2) Cluster segments and define prototypes
3) Classify

DBSCAN (Clustering)

Density-Based Spatial Clustering of Applications with Noise.

- Clusters points together and identifies points not belonging to a cluster as outliers.
- Does not require specifying the number of clusters (K).

Procedure:

1. Randomly select an unvisited point $x_i \neq$ outlier $\notin$ cluster (*unvisited*).
2. Label it as visited.
3. Check if it is a **core point**: at least *min_samples* points around it within distance. (the must be at least a value of sample, which is defined by this threshold *min_samples*, around this point so that the distance of this sample points is lower)
4. You must have at least *min_samples* points around x_i, if:
   - **No:** then x_i is noise.
   - **Yes:**
     1. Create a cluster C_i with all points with $d < \epsilon$ (all directly-reachable points).
     2. For each point $x_j \in C_i$ that is not visited
        I. Label it as *visited*
        II. Find all points x_k s.t. $d(x_k, x_j) < \epsilon$

III. add these to C_i if at least min_samples points x_j

3. For each point $x_j \in C_i$ visited (but not in cluster)
   I. Add it to C_i

CLEAN DATA LEARNING

Used if system behavior can be predicted.

- **Training:** Clean nominal data only.
- **Testing:** Both nominal and anomalies.

- **Forward projection**
  - Past observations can be used to predict the future
  - If the data divert strongly from the predictions you can have detect an anomaly
  - Already employed in ARMA models (Auto-Regressive Moving Average).
  - Recurrent Neural Networks
- **Physical models to predict behavior**
  - Engineering approach
  - Partial Differential Equations, dynamical systems
  - Circuit or network simulation
  - Uncertainty quantification

DECISION TREE

Classification performed by progressive separation using different features.

Given a single split nodes characterized by , define the impurity metric:

- **Impurity metric:** $\boldsymbol{\sigma} = \langle \boldsymbol{i}, \boldsymbol{\theta} \rangle$ (where i is feature index and $\boldsymbol{\theta}$ is threshold).
- **Energy minimization:** $I(D, \sigma) = \frac{N_{left}}{N_D} I(D_{left}) + \frac{N_{right}}{N_D} I(D_{right})$

Find $\boldsymbol{\theta}$ that minimize the energy while splitting the node in the tree.

This can be represented by a tree where I consider as root the value x_1 and if the value is higher going the right if is lower go to the left and so on. If I receive a new sample I cross the tree following this rule and place the new sample in the correct spot.

The "**Impurity metric**" tells you how much heterogeneous the data in the subset are, and therefore you want to find a threshold (for splitting) so that the impurity of the subtree is minimized. The impurity of the tree is the combination of the two impurity of the left-subtree and the impurity of the right-subtree weighted by the number of samples for each subtree.

ISOLATION TREE

Easier to identify anomalies because they require fewer separations (shorter path length).

- **Fully random binary tree**:
  1. Choose attribute at random.
  2. Choose $\theta \in [\min x_i, \max x_i]$ threshold .
  3. Repeat 1,2 until every point is in a leaf. $d(x_i)$ is the depth of leaf x_i

  Randomly dived your data used threshold requires more or less "cuts" depending if the point that I want is an outlier or not. For outlier I need few separation instead nominal one required a lots.

So, performing random separation we create a tree where each leaf is one sample. In this tree the highest leaves will be outliers and the deepest leaves will be nominal data.

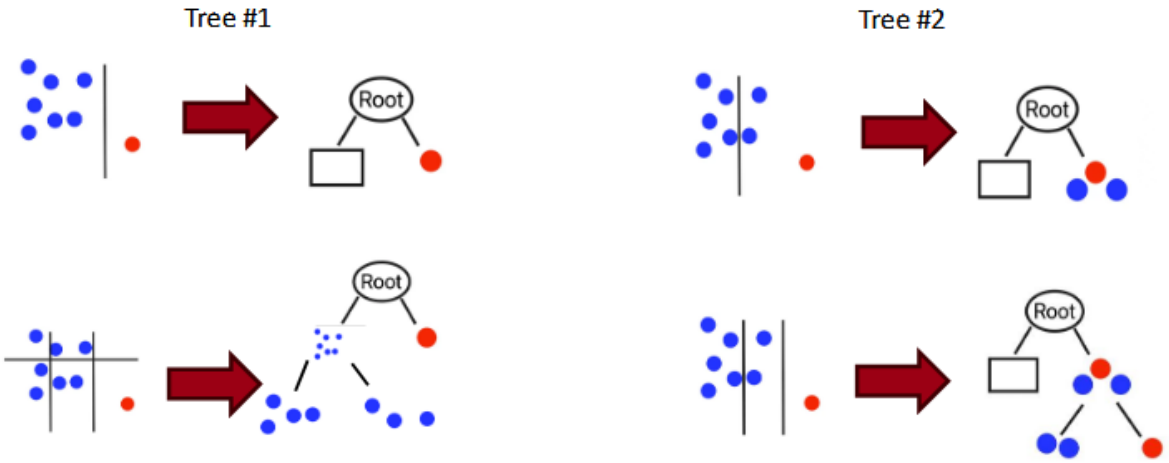
You repeat this until you will have a forest of trees where you can compute the average depth of every sample. (IF)

- Anomaly:
  a) Compute $\bar{d}(x_i)$ average depth of x_i
  b) $r(x_i)$ is the expected depth
  c) Compute the score: $score(x_i) = 2^{\frac{\bar{d}(x_i)}{r(x_i)}} > 0.5$ anomaly
- All samples with score close to 0.5 = no anomalies
- Sample with score close to 1 = anomaly
- Score lower than 0.5 = nominal

ISOLATION FOREST

Randomly split data

- It is easier to identify anormalities … less separations!



RECAP ON IF

a) Compute $\bar{d}(x_i)$ average depth of x_i

b) $r(x_i)$ is the expected depth

c) Compute the score: $score(x_i) = 2^{\frac{\bar{d}(x_i)}{r(x_i)}} > 0.5$ anomaly

$$r(x_i) = \begin{cases} 2H(m-1) - 2\frac{m-1}{n} & if\ m > 2 \\ 1 & m = 2 \\ 0 & otherwise \end{cases}$$

with *H(i)* harmonic number estimated as $H(i) = \log(i) + \gamma$ ($\gamma = 0.57$ Euler-Mascheroni constant) and *m* equal to sample set size (n = test set size)

LECTURE 14 (DEEP AD)

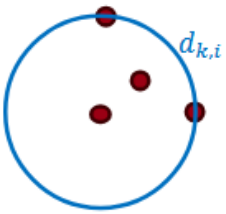
LOCAL OUTLIER FACTOR (LOF)

LOF is an algorithm that focuses on density. I consider a node as outlier if it has not enough neighbor around.

For each sample x_i

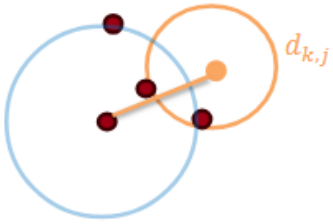
1. Compute distance of x_i with respect to any other point
2. find the k-th closest point $x_{K,i}$ and the k-closest distance s.t. $d < d_{K,i} = d(x_{K,i}, x_i)$
   = define the set $N_k(x_i)$ (the number of points s.t. there are near (*credo*))
3. compute the Local Reachability Density (LRD) e.g., $\mathrm{LRD}_k(x_i) = \frac{1}{E_{N_k(x_i)}[RD(x_i,x_j)]}$: how far we have to travel to reach the next point or cluster (is basically 1/average distance (*credo*))
4. compute the local outlier factor $\mathrm{LOF}(x_i) = \frac{E_{N_k(x_i)}[\mathrm{LRD}_k(x_t)]_t}{\mathrm{LRD}_k(x_i)}$

k=2 $N_k(x_i)$=3



Reachability density (RD)

$\mathrm{RD}(x_i, x_j) = \max(d_{k,j}, d(x_i, x_j))$



Local Reachability density (LRD)

$$\mathrm{LRD_k}(x_i) = \frac{1}{\frac{1}{|N_k(x_i)|}\sum_{x_j \in N_k(x_i)} RD(x_i, x_j)}$$

Low values of LRD implies that the closest cluster is far from the point.

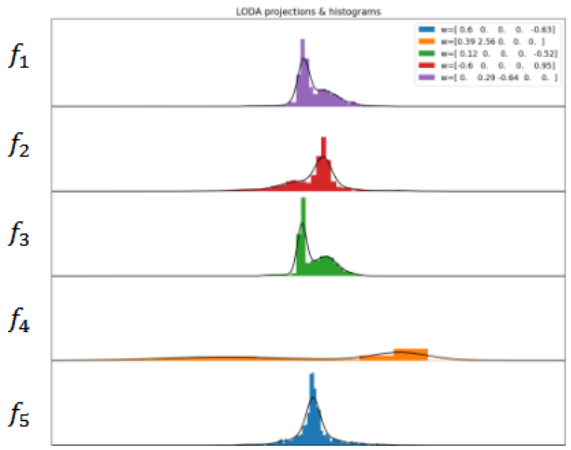
LOF > 1 **outlier!**

So if you have a node with low reachability (it means it's far from the other) but you have also that all these other nodes have low reachability well means that we don't have an anomaly but we have a graph really scattered. But if you have some nodes with high reachability and one node with low reachability that means you have an outlier. So the LOF is a number that compares the reachability of the neighbors with the reachability of the node and tells you if the node is an outlier or not.

LODA

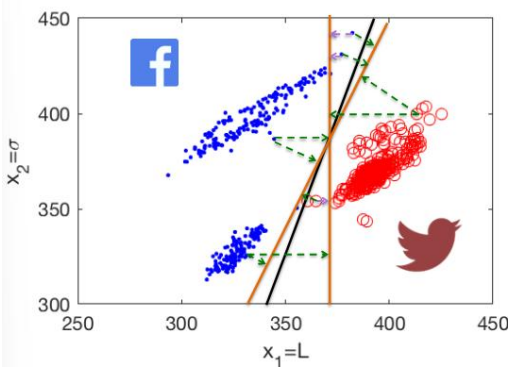
You have that your data are in a multidimensional space, so you project this data in a smaller space. You do this projection randomly.

- Compute a set of random projections $\Pi_1, \Pi_2, \ldots, \Pi_k$
- Computes $f_1, f_2, \ldots, f_k$ a set of 1-dimensional density estimator on L2 distance for projections.
- For some projections you have that some statistics you have high probability and for other projections instead are flatter. For this reason we do the Average surprise: $S(x) = \frac{1}{k}\sum_m -\log f_m(x)$
  a) If f_m very high (close to 1) → no surprise
  b) If f_m very low (close to 0) → high surprise
- Outliers can be outliers for some 1-D densities



- LODA has a built-in way to get a little bit more information
- one-tailed two-sample t-test between probabilities with and without a specific features.

THE ROAD TO SVM - OPTIMAL SEPARATOR



Maximize the average distance of points from the hyperplane!

$$w = \frac{|ax_1 + bx_2 + c|}{\sqrt{a^2 + b^2}} \lessgtr 0$$

-->positive (correct classification)

-->negative (wrong classification)

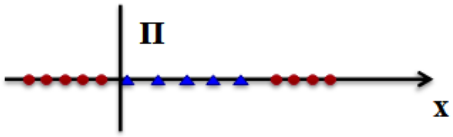
The task is to optimize the parameters (a,b,c) so that all the red have positive value and all blue points have negative value.

Summing all the distances, we can get a measurement of the suitability of the classifier.

It is possible to use a non linear separator.

LINEARITY DOES NOT WORK ALL THE TIME!

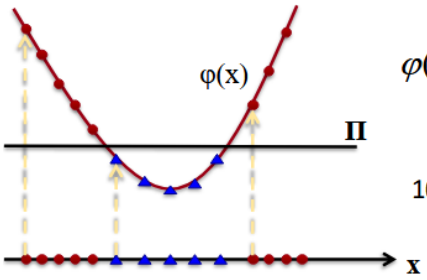
Example: in R^1 (1 dimension) the hyperplane (the linear separator) is a threshold! How do you define an optimal threshold?



66% accuracy. Poor performance!

One dimensions it's not well suite to separated the red point with the blue point, so you do an extension.

To improve things, we can map the feature vector into a new space using a non-linear function.



$$\varphi(x) = (x - m)^2$$

100% accuracy

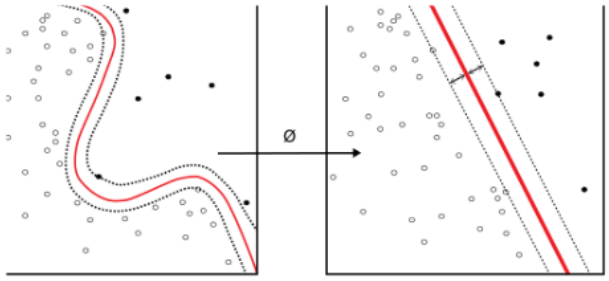
So you have more accuracy simply expanding the space. (kernel base methods)

Similarly for R2 feature vectors...

SUPPORT VECTOR MACHINE

The idea is to have one kernel function such that the data, in original, are not very separable but if perform the separation in $\varphi(x_i)$ become easy separatable.

- Supervised learning
- Based on probabilistic linear classifiers computed from a set of training data organized into **arrays** of **features**
- If partitioning is simple, no problem; otherwise, **kernel trick**

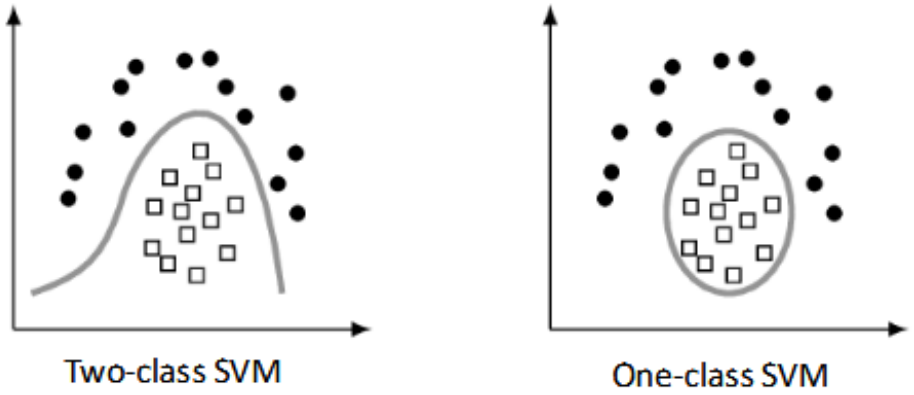


Kernel: it is associated to a transform $\varphi(x_i)$ of the input array x into a different space (typically with higher dimensionality, where data separation is easier).

$$k(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$$

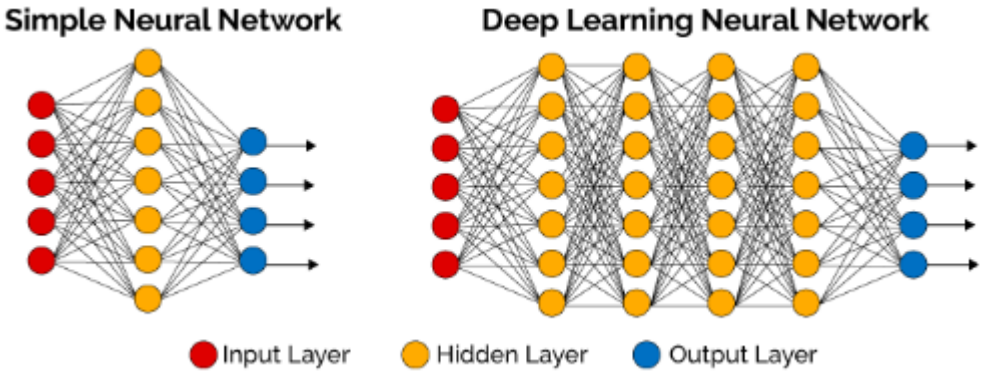
ONE-CLASS SVM

- In AD sometimes we have **clean data** condition, it's means we have only one class (only nominal data).
- Instead of using a hyperplane to separate two classes of instances, use a hypersphere to encompass all of the instances. We encapsulate the nominal data without knowing the anomalies. Leaving the anomaly "outside".
- "the largest possible margin" = "the smallest possible hypersphere".



DEEP LEARNING IN A NUTSHELL (1/3)

- Function approximation techniques
- More flexible than traditional numerical methods
- Need to encode the problem in an error function
- Differentiability is required in order to train it by Stochastic Gradient Descent (SGD)



DEEP LEARNING IN A NUTSHELL (2/3)

- Composing multiple stages = complex function (combine the neuron – (*credo*))
- Different non linear activation functions $a(\cdot)$ are possible. (simple threshold = step)
- Multidimensional output is possible

DEEP LEARNING IN A NUTSHELL (3/3)

The learning process:

- Collect inputs and outputs — dataset $(\boldsymbol{x_i}, y_i), i = 0, \dots, n$
- Define the space of allowable functions — function $y = f_\theta(x). \quad \theta \in \Theta$
- Define a metrics to evaluate the fitness of the function — loss $J(\theta) = ||y_i - f_\theta(x_i)||$
- Use calculus to fit function parameters to data — best option $f_\theta^* = \arg\max_\theta J(\theta)$
- Landscape defined by the -> walk to valley error function — **Training:** $\theta \leftarrow \theta - \alpha \nabla J(\theta) \quad \forall (x_i, y_i)$
- After training, hope that the results fit the new data — Hope $f_\theta(\hat{x})$ fits $\hat{y}$ (new data)

It's an iterative method, if you have a local minimal you are in "trouble". (*credo*)

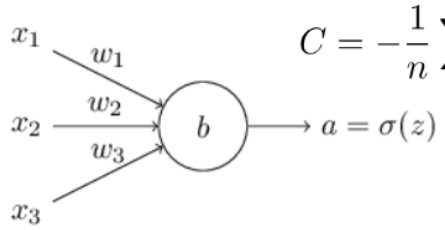
SIMPLE EXAMPLE: CROSS-ENTROPY FUNCTION (1/2)

MSE classification $C = -\frac{1}{n}\sum_i || y_i - a(x_i)||^2$

It may happen that learning speed goes down … the learning is too slow.

How can we address the learning slowdown?

quadratic cost -> cross-entropy. (very use in classification problem)



$$C = -\frac{1}{n}\sum_i y_i \ln(\mathbf{a}(\mathbf{x_i})) + (1 - y_i)\ln(1 - a(\mathbf{x_i}))$$

$$a(\mathbf{x_i}) = \sigma(\mathbf{w}^T \mathbf{x}_i + b)$$

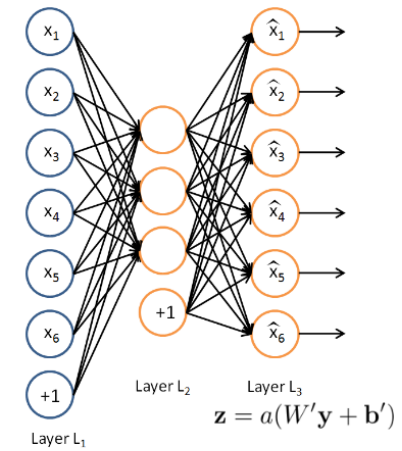
Partial derivatives then becomes

$$\frac{\partial C}{\partial w_j} = -\frac{1}{n}\sum_x \left(\frac{y}{\sigma(z)} - \frac{(1-y)}{1-\sigma(z)}\right)\frac{\partial \sigma}{\partial w_j} = -\frac{1}{n}\sum_x \left(\frac{y}{\sigma(z)} - \frac{(1-y)}{1-\sigma(z)}\right)\sigma'(z)x_j$$

It can be generalized to all the neurons in a network! Error defines the learning speed!

$$C = -\frac{1}{n}\sum_x \sum_j \left[y_j \ln a_j^L + (1 - y_j)\ln(1 - a_j^L)\right]$$

SIMPLE EXAMPLE: AUTOENCODER (MSE LOSS)



$W' = W^T$

The idea is to use this type of network to reconstruct *something*

Minimize the distortion or cost function

$$L(x, z) = |x - z|^2$$

or

$$L(x, z) = -\sum_{k=1}^d x_k \log z_k + (1 - x_k)\log(1 - z_k)$$

Basically we are solving a reconstruction problem.

Hidden layer define a set of features for the input data

Can be used to denoise input-data: exploits the regularity of statistics.

WHISHLIST FOR DEEP LEARNING IN AD

a. Low recall rate

- anomalies are usually rare and heterogeneous -> difficult to characterize
- high false positive rate

b. high-dimensional and/or not-independent data

- anomalies can become hidden or not noticeable in high-dimensional space
- DL for feature reduction (select automatically feature that tells if certain data are irrelevant or not (*credo*))
- build features using spatial, temporal or other correlations

c. data-efficient learning of normality/abnormality

- weak- supervision or semi-supervision is better
- DL can integrate few labelled samples in the learning process

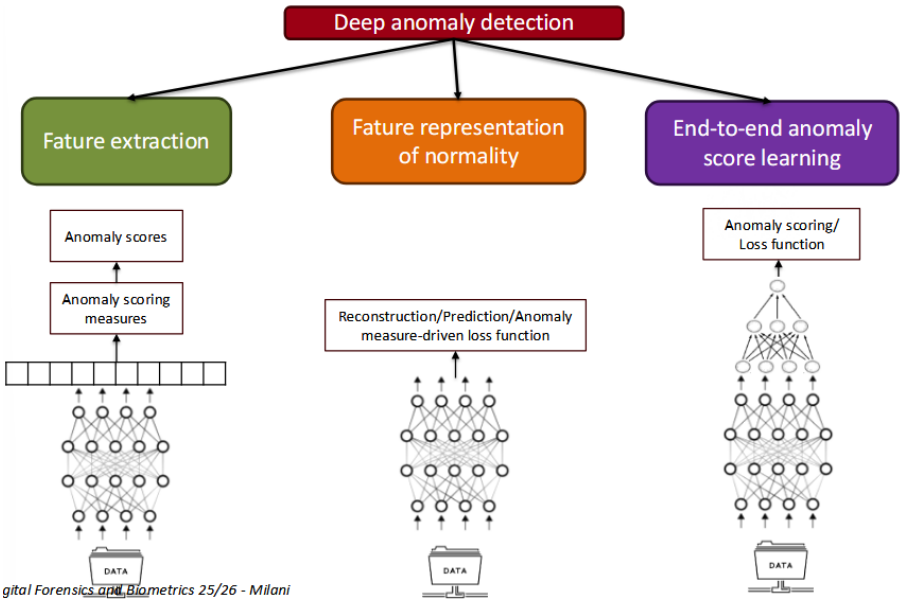
d. Noise-resiliency

- learn noise patterns (could be seen as anomaly)
- you can use strategies that identify the anomaly and at the same time clean out the noise

e. Detection of complex anomalies

f. Explainability (explain why the function is an anomaly)

DEEP ANOMALY DETECTION TAXONOMY



In deep AD we have 3 main classes:

- based on **Features extraction**: the idea is to take the input data, used deep learning to create some good representation of your feature data so than we can build some classic AD method.
  The end is classical but the core is build with deep learning.
- **Features representation of normality**: in this case we want a feature that represent what is a nominal behavior for your phenomenon.
- **End-to-end anomaly score learning**: networks that output an anomaly score directly and given this score you want to minimize the loss function between the anomaly score estimate by the network and the anomaly score provided by real data.

Deep anomaly detection:

- **Feature extraction**
- **Feature representation of normality**
    - **Generic normality features**
        - Autoencoders
        - GANs
        - Predictability modelling
        - Self-supervised classification
    - **Measure-dependent features**
        - Distance-based measures
        - Clustering-based measures
        - One-class classification measures
- **End-to-end anomaly score learning**
    - Ranking modes
    - Prior-driven modes
    - Softmax likelihood modes
    - End-to-end One-class

LECTURE 16 (DEEP AND XPLAINABLE AD)

FEATURE EXTRACTION

Feature extraction and anomaly scoring are independent

Feature extraction as dimensionality reduction only $f = \phi(x; \theta)$ (x is the input, θ are the parameter of the network). Network can be seen as function

Rather than classical methods, better capacity to extract significant features

- use backbone models (VGG, ResNet, AlexNet)
- unmasking framework for online anomaly detection (iteratively train classifier to separate significant features, sample video on significant regions)
- transfer learning (features representation trained on a different dataset)

Features can be classified with traditional schemes (e.g. one-class SVM)

The idea is to create a classification that is more useful than the input data that you process.

PROS AND CONS FOR DL FEATURE EXTRACTION

Pros:

- Large number of models
- DL features more powerful and more effective than traditional methods for dimensionality reduction
- Easy to implement

Cons:

- Feature creation and anomaly scoring are disjoint (you may be discarding useful information) (not necessarily the feature classification and the feature creation are jointed)
- Following the fidelity principle, I create a latent representation that is capable of reconstruction. However, this representation may not preserve fidelity when used for classification.
- Pre-trained models are limited to specific data types.

Anyway, dimensionality reduction

- reduce false positives and reveals hidden anomalies
- multiple types of features can be integrated

GENERIC NORMALITY FEATURES

LEARNING NOMINALITY

Learn the nominal value, you do it using the autoencoder(*credo*) structure. Optimizing a generic feature learning objective function that is not primarily designed for anomaly detection, but empowers anomaly detection

$$\{W^*, \Theta^*\} = \arg\min_{W,\Theta} \sum_{x \in X} l\big(\psi(\phi(x; \Theta); W)\big)$$

$$s_x = f(x; \psi_{W^*}, \phi_{\Theta^*})$$

ϕ: maps data into a latent representation (parameters Θ)

ψ: surrogate learning that enforces learning regularities (parameters W)

f: scoring function

Given the autoencoder structure you can find out f (the scoring function) that given two vectors it gives to x a real value.

- Data reconstruction
- Generative modelling
- Predictability
- Self-supervised classification

The idea: create a feature space so that you have that the data mapped into the space itself.

DEEP AUTOENCODER

Latest clustering approach can resort to DNN structures

Compressed data = feature vector in the latent space

Training phase: Autoencoder maps nominal samples into a feature space; representation is learned so that samples are mapped into well clustered regions. (**clean data set-up**)

Monitoring phase: Autoencoder is applied on the data The reconstruction error is a proxy for anomalies (something not seen before). So if you have an outlier you have an object that not appear on the training set. So if this object is compressed and reconstructed, the reconstruction will have a low high distribution (*credo*). In this case the reconstruction error is my anomaly function.

ASSUMPTION: Normal instances can be better restructured than anomalies.

Some examples: Convolutional, U-Net, Adversarial Autoencoder

AE FOR ANOMALY DETECTION

Given encoding and decoding networks, data reconstruction error can therefore be directly used as anomaly score.

encoder $\boldsymbol{z} = \phi_e(\boldsymbol{x}\,; \Theta_e)$ decoder $\hat{\boldsymbol{x}} = \phi_d(\boldsymbol{z}\,; \Theta_d)$

$$\{\Theta_e^*, \Theta_d^*\} = \arg\min_{\Theta_e, \Theta_d} \sum_{\boldsymbol{x} \in X} \|x - \phi_d(\phi_e(\boldsymbol{x}; \Theta_e); \Theta_d)\|^2$$

scoring $s_{\boldsymbol{x}} = \|x - \phi_d(\phi_e(\boldsymbol{x}; \Theta_e^*); \Theta_d^*)\|$? $< T$ Yes → nominal; No → anomaly

Several structures and data are available:

- Sparse AE: *encourage sparsity in activation*
- Denoising AE: *for small variations*
- CNN AE: *for correlated data*
- Contractive or Variational AE: *to regularize and avoid noisy data*
- LSTM-AE: *time-dependency*
- Graph convolutional AE: *spatial relation on graph*

ANOMALY DETECTING AUTOENCODER

Setting a threshold on the MSE detect the anomaly

AE FOR ANOMALY DETECTION

Used on:

- Tabbed data
- Sequences
- Graphs
- Images and videos

Adapt the type of AE to the input data you are processing

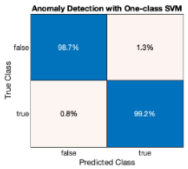
Pros:

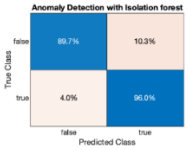
- Easy to implement
- idea is straightforward and generic to different types of data
- Many AE variants are available in literature

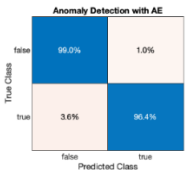
Cons:

- learned features can be biased by infrequent (strange) regularities and the presence of outliers training data
- objective function designed for dimension reduction or data compression – not designed for AD

SOME QUICK COMPARISONS







- Vibration analysis
- Predictive maintenance

'After': nominal

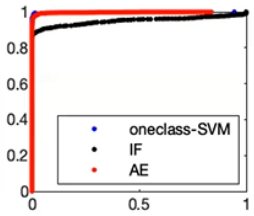
'Before': Anomaly

- Computed on a set of feat

IF = linear

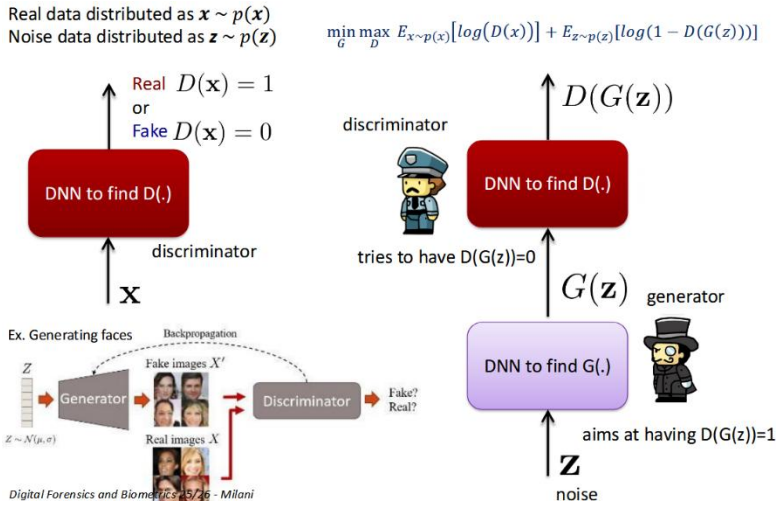
SVM = rbf

AE 2 layers



The look that sometimes the classical ML performs better than DL. This happened when you have "tap data" (heterogeneous data gather together in an single feature array). So very different feature, that are converted in numeric value, that creates discontinuity. This creates a problem especially for autoencoder because they assume that there is a correlation between samples, and then if they don't find one, they create one. So what are they done? They smooth the data.

GENERATIVE ADVERSARIAL NETWORKS (GANS)



GAN was used for generated synthetic images. GAN are couples of network, one network (**generator**) create the images, the other network (**discriminator**) receives the image and output the decision if the image is fake or not. So we start from a dataset of real images, then we generates fake images and we feed the discriminator with both; the discriminator has to learn the difference between fake and real images. Then you repeat for upgrade the generator and so on.

GENERATIVE ADVERSARIAL NETWORKS FOR AD

Being G and D the generator and the discriminator networks, you want

$$\min_G \max_D V(D,G) = E_{x\sim p(x)}\left[\log\left(D(x)\right)\right] + E_{z\sim p(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right]$$

V is the value function of two-players minmax

The discriminator wants to have this close to 1, instead wants to have its close to 1.

How to use this for AD? Search the space looking for the right noise z at different iterations *t=1,…,T*.

Two losses are used to guide the search: **discrimination** loss and **residual loss**.

$l_R(x, z_\gamma) = ||x - G(z_t)||_1$

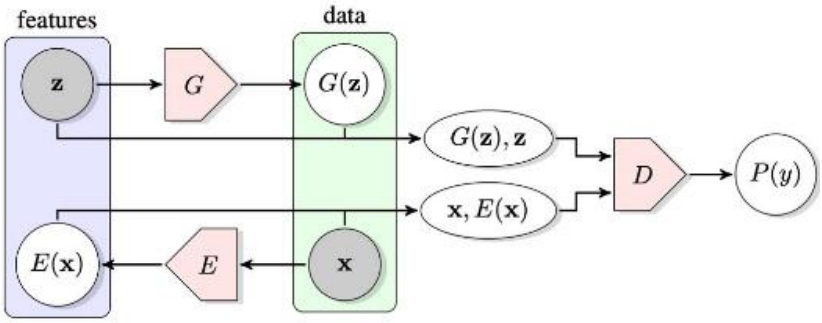
$l_D(x, z_\gamma) = ||h(x) - h\left(G(z_t)\right)||_1$. with *h* feature mapping

Start with random z then update so that $(1-\alpha)\cdot l_R(x,z_t) + \alpha \cdot l_D(x,z_t)$ is minimized.

Final scoring $s_x = (1-\alpha)\cdot l_R(x,z_t) + \alpha \cdot l_D(x,z_t)$

GENERATIVE ADVERSARIAL NETWORKS FOR AD

- **AnoGAN** standard GAN scoring
- **EBGAN** use inverse GAN for search, encoder maps x to z



This type of structure allow us to detected anomalies, because you have loss → the loss provide you with an anomaly score → this must lower than a threshold, when is higher you have the anomaly.

Better search since you want to discriminate (x,E(x)) from (z,E(z)). → D(x,z)

$s_x = (1-\alpha)\cdot l_R(x,z_t) + \alpha \cdot l_D(x,z_t)$

$l_R(x, z_\gamma) = ||x - G(z_t)||_1$

$l_D(x, z_\gamma) = ||h(x) - h\left(G(z_t)\right)||_1$

- **ALAD** more than one discriminator
- **GANomaly** end-to-end training $x \to z \to \hat{x} \to \hat{z}$
- Employ different GAN architectures: CycleGAN, WGAN

PROS AND CONS FOR GAN'S USE IN AD

Pros

- Good capacity to create realistic instances
- Many GAN models exist and can be adapted

Cons

- Difficult to train (failure to converge and mode collapse)
- generator network can create data instances out of the manifold of normal instances (complex distributions)
- Anomaly detection is built on generator (optimized for creation; suboptimal for AD)

PREDICTABILITY-BASED

Idea: predict current instances from previous ones within a temporal window as context.

e.g. instances = video frames in a sequence.
The network grasps dependencies: nominal instances normally adhere to such dependencies well, whereas anomalies often violate them.
The idea: if you have a nominal data (with a specific behavior) and if this behavior has correlation you can predict the next value.
Largely used for video: motion dependencies are difficult to characterize.

PREDICTABILITY-BASED
Idea: predict current instances from previous ones within a temporal window as context.

$$\hat{\boldsymbol{x}}_{t+1} = \psi(\phi(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \dots, \boldsymbol{x}_{t-N}; \Theta); \mathrm{W}) \qquad (1-\alpha) \cdot \ell_{pred}(\hat{\boldsymbol{x}}_{t+1}, \boldsymbol{x}_{t+1}) + \alpha \cdot \ell_{adv}(\hat{\boldsymbol{x}}_{t+1})$$

network (e.g. U-Net) *loss*

Anomaly score $s_{x_{t+1}} = ||x_{t+1} - \hat{x}_{y+1}||$
Uses in video alarm detection. Detection can be improved by:

- Autoencoder-based
- Autoregressive models.
- Introducing probability (*credo*)

$z_t = \phi(x_t; \Theta)$. $p(z) = \prod_{j=1,\dots,T} p(z_j | z_{1:j})$
Loss depends on reconstruction and probability as well

PREDICTABILITY-BASED: PROS AND CONS
Pros

- Several methods for sequence learning are available
- Allows to grasp both spatial and temporal dependencies

Cons

- Limited to sequence data (no tabbed) – (you need something high correlated)
- Predictions could be computationally-expensive
- Suboptimality (like previous ones; same reasons) (this method intend for predictability not necessarily for AD)

SELF-SUPERVISED CLASSIFICATION
Idea: build a self-supervised classification model; instances that are inconsistent to the classification model are anomalies. Classifiers are drawn to compute features that are relevant to classification: nominal instances are more consistent to self-supervised classifiers than anomalies. Nominal instances consistent → anomaly detection work.
Scoring can be based on

- prediction error
- majority voting of binary decision (for classification over all the features)
- log loss surprisal

One of the first approaches [Izh18]: create a classifier from transformed data. Then augment test instances and apply the classifier.

- Gradients for nominal are larger than for anomalies

- stronger agreement for nominal

SELF-SUPERVISED CLASSIFICATION

Composing multiple classifiers:

- Majority voting or average
- Using density (statistical methods)
- Entropy

SELF-SUPERVISED CLASSIFICATION: PROS AND CONS

Pros:

- Works for unsupervised and semi-supervised as well
- Score is somehow related to properties of gradients (relative variation)

Cons

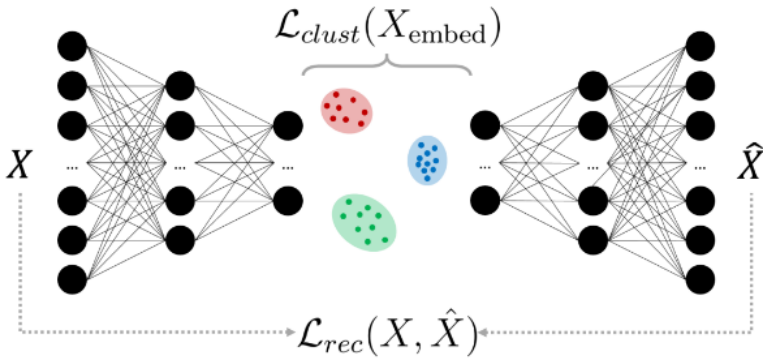
- Feature transformations are usually data-dependent (rotation and flipping are possible for images; what if we have audio?)
- Features are learned end-to-end; scores are computed separately. Is this optimal?

MEASURE-DEPENDENT FEATURE LEARNING

Given an anomaly metric, find parameters that maximize the score

$$\{W^*, \Theta^*\} = \arg\min_{W,\Theta} \sum_{\boldsymbol{x} \in X} l\big(f(\phi(\boldsymbol{x}; \Theta); W)\big)$$

$$s_{\boldsymbol{x}} = f(\phi(\boldsymbol{x}; \Theta^*); W^*)$$



Different types of metrics:

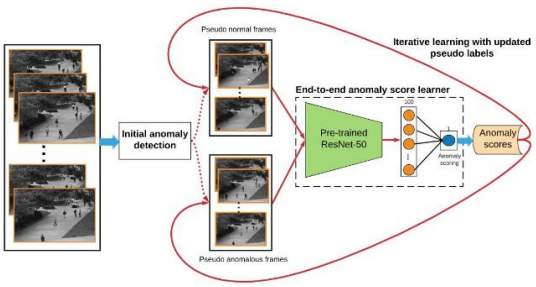
- **Distance-based:** Learn a feature representation that allows detecting anomalies with some distance metrics
- **One-class Classification-based Measure:** Learn a feature representation customized to one-class classification-based methods
- **Clustering-based:** learning representations so that anomalies are clearly deviated from the clusters

END-TO-END ANOMALY SCORE LEARNING

Learn a ranking for the processed data

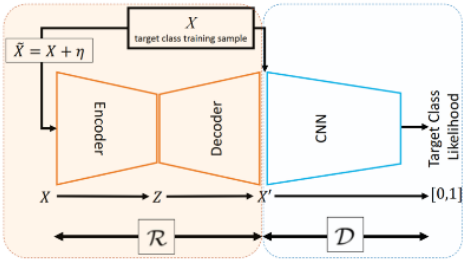
$$\Theta^* = \arg\min_{\Theta} \sum_{\boldsymbol{x} \in X} l\big(\tau(\boldsymbol{x}; \Theta)\big)$$

$$s_{\boldsymbol{x}} = \tau(\boldsymbol{x}; \Theta^*)$$



You give the scoring for each image, video, etc. and then you want the network to predict such score.
Simultaneously-learn features and score Similar to one-class metrics, but they do not inherit the weakness of metrics

- ranking models
- prior-driven models
- softmax likelihood models
- end-to-end one-class classification model



CHALLENGES FOR ANOMALY DETECTION

Challenges:

- Heterogeneous data: high-dimensional, spatial and/or temporal, highly variable
- Difficult to find a general solution (specific to an engineering system)
- Data quality is poor (non-digital, one-off observations)
- Non-stationary datasets
- Large projects = large computational overhead

Deep learning can significantly help, but … Sometimes behave in strange ways (overfitting, lack of adaptation/generalization, …)

OPEN CATEGORY CLASSIFICATION

You have data from an network, first we want verify if there is an attack (anomaly), the second is to classified an anomaly → this is the open category classification.

Problem: Training on classes $\{1, \dots, K\}$. Test of data with news classes? Can we detect them?

EXPLANABLE ANOMALY DETECTION

FEATURE IMPORTANCE FOR ANOMALIES

Sequential Feature Explanation (SFE): (take your array of latency (*credo*) and include one feature at the time, visualize what your accuracy is.

- Expose one feature at a time
- Proper visualization tools

Excluding features and evaluating performance variations

Include analyst feedback: weights different samples in the training

Another way is to perfotm a projection on the features – (example of algorithm is TSNE (*credo*))

TSNE is an algorithm that project your multidimensional arrays into two 2D variables. (it's a projection matrix – this is computed in such a way that the distance between two point on the 2D graph can be related on the original distance of your multidimensional space – the distance must be preserved - so if is very far is an anomaly).

LOCAL AND GLOBAL INTERPRETABILITY

Local interpretability: reasons behind individual predictions.

Global interpretability provides insights into the overall behavior: general rules or patterns that define "normal" versus "anomalous" behavior

Example 1: in a security system of a large company, AD flags suspicious activities in employees' network usage (login times, data access, and file downloads) nominal: logs from 8 a.m. to 6 p.m., download 50 files per day; anomalous: logs at 0:00 and download 200 files. (Global)

Example 2: Barbara logs at 10pm, downloads 100 files and access servers she does not access usually. working late is unusual, but accessing these types of files is common for her role. (Local)

COUNTERFACTUAL EXPLANATIONS

Generate instances that, when perturbed, would no longer be flagged as anomalies (how small a change must be). Understand that if you remove some of the characteristics of those specific elements (features) strange, you get that this anomalous sample will become nominal.

Example 3: Alex logs at 11pm and downloads a lot of files (120): suspicious!

Counterfactual explanation: If Alex had logged in before 8 p.m., his activity would not have been flagged. If Alex had accessed fewer than 50 files, even at 11 p.m., his activity would not have been flagged.

The counterfactuals show how Alex's behavior could have been considered "normal" @Alex: if he wants to work late, download less files @Security Manager: which aspects of Alex behaviors contributes to raise anomaly: modify model

SHAP

(SHapley Additive exPlanations) game theoretic approach that connects optimal credit allocation with local explanations using the classic Shapley values from game theory Divide anomaly score into multiple contributions as distributing gains in a coalition for a game

$g(x) = \phi_0 + \sum_i \phi_i$ where i are the features

SHAP values provide a unified measure of feature importance; reveal the contributions of different features to the identification of anomalies.

Example 4: An employee's login is flagged suspicious Login Time = 3 a.m. -> +0.5 to the anomaly score. Number of Files Accessed = 200 -> +0.7 to the anomaly score. Type of Server Accessed = Sensitive -> +0.3 to the anomaly score.

SHAP PROPERTIES

- **Efficiency:** the sum of SHAP values for each feature is equal to the final total anomaly score
- **Symmetry:** two features contribute equally, they receive the same Shapley value
- **Dummy:** if a feature does not change the final anomaly score, its Shapley value is 0 (useless)
- **Additivity:** Shapley value of two features is the sum of their individual Shapley value

Given an instance x and a model f, we have SHAP value ϕ_i for feature i

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!\ (n - |S| - 1)!}{n!} [f_x(S \cup \{i\}) - f_x(S)]$$

S: subset of features without i

$f_x(S)$: anomaly score computed on x with all features $\notin$ S marginalized (0s)

computationally expensive due to the need to evaluate the model on all possible subsets of features.

KERNEL SHAP AND TREE SHAP

Kernel SHAP model agnostic approximation that approximate SHAP with the LS minimization

$$\arg\min_{\phi} \sum_{z' \subseteq x'} \left(f_x(z') - \phi_0 - \sum_{j=1}^{M} \phi_j z_j \right)^2 \pi(z')$$

z': binary vector that signals the presence of features

$\pi(z')$: weighting vector

Tree SHAP designed for tree-based classifiers

1. Traverse the tree to determine the contribution of each feature at each node.

2. Aggregate the contributions across all paths leading to a prediction.
3. Sum the contributions to obtain the SHAP value for each feature.

MAIN ISSUES FOR SHAP

- Computational complexity
- Independence assumption
- Model-dependence
- Interaction effects
- Approximation errors
- Interpretability vs. Accuracy

LOF INTERPRETABILITY AND IF IMPORTANCE

- applied to models like LOF to understand why specific instances are considered outliers.
- feature importance can be extracted to understand which features contribute more in anomaly

LOF case: looks why behavior is very far from other samples.

IF case: nodes "split" on the data features, aiming to "isolate" the unusual ones. Outliers tend to require fewer splits to be isolated, while normal points take more splits to reach isolation.

- frequent splits on important features (model often uses x_k early to isolate outliers)
- Example Breakdown: IF flags a transaction for being unusual and calculates feature importance based on how often each feature was used to isolate it.

You can add **counterfactual explanations** as well.

TIME SERIES ANOMALY DETECTION

- **LSTM Interpretability:** understanding the contribution of each time step to the anomaly score (layer-wise relevance propagation or LRP)
- **Temporal counterfactual explanations:** modify features according to their position and time: understand temporal pattern.

Example 5: daily website traffic for an e-commerce site. Usually, traffic follows a predictable pattern:

- Weekdays: Traffic is steady, peaking during lunchtime and in the evenings.
- Weekends: Traffic is usually higher overall

KEY STRATEGIES

1. Visualization (visualize data)
2. Human-in-the-Loop
3. Threshold Setting
4. Explanations for False Positives
5. Time Series Anomaly Detection
6. Ensemble Models for Anomaly Detection.

ADVERSARIAL STRATEGIES

SECURITY-AWARE ATTACKERS

- Previous assumptions: WDAD
- Anomalies have their own fixed distribution
- No adaptation is considered

So if you are an attacker you try to change your strategies.

What if the anomalies evolve accordingly with the detection method? Ex. Network attacker that is aware of NIDS and HIDS Tune the strategy in order to trick the safety measures. Pay a cost for it!

ADVERSARIAL MACHINE LEARNING

I want to created a sample that fool the system.

Adversarial machine learning is a machine learning method that aims to trick machine learning models by providing deceptive input. (image classification, spam detection)

Adversarial example: sample purposely designed to create a mistake

Whitebox scenario: Attacker knows the detector (architecture and params)

Targeted attack: Tries to misguide the model to a particular class

Universal attack: Work for a dataset

Blackbox scenario: Attacker does not know the model (only outcomes of detector)

Untargeted attack: tries to misguide the model to predict any of the incorrect classes. (don't care which is the final class)

Individual attack: Works for a specific sample

Purposes: trustworthy AI systems, security standards for ML systems; mitigate potential risks of data corruption, model theft; often analyzed in association with privacy-preserving machine learning (PPML).

ATTACK TYPES

Poisoning attacks

- Attackers can affect training data or their labels
- Injection of malicious samples during data collection
- Very effective in AD

Evasion attacks

- Most frequent
- Attacker manipulates samples in order to trick a previously-trained detector/classifier
- Ex. Intrusion and malware

Model extraction

- Attacker probes a black box machine to reconstruct data or model
- Used when model is sensitive and confidential
- Ex. Stealing a stock market predictor

ADVERSARIAL STRATEGIES (1/2)

The idea: you have a function f (in *your* case is a loss function), and you want minimize this function (if minimal meaning that is good). Your classifier map your input in one feature array, that can be represented by a feature space, your array is one point into this space. In the space you have different type of class, and you want to bring (at least you want look like) your sample in the "safe" class. You want to slightly modify (add noise) your input sample so you "bring" in the "safe" class, instead a putting in the "attack" class.

Limited-memory BFGS (L-BFGS) BFGS: Broyden-Fletcher-Goldfarb-Shanno method. (minimize $f(x_k)$)

The signal is the difference between the fake image and the real image, you want to minimize this difference (which is the function). For doing this we:

- find a direction $B_k d_k = -\nabla f(x_k)$
- find the step $\alpha_k = \arg\min_{\alpha_k} f\,(x_k + \alpha_k p_k)$
- set $s_k = \alpha_k p_k$ and update $x_{k+1} \leftarrow x_k + s_k$ (x_k depends on the modification that we have done)

- setting $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$ update $B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^T B_k^T}{s_k^T B_k s_k}$
  - Non-linear gradient-based method to minimize the number of perturbations in altering image

✓Effective to create adversarial samples. ✗Computationally-expensive

The most of the time we don't have the computational resources and we don't have enough time to develop this so we do FGSM.

Fast Gradient Sign method (FGSM) Fast method to minimize perturbation to minimize a loss function J (between target and actual) (you basically changing one feature at the time) (less effective)

The loss function is the difference between the features of the fake images and the feature of the real images, the ϵ determine how much you want to modify the images (most effective solution is to use many small ϵ and than reupdate the gradient).

$$\tilde{x}_k = x_k + \epsilon\, sign\big(\Delta_x J(\theta, x_k, y_k)\big)$$

✓Computationally-reasonable. ✗ Perturb one feature at a time

ADVERSARIAL STRATEGIES (2/2)

Jacobian-based Saliency Map Attack (JSMA) Like FGSM with flat perturbations to features iteratively according to saliency value

✓Minimum necessary features are perturbed

✗more complex than FGSM

Deepfool attack Untargeted, decision boundaries between classes are estimated, and perturbations are added iteratively.

✓Efficient in creating adversarial samples.

✗more complex than FGSM, JSMA; not so efficient all-the-times

Carlini & Wagner Attack (C&W) L-BFGS attack without box constraints and different objective functions.

✓Effective against defensive distillation and adversarial training Effective, can defeat some defenses.

✗more complex than FGSM, JSMA,

Zeroth-order optimization attack (ZOO) estimates gradient and hessian by querying the model with modified individual features (use perturbations).

✓similar to C&W fully black-box (no train or info)

✗requires many samples

FOCUS ON FGSM

The idea: perturb your features progressively, time to time; this perturbation follows the direction of the maximum difference.

In a **targeted attack** you want to minimize the distance (close to wrong classification).

The **untargeted attack** you want to maximize the distance (far from a correct classification).

Fast Gradient Sign method (FGSM) Move in the feature space computing gradients to find out the most effective direction to exit the region. $\tilde{x}_k = \boldsymbol{x_k} + \epsilon\, sign\big(\Delta_{\boldsymbol{x}} J(\theta, \boldsymbol{x_k}, y_k)\big)$

Added noise: $\epsilon\, sign\big(\Delta_{\boldsymbol{x}} J(\theta, \boldsymbol{x_k}, y_k)\big)$ where ϵ controls the amount of motion

ϵ can be adapted during the walk: more than one step!

DEFENSES

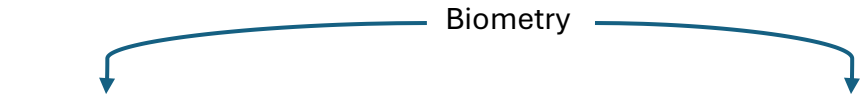
- Adversarial training
- Ensemble of neural networks
- GAN Defense

THE ROAD TO GAN

- Detectors are generated on known statistics → Anomaly detection/classifiers
- Attacker knows that analyst use detector … optimize the adversarial sample on the detector → Adversarial Machine Learning
- Detector is refined so that attacker must be smarter…. Non-cooperative game → Equilibrium point? We don't kwon if is possible to reach
- Process if iterated … GAN → Better generator and discriminator

LECTURE 17 (INTRO ON BIOMETRICS: FP)

A TAXONOMY OF BIOMETRY



Physiological:

- Fingerprints
- Face
- Hand
- Eyes
- DNA
- Anthropometric measurements

Behavioral:

- Voice
- Signature
- Keystroke dynamics
- Gait
- Social interactions

IDENTIFICATION/VERIFICATION

Identity verification (or simply verification): A person declares his/her identity (e.g., PIN); the system directly matches (1:1) the person's current biometric characteristic with previously-acquired one stored on a database and which retrieved via the identity.

Identification: The system scans a set of candidates, and decides whether one of them matches the person to be identified (1:N). (you doing search)

Multi-factor authentication Since every strategy presents some weakness, the combination of different factors is used. e.g. ATM card + PIN

ATTACKS (1/2)

Masquerade: Attacker convinces the system that he/she is another person in order to get access to some particular ability or attribute some actions/responsibilities to another user.

Multiple identities: The same user/person can have access multiple times to the same ability/resource.

Identity theft: Attacker subscribes to the system and creates new account pretending to be the victim.

Trial and error attacks (interactive): Password guessing or you reproduce the authenticator method.

Offline attack: try deciphering an encrypted password

- Small words
- Dictionary attack

Tokens and smartcards

Reproduces the authenticator

ATTACKS ON BIOMETRICS (2/2)

Replication

- Attacker creates a copy of the authenticator from another person
- Mimic the identity
- Artifacts

Theft

- Steals the authenticator
- Difficult on biometrics
- Difficult for biometrics (they can’t steal your eyes)

Digital spoofing (playback attack)

- Attacks operate during the transmission
- The attacker intercepts the transmitted value and replays it
- Sometimes preventing the identification is enough

DEFENSES

Trials-and-errors

- Increase the size of the base secret and the sensitivity of the verification
- Limit the number of guesses
- Reduce the FAR by getting more information from sensors
- Adjust how closely the authenticator must be close to the verifier; this could increase the rejection of positive samples.

Replication

- A set of strategy can be used to prevent replication of authentication devices and biometric traits
- Their effectiveness depends on the facilities of the attacker
- A common countermeasure is to incorporate liveness information (traits or behaviors that aren't present in a static copy of a biometric trait). E.g.; temperature in fingerprints, 3D aspects on faces.

Digital spoofing

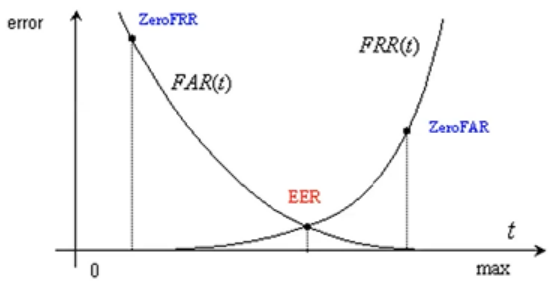
- Attacks operate during the transmission
- Use encryption and digital signature strategies

BIOMETRIC PERFORMANCES

- Two samples acquired in different sessions, usually do not exactly coincide. (e.g. a face acquired in a dark room or in a light one)
- Compute a similarity score and compares it with an acceptance threshold.
- Need to be robust for difference conditions
- Similarity greater than the threshold → two samples coincide.

FRR (False Rejection Rate): frequency of rejections relative to people who should be correctly verified. Note that a false rejection does not mean necessarily an error of the system (e.g., an incorrect positioning or dirtiness).

FAR (False Acceptance Rate): the frequency of fraudulent accesses (e.g., impostors claiming a false identity).



FAR is also called False Match RATE (FMR)

Given a threshold t, if it's low is more likely to accept (you increase t and FAR decreases). Instead in the FRR t starts very low and increases. You want to look for the interception point (EER – equal error rate), the point where the error rate between the false positive and the false negative is equals.

ERRORS

Metric	Problem	Solution
All	Template aging as a result of age, gender, other factors	Periodic re-enrollment
Fingerprint	Degradation of fingerprints caused by occupation, age, trauma	Enrollment of multiple fingers
Facial	Environmental factors such as lighting, background, contrast; pose, movement of subject, glasses	Use in controlled environmental scenarios; use multiple scans from different angles
Voice	Illness, age, quality of communication system	Allow for re-enrollment as necessary
Hand	Injury, trauma to hand	Enrollment of both hands to reduce reliance on single hand
Iris	Positioning, eye angle, glasses	Increased user training and proper placement of scanner

BIOMETRICS AND SOFT-BIOMETRICS

Some measurements univocally identify a person (e.g., fingerprints) Some others can be used to select a subset of users (e.g. height)

Soft Biometrics traits are physical, behavioral or adhered human characteristics that, unlike classical biometric traits, aim at differentiating individuals.

- Physical: skin color, eye color, hair color, presence of beard or moustache, height, weight, gender, race, ethnicity, wrinkles.
- Behavioral: gait, keystroke, signature.
- Adhered human characteristics: clothes color, tattoos, accessories.

So we don't identify a specifics suspected but can exclude suspects.

For a legal point of view can be a problem because there is not a UE regulation but every country use soft biometrics as they used to. For Italian regulation soft biometrics are consider as regular biometrics.

MOTIVATIONS OF BIOMETRICS

- **Convenient authentication**: nothing to lose or forget since the characteristics or traits of the person serve as the identifiers, eliminate the need to replace badges or reset PINs.
- **Increased need for strong authentication**: biometry reduces the risk that an adversary can present a suitable identifier and gain unauthorized access.
- **Decreased costs**: hardware and software improvements have brought down the costs of biometric authentication and increased their efficiency
- **Increased government and industry adoption**

BIOMETRIC MEASUREMENTS

- Facial recognition
- Fingerprints
- Iris scan

- Palm print
- Voice recognition
- DNA
- Emerging biometrics
    - Footprints
    - Hand geometry
    - Gait recognition

USE OF FINGERPRINTS

One of the oldest biometric measurements to be used

Many different applications

- Criminal investigations
- Border controls
- Victims identification (murders, disasters, ...)
- Access control (phones, applications, buildings, ...)

Different types of acquisition systems/scanners

- Optical: measures the reflected light emitted by the sensor
- Capacitive: relies on the conductivity of skin-cells, depending on where ridges touch the sensor we have different voltages.
- Thermal scanners: measures the difference in temperature between ridges and furrows of friction ridge skin.
- RF scanners: radiate skins with e-m waves and an array of antennas re-acquire the signals.
- Ultrasonic: are similar to RF but using soundwaves.
- Pressure-based: most of the time acquired from surfaces and then scanned.

Often characterized by the so-called "CSI effect", i.e., people having unrealistic expectations for the use of scientific evidence.

So at the end you have an image to analyze.

FINGERPRINTS MATCHING

Matching can not be very easy! (usually they give you a range of solutions and the final identification is done by vision inspection)

- Images could be partial or unclear
- Search for large database
- Human can make errors or can be biased

FINGERPRINTS IMAGE

Fingerprints can be modelled as 2D greyscale signals

$I: R^{\mathbb{2}} \mapsto R$ or [0, 255]

Typically, acquisition can be performed by different sensors Most common one: imaging

- Need to match the acquisition with a database. (you have multiple acquisition for that)
- Each individual multiple acquisition
- Different resolutions are possible
- Recent solution by Xiaomi using multiple sensors

FINGERPRINTS PATTERN MATCHING

You can find out the rotation and translation that maximize the correlation between this finger and this finger. Problem if the finger print is partial.

Image-based: minimize global correlation

$$S(T, I) = \max_{\Delta x, \Delta y, \theta} C\, C\left(T, I^{\Delta x, \Delta y, \theta}\right)$$

simple but:

- Many orientation and shifts must be tested
- Troublesome with partial fingerprints, noise, latent or occluded part
- Central rotation misclassified as obliteration

Fingerprint Alterations Type Detection Using Deep Convolutional Neural Network

FEATURE BASED

Define local characteristics that can be matched with respect to a template fingerprint. Local features can be organized into multiple hierarchical levels

Level 1: ridges orientation and frequencies

Level 2: ridges and minutiae

Level 3: pores and partial ridges

Ridge orientation and frequency estimation (level 1) → Ridge extraction (level 2) → Singularity extraction → Minutiae extraction

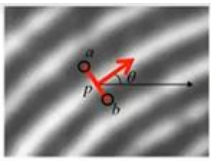
Singularities are single special points that can be used to orient the fingerprint.

Minutiae are small points can be used for alignment, distortion and identification.

Connected to the image resolution (250, 500, 1000 dpi)

LEVEL 1 FEATURES

Ridge orientation map: local ridge orientation. Ridge characterize by:



- Ridge flow (direction)
- Orientation frequencies

(exact location and dimension are ignored)

To have ridge we need at least a resolution of the image of 250 dpi sensors are required (min)

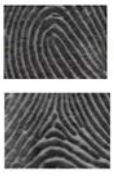
θ local ridge orientation at (x, y): is basically the tangential direction of ridge through (x, y).

Local ridge frequency at (x, y): is the average number of ridges per unit length along a line segment centered at (x, y) and normal to the local ridge orientation $(\perp, \theta)$ (given the ridge with a certain direction and position and given a unit I count around this direction how many ridges we find); frequency is the inverse of period p (less important than orientation).

Ridge orientation map: contains salient location where orientation changes abruptly. It's what a need to identify the salient information .

Two singular types:

- **Loop (core):** ridges enters from one direction and exits in the same direction.
- **Delta:** local area where three ridge system appear to meet.



REPRESENTATION OF ORIENTATION MAP

Loop can be used to align fingerprints. The number of loops is the number of data. This imply that the number of singular points are either 0,2 or 4.

Core: (special singularity) north most loop-type singular point in a fingerprint; if no singular point (e.g., arch-type fingerprint), point of maximum ridge curvature.

The orientation map can be represented by:

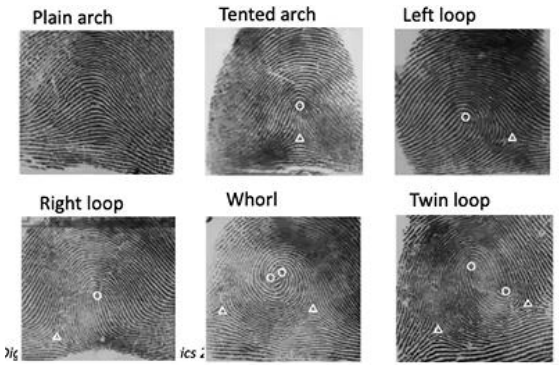
- Set of singular points.
- Pattern type (fingerprint class): number of loops and deltas with the spatial relationship between them.

Constraints for full prints:

- #loops = #deltas
- #singular points = 0, 2, or 4.

Major fingerprint types:

- Plain arch – no loops present → no data
- Tented arch – 1 loop and 1 delta, delta is pointing towards the loop
- Left loop – 1 loop and 1 delta, the loop is on the left respect to the direction of delta
- Right loop -1 loop and 1 delta, the loop is on the right respect to the direction of delta
- Whorl – 2 loops and 2 deltas, the loops already closed that create a kind of a vortex and two delta aside
- Twin loop - 2 loops and 2 delta



HOW TO COMPUTE ORIENTATION: GRADIENT

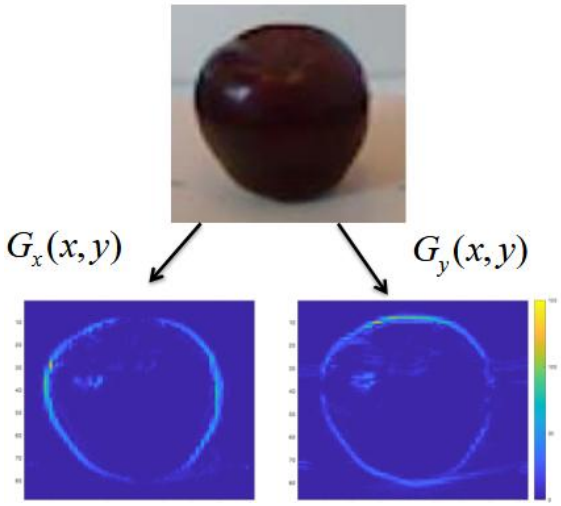
Gradients give us important information

You take on pixel at position (x, y) and take another pixel at the position $(x + 1, y)$, you compute the difference and you gate the **horizontal gradient**.

You take on pixel at position (x, y) and take another pixel at the position $(x, y + 1)$, you compute the difference and you gate the **vertical gradient**.

Combining vertical and horizontal gradients you get the orientation (gradients object edges)

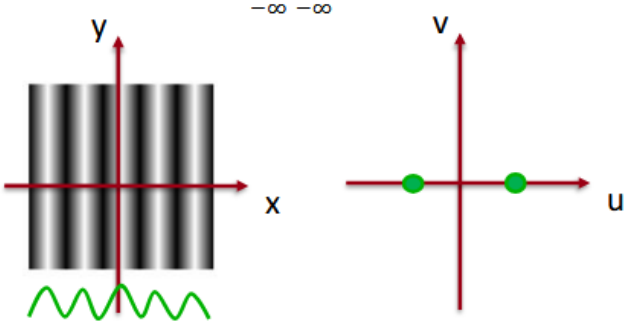
- $G_x(x, y) = I(x + 1, y) - I(x, y)$ (horizontal)
- $G_y(x, y) = I(x, y + 1) - I(x, y)$ (vertical)



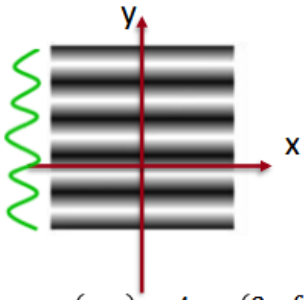
SOME CLARIFICATIONS ON 2D FOURIER TRANSFORM

So I can transform a small portion of the finger (using Fourie Transform) and looks for the pics (*credo*), so I can find out the orientation of my cosine.

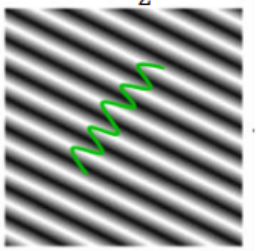
$$I_a(f_x, f_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} i_a(x,y)\, e^{-j2\pi(f_x x + f_y y)} dxdy = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} i_a(x,y)\, e^{-j2\pi(f_x x + f_y y)} dxdy$$

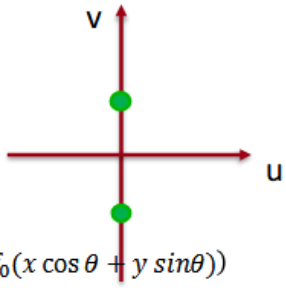


$$w(x.v) = A\cos(2\pi f_0 x)$$
$$W(u,v) = \frac{A}{2}[\delta(u - f_0) + \delta(u + f_0]$$

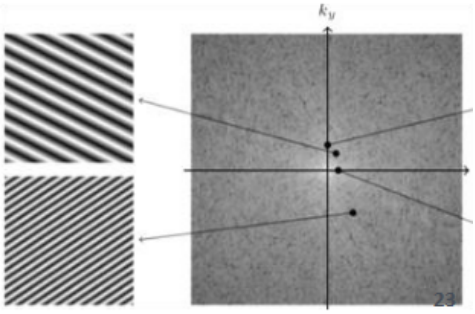


$$w(x,y) = A\cos(2\pi f_0 y)$$
$$W(u,v) = \frac{A}{2}[\delta(v - f_0) + \delta(v + f_0]$$





$$w(x,y) = A\cos(2\pi f_0(x\cos\theta + y\sin\theta))$$
$$W(u,v) = \frac{A}{2}[\delta(u - f_0\cos\theta, v - f_0\sin\theta) + \delta(u + f_0\cos\theta, v + f_0\sin\theta)]$$



RIDGE ORIENTATION AND FREQUENCY ESTIMATION (1/2)

Ideal model: local ridge pattern = a cosine wave

$$w(x,y) = A\cos\big(2\pi f_0(x\cos\theta + y\sin\theta)\big)$$

which can be represented by the Fourier transform

$$W(u,v) = \frac{A}{2}[\delta(u - f_0\cos\theta, v - f_0\sin\theta) + \delta(u + f_0\cos\theta, v + f_0\sin\theta)]$$

Find the maximum points → for the cosine model they are located at

$(-f_0\cos\theta, -f_0\sin\theta), (f_0\cos\theta, f_0\sin\theta)$

Max. at $(\hat{u}, \hat{v})$ ⟶ $\hat{A} = |\hat{W}(\hat{u}, \hat{v})|$ $\quad \hat{\theta} = \arctan\left(\frac{\hat{v}}{\hat{u}}\right)$ $\quad f_0 = \sqrt{\hat{u}^2 + \hat{v}^2}$

With Fourie Transform I can estimate both the orientation and the frequency at the same time.

RIDGE ORIENTATION AND FREQUENCY ESTIMATION (2/2)

Real signal: blurred signal which require a signal processing approach

1. Fast Fourier Transform
2. Smoothing with a low pass filter
3. Maximum detection

Sometimes you do some low-pass filtering in order to smooth a little. You want a flow more regular as possible.

Note that FFT of a real sine signal shows symmetry

1. Build a vector field $V = (V_x, V_y) = (\cos 2\,\theta, \sin 2\,\theta)$
2. Low pass filtering on $\boldsymbol{V} \rightarrow \boldsymbol{V'}$
3. Orientation is computed as $\hat{\theta} = \arctan\left(\frac{V_y'}{V_x'}\right)$

SINGULARITY DETECTION

Poincaré index refers to the cumulative change of orientations along a closed path in an orientation field. (in our case, path of 8 neighbors) (you take a position of a specific point and you look at the orientation of the tiles around that)

Given the 8 neighbor's orientations $O(i) = 0, \ldots, 7$

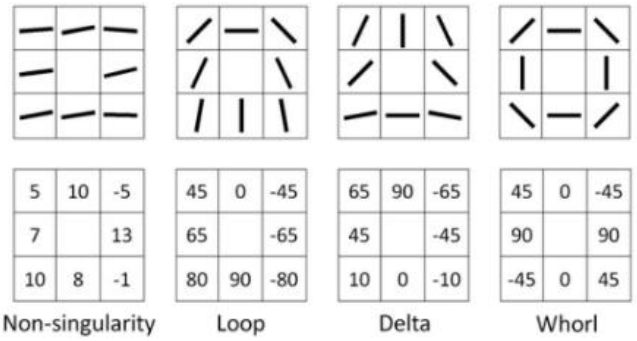
$$PI = \frac{1}{\pi}\sum_{i=0}^{7} \delta(O[(i+1)_{mod\,8}] - O[i])$$

where

$$\delta(\theta) = \begin{cases} \theta - \pi & if\ \theta > \frac{\pi}{2} \\ \theta & if\ -\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2} \\ \theta + \pi & if\ \theta < -\frac{\pi}{2} \end{cases}$$

Cases:

- PI=0 non-singular
- PI=1 loop
- PI=-1 delta
- PI=2 whorl (2 loops combined)



SINGULARITY ORIENTATION

Starting from a reference singularity, the orientation field for loops and deltas is

$$RO_l(x,y) = \frac{1}{2}\arctan\left(\frac{V_x}{V_y}\right) = \frac{\theta}{2} \qquad RO_d(x,y) = -\frac{1}{2}\arctan\left(\frac{V_x}{V_y}\right) = -\frac{\theta}{2}$$

Now assume that the singularity is rotated by α; therefore, RO becomes

$$RO_l(x,y;\alpha) = \frac{\theta - \alpha}{2} + \alpha = \frac{\theta}{2} + \frac{\alpha}{2} \qquad RO_d(x,y;\alpha) = -\frac{\theta - \alpha}{2} + \alpha = -\frac{\theta}{2} + \frac{3\alpha}{2}$$

Therefore, the difference between orientation fields of a singularity in (x_0, y_0) and the reference one is

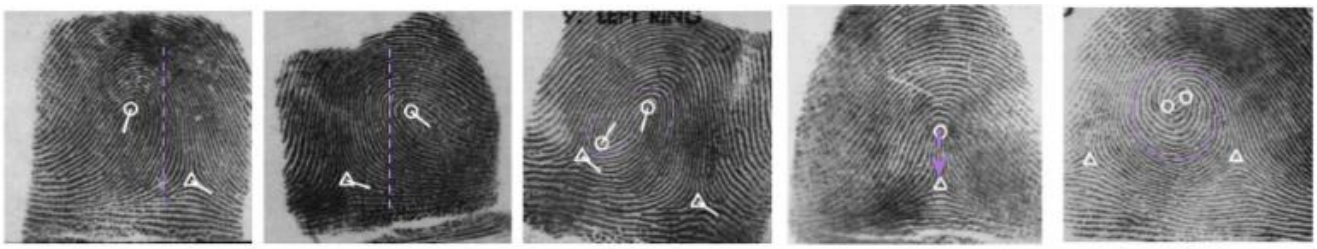
$$D_{\{l,d\}}(x,y) = O(x\lambda + x_0, y\lambda + y_0) - RO_{\{l,d\}}(x,y)$$

which allows to estimate the angle

$$\alpha = \arctan\left(\frac{\sum_{x=-r}^{r}\sum_{y=-r}^{r}\sin\left(2D_l(x,y)\right)}{\sum_{x=-r}^{r}\sum_{y=-r}^{r}\cos\left(2D_l(x,y)\right)}\right) \qquad \alpha = \frac{1}{3}\arctan\left(\frac{\sum_{x=-r}^{r}\sum_{y=-r}^{r}\sin\left(2D_d(x,y)\right)}{\sum_{x=-r}^{r}\sum_{y=-r}^{r}\cos\left(2D_d(x,y)\right)}\right)$$

The rotation of the finger can be a problem only if you have to align the finger, in general is not a problem.

CLASSIFICATION



- Plain arch: no singular point
- Left loop: 1 delta + 1 loop pointing to the left side of delta
- Right loop: 1 delta + 1 loop pointing to the right side of delta
- Tented arch: 1 delta + 1 loop pointing towards delta
- Whorl: 2 loops + 2 deltas with ridges around loops forming a circle
- Twin loop: 2 loops + 2 deltas with ridges around loops not in a circle

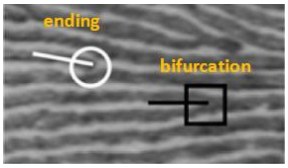
LEVEL 2 FEATURES

We want have large amount of points because we cannot think of identification only using loops and deltas.
Ridge skeleton image in which each ridge is only one-pixel wide.

- Location are recorded
- Geometric and dimensional details are not considered

In case of partial fingerprint using only loop and delta is not enough, for this reason it was introduced minutia. Minutia are points that are characteristics of the ridge.

Minutia (pl. minutiae) (finer elements) or ridge characteristic: locations where a ridge emerges, ends, splits, or merges with another ridge. Characterized by 3 attributes:



- Location
- Direction
- Type

Minutiae-based recognition is largely involved in automated recognition.
Required resolution: 500dpi.

LECTURE 18 (FINGERPRINTS)

RIDGE EXTRACTION

Simple thresholding does not work since:

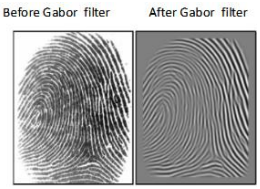
a) Pores are brighter.

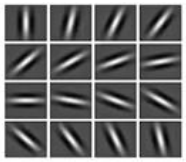
b) Ridges can be broken due to cuts or creases.

c) Adjacent ridges may appear joined due to moist skin or pressure.

Filtering (operation that highlights and give an high response) the ridge: in this case the filter define orientation. This set of filters highlights those points where a specific orientation is present: they are called gamma filter.

Filtering is better (Gabor filters):





Then, the enhanced image can be converted into binary by a threshold, followed by **thinning** (ridges are reduced to one pixel width).

For every pixel we have from 0 to 255, but ridges are simple line so I don't need 256 different level. So I can do a thresholding, so that all the pixel higher than the threshold are classified as 1 and all pixel lower are classifier as 0.

THINNING (SKELETONIZATION)

Morphological operation that simplifies an image into a topologically equivalent one.

- Typically operated on bilevel {0,1} images.
- Defined by a structuring element and uses a hit-and-miss transform.

After a thinning operation the lines are exactly 1 pixel wide.

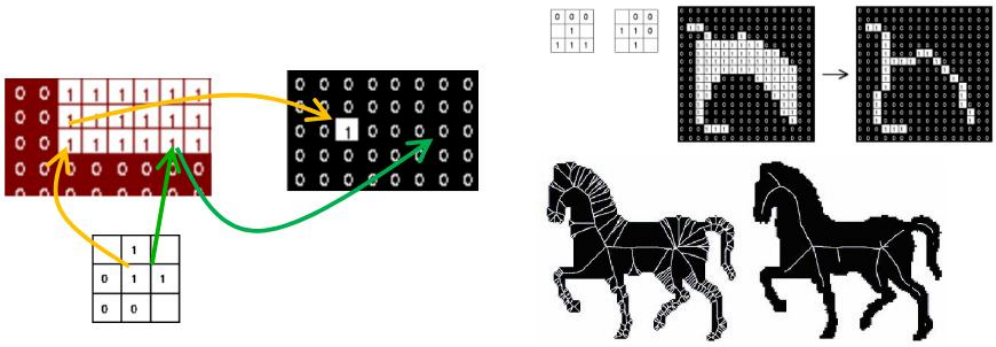
Hit-and-miss transform HM(I,J):

ASK GEMINI HOW IT WORKS

a) Center the structuring element J on a pixel.

b) Compare the element J with the image I.

c) If pixels match, then the central pixel is set to 1; 0 otherwise.

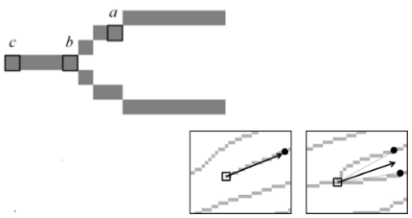


Thinning formula: $Thinned = I - HM(I,J) = I \cap \neg HM(I,J)$

Note: more than 1 operator can be applied in sequence.

Used to highlight some specific point where the image has a specific structure. It's like have a kind a skeleton of the image. I can apply this multiple times.

MINUTIAE EXTRACTION



#neighbours = type of minutia

These 3 points depend on the number of neighbors each point have.

- 1 neighbor: **ridge ending**. (c)
- 3 neighbors: **bifurcation**. (b)
- 2 neighbors: **typical ridge pixel**. (a)

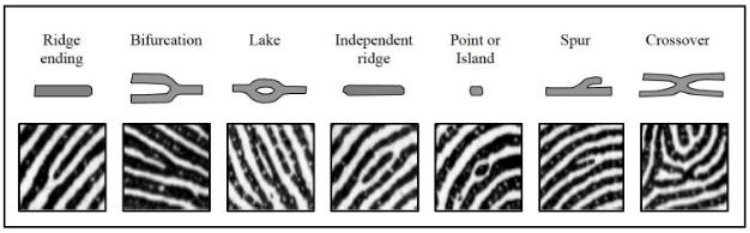
Directions of endings and bifurcation can be computed as well. Many minutiae appear because of noise and poor image quality. The important points are (b) and (c).

These leads you to a certain number of points that are characterized by: coordinates, directions and class. The problem is that there are a lot of points, probably some of them are not useful. Therefore you can filter out some points: spurious minutiae filtering.

Spurious minutiae filtering:

1. Minutiae that do not have an adjacent ridge on either side (finger border).
2. Minutiae that are close and almost opposite in direction.
3. Too many minutiae in a small neighborhood.

FINAL CONSIDERATIONS ON MINUTIAE



Notes:

- **Main minutiae types:** ending and bifurcation.
- Other types can be considered (not used because of robustness and complexity).
- **Ambiguity:** Ambiguity between types is possible (it depends on how much you press).

Different conventions:

- *ANSI/NIST-ITL 1, 2007* proposes 4 types: **ending**, **bifurcations**, **compound** (trifurcations or crossovers), **undefined**.
- *FBI minutiae-coordinate model (McCabe, 2004):* **ridge endings** and **bifurcations**.
- *ANSI/NIST-ITL 1 (2007) Type-9:* records each minutia as class, (x,y) coordinates and orientation.

PROBLEMS IN MATCHING

Fingerprint matching can be impaired by many different factors (intra-class variability):

- Displacement
- Rotation
- Partial overlap
- Non-linear distortion
- Pressure and skin conditions
- Noise
- Feature extraction errors

All of these are problem you can deal with.

MATCHING FINGERPRINTS

Process Flow: Query minutiae / Template minutiae → Alignment → Pairing → Scoring → Score (number of minutiae that corresponds between the two fingerprints)

- Compute the transformation between two minutiae sets so that the coordinates are the same.
- Match couples of minutiae.
- Compute the score.

PROBLEM FORMULATION

Let's consider two sets of minutiae from two images: input image I and template image T.

- $T = \{m_1^T, m_2^T, m_3^T, \dots, m_m^T\} \quad I = \{m_1^Q, m_2^Q, m_3^Q, \dots, m_n^Q\}$
- $m_i^T = (x_i^T, y_i^T, \theta_i^T) \qquad m_j^Q = (x_j^Q, y_j^Q, \theta_j^Q)$

Two minutiae are matching if the distance between the coordinates is lower than threshold, and if the difference between the orientation is lower than a specific threshold.

Two minutiae $\boldsymbol{m_i^T}$ and $\boldsymbol{m_j^Q}$ are to be considered "matching"

$$MM(\boldsymbol{m^T}_i, \boldsymbol{m^Q}_j) = 1 \quad if \quad \begin{cases} d_c(\boldsymbol{m^T}_i, \boldsymbol{m^Q}_j) = \sqrt{\left(x_i^T - x_j^Q\right)^2 + \left(y_i^T - y_j^Q\right)^2} < Th_d \\ d_\theta(\boldsymbol{m^T}_i, \boldsymbol{m^Q}_j) = \min\left\{|\theta_i^T - \theta_j^Q|, 360° - |\theta_i^T - \theta_j^Q|,\right\} < Th_\theta \end{cases}$$

- Function MM is equal to 1 if these two conditions are satisfied. Function MM is 0 otherwise.
- Let's avoid considering type, distortion and scale.

Transformation: A possible transformation between I and T is a roto-translation H.

$$H\left(\boldsymbol{m}_j^Q\right) \; s.t. \; \begin{bmatrix} x_j'' \\ y_j'' \end{bmatrix} = \begin{bmatrix} cos\Delta\theta & -sin\Delta\theta \\ sin\Delta\theta & cos\Delta\theta \end{bmatrix} \begin{bmatrix} x_j^Q \\ y_j^Q \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

i.e.

$$\max_{\Delta x, \Delta y, \Delta\theta} \sum_{i=1}^{m} MM(\boldsymbol{m}_i, \boldsymbol{m''}_j) = \max_{\Delta x, \Delta y, \Delta\theta} \sum_{i=1}^{m} MM\left(\boldsymbol{m}_i, H(\boldsymbol{m^Q}_j)\right)$$

COMMENTS

- Note that the pairing found by the previous equation does not mean that m_j^Q matches m_i^T, it means that for the considered transformation m_j^Q is the most likely match for m_i^T.
- It is possible to generalize the matching with the pairing function $j = P(i)$.
- **Different solutions to solve the problem:** Hungarian algorithm, quantization + exhaustive search, coarse-to-fine search over the parameter space, etc.

Similarity score: $score = \frac{k}{(n+m)/2}$

The score tells you how much these two minutiae are matching.

It is possible to:

- Weight the matching according to minutia quality.
- Consider the intersection of minutiae set (useful for partial fingerprints).

POINT PATTERN MATCHING

Algebraic matching:

- Assume that all the minutiae are matching to a minutia in the template.
- Tolerance boxes do not intersect.

Hough Transform:

- Compute the most probable transformation: fits most of the coupled points.

Relaxation:

- Adjust the confidence of point pairs according to the consistency with the others.

Operation Research Solution:

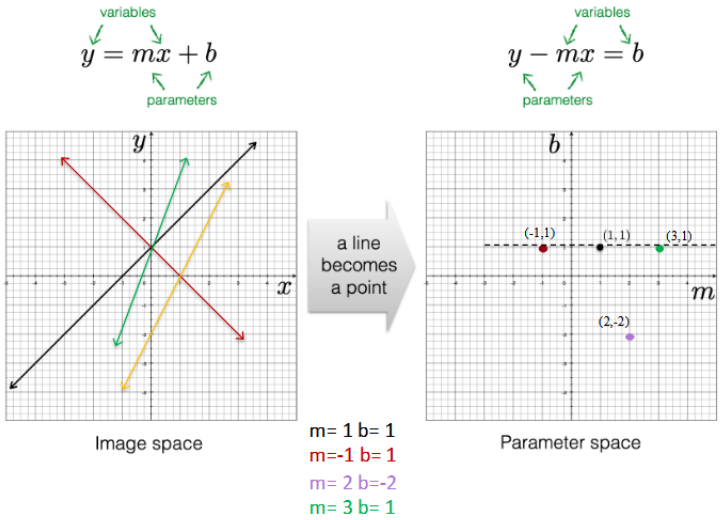
- Searching over a tree of possible matches with different tree-pruning methods.

Energy Minimization:

- Associates an energy or fitness with each solution of the problem.

HOUGH TRANSFORM: PARAMETER SPACE

Suppose you have a set of points and you want find out the line that intersects all points. You want change that space from variable space to parameter space (you will use the *m* and *b* values). It's interesting to notice that when a line intersects a single point then these points are aligned with respect to a single line. Image to represent a line using the slope and intercept parameters:

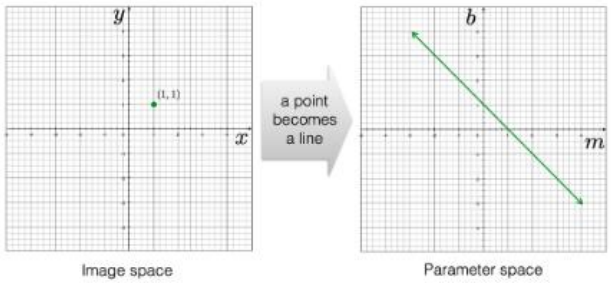


Concept:

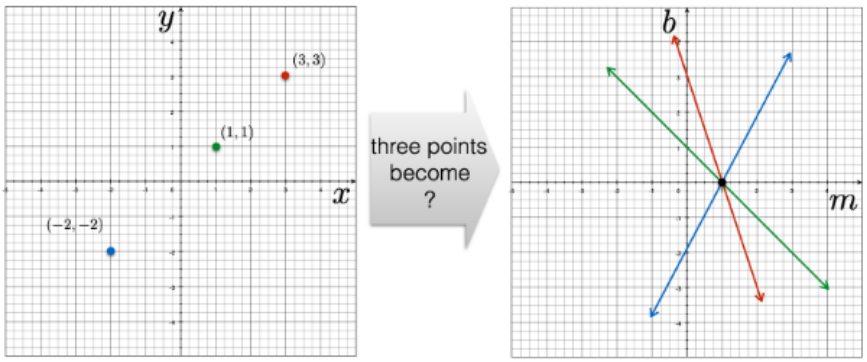
- **Image space:** A line becomes a point in parameter space.
- **Parameter space:** A point becomes a line in parameter space.
- When multiple points are plotted: intersection of all lines in parameter space = fitting line.

HOUGH TRANSFORM: PARAMETER SPACE

A point can be represented by a line (parameters of all the lines passing by that point)



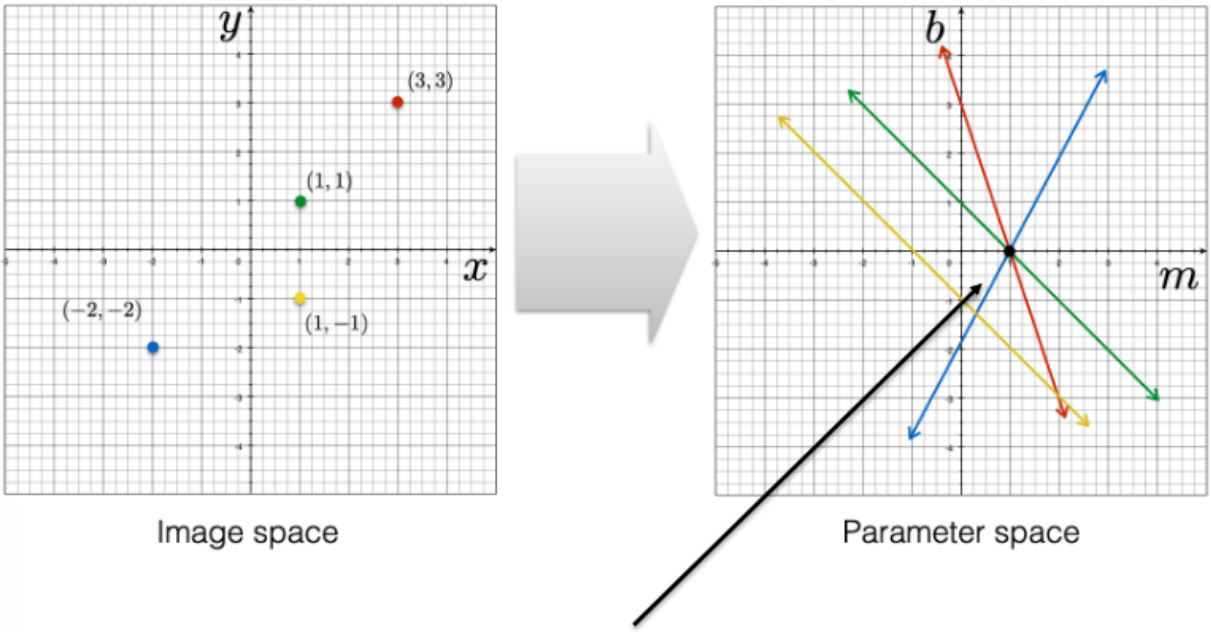
When multiple points are plotted: intersection of all lines in parameter space = fitting line. A line in parameter space defines the parameters around the line passing that point. So when multiple points are plotted, the intersection basically identifies the level that fits.



HOUGH TRANSFORM: FITTING LINES

Assuming that you have a set of points and some of them can be fitted into a line, how can we do?

Starting from a set of points, draw the lines corresponding to the set of points and if you see that these lines has some common intersection you may think there is at least one line fitting all these points (or some of these points). If you count how many lines intersect on these points, it will give you an idea on how many points are encapsulated, defined by that line. You are interested in the lines that connect/defines most of the points.



HOUGH TRANSFORM: ALGORITHM

Starting from a set of points (x_i, y_i):

1. Quantize the parameter space (m, b).
2. Create the accumulator array $\mathrm{A}(\mathrm{m}, \mathrm{b})$.
3. Set $\mathrm{A}(\mathrm{m}, \mathrm{b}) = 0 \ \forall \ \mathrm{m}, \mathrm{b}$.
4. **For** each point (x_i, y_i):

   For each couple of parameters (m, b):

   If the point (x_i, y_i) belongs to the line defined by (m, b), i.e., $y_i = mx_i + b$

   $\mathrm{A}(m, c) \leftarrow \mathrm{A}(m, c) + 1$
5. Find the local maxima of $\mathrm{A}(m, c)$.

ASK GEMINI how the algorithm works.

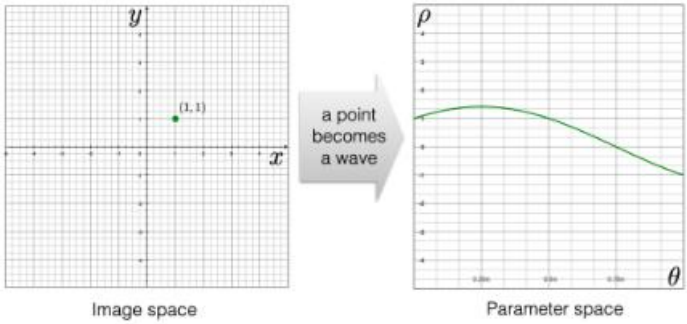
Pay attention that:

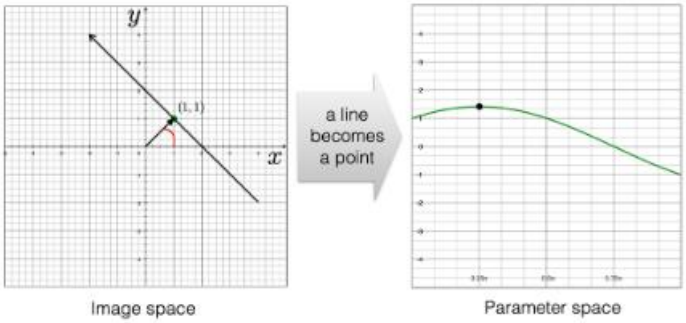
- You can find more than one line (local maxima).
- Quantization of parameters affects your precision.
- Complexity explodes with the number of parameters.

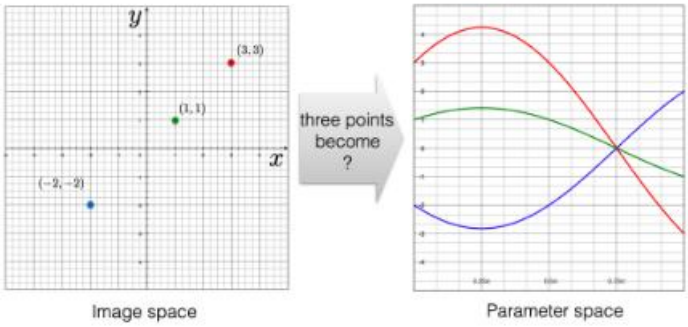
HOUGH TRANSFORM: OTHER PARAMETERS

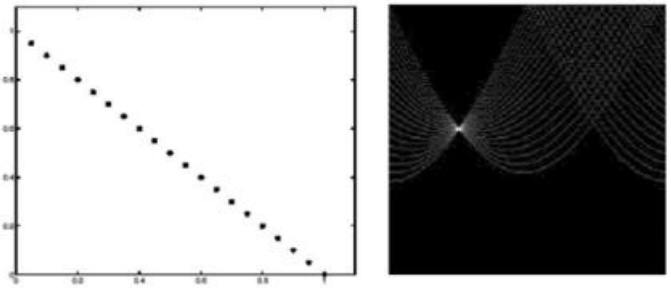
Use a different parameterization: $y = mx + b \rightarrow x\cos\theta + y\sin\theta = \rho$

Using θ and ρ these points becomes a curve.







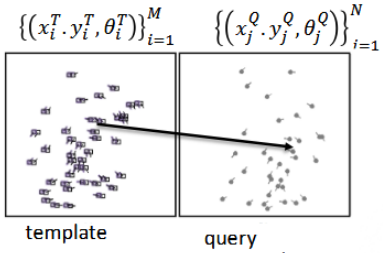


ALIGNMENT

The alignment can be defined by the rotation and the translation. The translation is defined by two parameters $(\Delta x, \Delta y)$, the rotation is defined in 2D by an angle (θ). Therefore you can define a 3-dimentional accumulator(*credo*) where the parameters are $(\Delta x, \Delta y, \theta)$.

The idea(credo): Given every point in the left fingerprints, corresponding to any other points of the fingerprints, you mismatching define one transformation and then with the transformation I count how many times data transformations for that matching is verified in my plot. If you have that lots of points moved from that specific transformation, I identified the most probable rotation and translation that allows me to align one fingerprint with the others.

Generalized Hough transform



The relation/transformation is characterized by a traslation (two parameters $\Delta x, \Delta y$) and by a rotation (1 parameter with angle $\Delta\theta$).

$A[\Delta\theta][\Delta x][\Delta y] = 0 \quad \forall\, \Delta\theta, \Delta x, \Delta y$

for i=1 to M **do**

  for j=1 to N **do**

    $\Delta\theta = \theta_i^T - \theta_j^Q$

    $\Delta x = x_i^T - x_j^Q \cos(\Delta\theta) - y_j^Q \sin(\Delta\theta)$

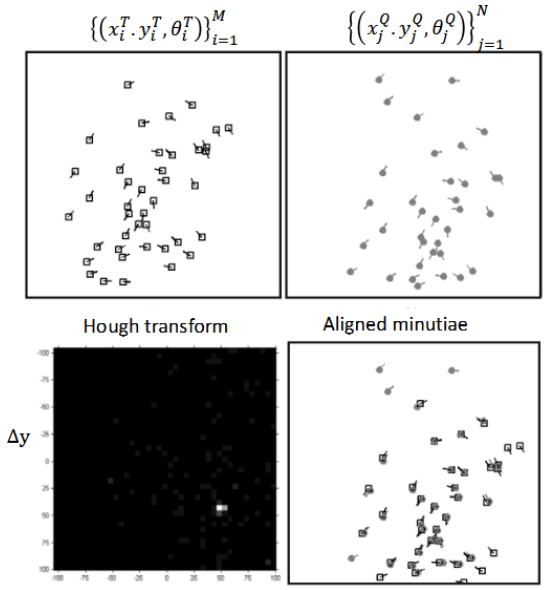
    $\Delta y = y_i^T + x_j^Q \sin(\Delta\theta) - y_j^Q \cos(\Delta\theta)$

    $A[\Delta\theta][\Delta x][\Delta y] = A[\Delta\theta][\Delta x][\Delta y] + 1$

  end

end

Return $\Delta x, \Delta y, \Delta\theta$ of the maximum A



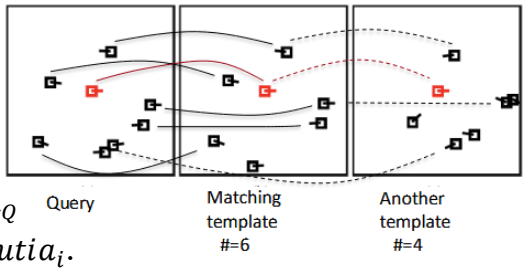
Example with a simple translation

In this case, we performed a simple translation. For the sake of simplicity, I avoided the procedure the rotation. So here, I'm looking simply for the Δx *and* Δy. So you have different Δx here, different Δy here and you can see that different values of Δx have different modes. The most voted Δx *and* Δy lead to misalignment between minutiae. So I was able to find out which is the transformation that matches most of the minutiae. (*credo*)
You want to maximize the number of correspond between the two images.

PAIRING MINUTIAE

After alignment, minutiae need to be matched(means how many minutia are closed enough) and counted.

- **#matches:** probability of correspondences.



How do we compute the number of matches?
Basically you keep two flagging arrays $f^T[i]$ *and* $f^Q[j]$ which are Basically arrays that tells you whether I already have associated values. If the $minutia_T$ has not been assigned yet, and the $minutia_Q$ has not been assigned yet; I verify that the $minutia_j$ is close to $minutia_i$.
How much close? Lower than the threshold. The same fore the angle. If this condition is verified then I can say that i correspond to j and therefore there is one capital minutia matching. You need to have one-to-one matching; you cannot assign minutia to more than one minutia on the other fingerprints.

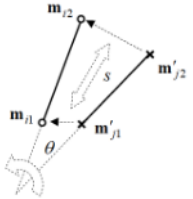
$f^T[i] = 0 \quad f^Q[j] = 0$
for i=1 to M **do**
  for j=1 to N **do**
    if $f^T[i] = 0 \wedge f^Q[j] = 0 \wedge \left\|\left(x_j^Q, y_j^Q\right)' - \left(x_i^T, y_i^T\right)\right\| < t_d \wedge \left|\theta_j^Q - \theta_i^T\right| < t_r$
      set $f^T[i] = 1, f^Q[j] = 1$
      $count \leftarrow count + 1$
    end
  end
end

Check they are not matched
Check they are close (similar) enough

FURTHER ALIGNMENT REFINEMENT

1. Detect the minutiae pair (called the **principal pair**) that receives the maximum Matching Pair Support (MPS).
2. The remaining minutiae mates (i.e., the function P) are then determined once the two fingerprints have been registered.
3. The exact alignment is computed in the least square sense once the correspondence function is known.

- $\theta = angle(\overline{m_{i1}m_{i2}}) - angle\left(\overline{m'_{j1}m'_{j2}}\right)$



- $s = \frac{len(\overline{m_{i1}m_{i2}})}{len(\overline{m'_{j1}m'_{j2}})}$

Usually a scale change is required: the set of parameters becomes (Δx, Δy, $\Delta\theta$, s). Such change is integrated in an iterative optimization.

PRE-ALIGNMENT

In order to maximize the matching performance, both template and query fingerprints can be pre-aligned according to a defined direction.

Absolute pre-alignment: Fingerprints are translated and oriented according to a fixed criteria (with no reference fingerprint). (not required comparing between fingerprints(*credo*))

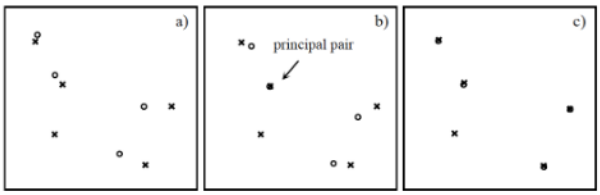
- *Translation:* There is always a **core point**, the coordinates of the core point must be always (0,0).
- *Orientation:* shape of FP, orientation of segment, core-delta, main regions around core, singularities.
- *Risk:* Wrong alignment = FP do not match.

Relative pre-alignment: Less efficient in terms of computational complexity but more effective in terms of matching/unmatching errors.

PRE-ALIGNMENT EXAMPLE

M82(Used for AFIS)

- **Coarse pre-alignment** using core positions and two side regions.
- Define minutiae pairs (proximity criterion) and assign a matching weight (a).
- List is sorted according to the matching weight. (minutia sorted respect to this weight)
- You take the Top 2: **principal pair** -> define a translation (b). This is the principal pair and you translate the two images so that the difference between the principal minutia and the others are basically zero. So in this case, the difference in the principal pair is minimized, subject to zero.
- Deformation tensor compensate small linear distortion and rotations (c).



Relative pre-alignment strategies:

- Superimposing singularities.
- Correlating the orientation images.
- Comparing ridge features (e.g., length and orientation of the ridges).

OTHER RELATIVE PRE-ALIGNMENT EXAMPLES

1. Singularity-based

2. Orientation image-based

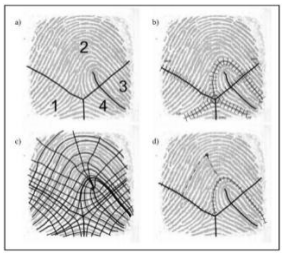
- Compute the orientation field of I and T and then apply a transformation H with parameters (Δx, Δy, $\Delta\theta$).

3. Ridge-based relative

- a. Each minutia can be associated to a ridge.
- b. Each ridge is a curve with origin in the minutia.
- c. Ridges are normalized and sampled.

- d. Ridges are matched finding ($\Delta x, \Delta y, \Delta\theta$).
- Keep on matching pairs of ridges until a certain matching score is achieved.

AVOID ALIGNMENT



Define an intrinsic coordinate system (ICS):

- FP is divided into regions.
- Define axes and coordinates.
- Example of minutia location (defined by region and coordinates).

LOCAL MINUTIAE ALIGNMENT/MATCHING

Some authors have proposed **local minutiae matching techniques** that consist of comparing two fingerprints according to local minutiae structures.

- Local structures are characterized by attributes that are invariant with respect to global transformation (e.g., translation, rotation, etc.): suitable for matching without any a priori global alignment.
- Relaxes global spatial relationship. (useful in case of partial fingerprints or if you have deformation in the fingerprints)

Comparison:

- **Global:** High distinctiveness.

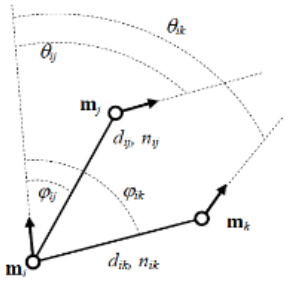
  vs
- **Local matching:** Simplicity, low computational complexity, high distortion-tolerance.

Recent matching techniques tend to combine both.

LOCAL STRATEGIES: INVARIANT DISTANCES AND ANGLES

Look for the two neighbors m_j, m_k of a minutia m_i.

Define a feature vector: $v_i = [d_{ij}, d_{ik}, \theta_{ij}, \theta_{ik}, \varphi_{ij}, \varphi_{ik}, n_{ij}, n_{ik}, t_i, t_j, t_k]$



- **Location distance:** d_{ij}, d_{ik}
- **Orientation difference:** θ_{ij}, θ_{ik}
- **Difference between orientation and edge direction:** n_{ij}, n_{ik}
- **Ridge count:** n_{ij}, n_{ik}
- **Minutia type:** t_i, t_j, t_k

Compute distance between vectors. The concept can be extended to graph structures using more than two neighbors. So I have an array (v_i) and if other fingerprint has a similar array I can say that there is a local structure, in the second fingerprint, which is exactly equal to the ones that they have in the current fingerprint. I can perform the matching even without done the alignment.

Other Local Structures:

- Nearest-neighbor based
- Fixed radius based
- Minutiae triangles
- Texture-based

BEYOND MINUTIAE: RIDGES

- Reliably extracting minutiae from poor quality fingerprints is very difficult.
- Minutiae extraction is time consuming.

- Additional features (in addition or alternatively) may increase system accuracy and robustness.

Texture-information (FingerCode): Input image -> Locate reference point -> Divide image in sectors -> Normalize each sector -> Filtering -> Compute A.A.D. feature -> Calculate Euclidean distance -> Matching result.

Local ridges structures:

- Spatial relationship of the ridge lines.
- Local orientation.
- Local density.

FINAL CONSIDERATIONS ON ALIGNMENT

- Multiple transformations
- Consensus of transformations
- Incremental estimation
- Asymmetrical local matching

DISTORTION

You press the finger with different strength.

Types of distortion:

- Large distortion accumulated.
- Relatively small distortion amount.
- Excessive prolongation and skew required.

Solutions:

- Tolerance boxes.
- Warping.
- Multiple registration and clustering (define a registration pattern). (like on iphone)
- Triangulation.
- Normalization.
- Distortion models.
- Use non-minutiae features.

FINGERPRINT INDEXING (1/2)

You can perform some quick matching by indexing the fingerprints by categorizing the fingerprints.
Assign a descriptor to fingerprints in order to speed-up search. Fingerprint = number of array of values.
Can we speed up this matching process? Yes with:

Triplets of minutiae: 9 values vectors defining the geometric properties of the triangle joining minutiae.
These 9 values are the properties, from a geometrical point of view, of this structure.

- Length of the sides.
- Number of ridges between couples of minutiae.
- Directions of minutiae w.r.t. the longest side.

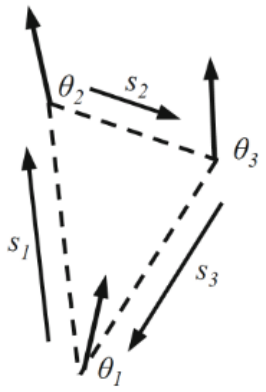
Properties:

- Invariant to rotation and translation, but not to non-linear distortion.
- Ordering of features is important: **geometric hashing**. For each triangle a specify an index (hash value).

- All the triangles of a fingerprint are stored in a hash table that includes the ID of the fingerprint.
- Given a query fingerprint, find the triangles, look into the tables, retrieve IDs lists and check the most voted fingerprint.

FINGERPRINT INDEXING (2/2)

Better descriptors (6 elements vectors):



- Length of the longest side.
- Two smaller internal angles.
- Handedness of triangle.
- Direction of triangle.

The last two elements depend on the type of minutiae.

Process:

1. After finding a match, geometric constraints must be verified.
2. List of corresponding enrolled images I_d.
3. Hypothesis: Query belongs to I_d.
4. For each minutia, find number of triangles including it. → A-posteriori probability.
5. Finally, an indexing score is computed for each I_d by summing its top n largest posterior probabilities.
6. The retrieved images are sorted according to their indexing scores (top N).

This concept is used for speed up the process and avoid to compute everything on all the features/minutie.

ISSUES WITH FP IDENTIFICATION

Need to deal with heterogeneous sensors:

- Low-quality (resolution, partial, …).
  - Wet fingerprints.
  - Cuts.
  - Small overlapping areas.
  - Partial fps.
  - Small resolution.
- Non-linear distortion.
- Latent fingerprints. (overlap with other fingerprints, maybe with the same but with different rotation)
- Altered/fake fingerprints.
- Interoperability between sensor, feature detector and matching.
- Reliability of system-on-devices.
- Security and privacy issues.

In addition to fingerprints, you can also leave “body material”. (other information)

PALM PRINTS

Same stuff, same principle, same technique. The only different is the resolution is higher.

Features:

- Distal transverse crease
- Proximal transverse crease
- Radial transverse crease
- Interdigital

- Ridges / Minutiae
- Hypothenar / Thenar
- Pores
- Palmar friction ridges
- Palmar flexion creases

Notes:

Creases are firmer attachment areas to the basal skin structure.

Claimed to be immutable, permanent, and unique to an individual.

- Many similarities with fingerprint detection.
- Requires resolution > 500dpi.

DEEP LEARNING FOR MINUTIAE EXTRACTION

Minutiae Maps: More recently, deep learning solutions have been proposed for minutiae extraction.

1. FingerNet (2007): It work on low images, basically you are using transitional/convolutional network together in order to estimates the Minutiae Maps.

- Image Normalization -> Group Gabor filters
- ConvNet -> Orientation estimation / Orientation Choose
- Segmentation -> Enhanced Map -> ConvNet -> Minutiae Maps

2. Region-based solution (Zhou et al. 2020): gives a feature map, instead of finding out minutie here we find out a set of features.

- Shared part (ConvNet layers)
- **Stage I:** First prediction (Probability-p, Location-h,w)
- **Stage II:** Patch feature -> Final prediction (Probability-p, Location-h,w, Orientation-o)

LECTURE 20 (FACE RECOGNITION)

FACE RECOGNITION

Face recognition is nowadays widely used in many forensics/security applications:

- People identification (duplicate removal, re-identify people)
- Access control (sort of access key/password; unlock phones)
- Security and surveillance (video surveillance)
- Database search (criminal investigations, missing people, people missing documents)
- Identity verification (match face to ID/credit card/bank account)
- Data organization and indexing

MILESTONES OF FACE RECOGNITION

Every face can be decompose as a linear combination of basic elements of the face.

Different approaches have been proposed to detect faces

- Classical (manifold learning)
- Holistic approaches
- AI-based solutions
- Statistical approaches
- Feature-based strategies
- Filter-based (Gabor/wavelet)
- Face descriptors
- Model-based
- 3D/Video methods

ADVANTAGES OF FR

- Can be performed with longer standoff distance using non-contact sensors. (only need a camera)
- Can be used in video surveillance
- Conveys different information such as emotions of a person (e.g., happiness or anger) and biographic information (e.g., gender, ethnicity, and age)
- Large legacy face databases: different modalities (individuality or scalability) (social media era helps)
- Faces are shared more often (social media applications) than other traits

CHALLENGES TO FR

Set of problems on how the faces have been acquired:

- **Illumination and acquiring conditions**
    - Distance from the camera, image resolution
    - Frontal poses are to be preferred (not always
    - we can have a frontal pose)
    - Exposure time and blurring
    - Light intensity

} Strongly dependent on environments

    - Pay attention to compression effects
    - All these parameters should be equalized
- **Facial expression**
    - Image warping (2D/3D)
    - Use of facial landmark (try to overcome the problem)
    - Warp expressions to normal

These two problems are the main problem and they are called **PIE** (pose illumination expression).

- **Wardrobe and occluding elements** (glass, hat, contact lenses)
- **Poor datasets**
    - Closed-set vs. open set database
    - Only a few acquisitions are available per person.
    - For some people the dataset can be poor.
- **Age** (if the person grows up)
- **Twins or Familiarity**

Face recognition:

- **Face identification**: 1 vs. many (Probes are in the DB / Probes are not in the DB)
- **Face verification:** 1 vs. 1

WHICH SENSORS?

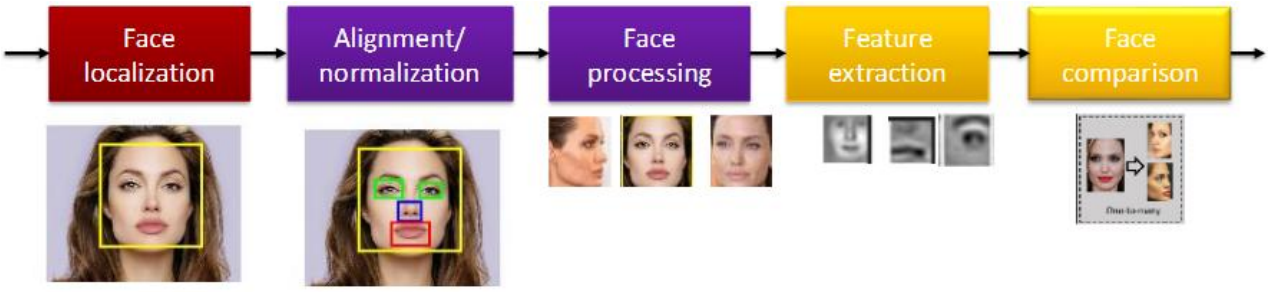
Not only images or video from standard camera.

- Standard RGB cameras
- IR (Infrared): useful because you can rid of biases, which can be related to the ethnicity of the person)
- 3D scanner: I can compensate the different deformation with the expression more efficiently.

FACE RECOGNITION STEPS

Steps of face recognition pipeline: (can be an open question)

1. **Face localization:** segments the face area from the background, you localize where the face is placed in the image.
2. **Alignment/normalization:** normalize the face geometrically to canonical coordinates.
3. **Face processing:** compensate illumination, pose, viewpoints (the idea: get rid of all the possible inferences defined by emotion like smiling, the fact the person looking are the camera or not, etc.)
4. **Feature extraction:** extract salient information that is useful for distinguishing faces of different individuals and is preferably robust with respect to the geometric and photometric variations.
5. **Face comparison:** features are compared against one (verification) or many (identification) face references. Compared with your dataset. Can do many-to-one and one-to-one.



REAL FR SYSTEMS

- Different phases can be merged together (e.g., feature detection and classification in DL-based methods)
- Additional modules can be present (e.g. anti-attack processing)
    - Presentation attack detection (PAD) (alarms are not moving or not blinking)
    - Morphing attack detection
    - ...
- You can do the generation of the feature. Can do many-to-one and one-to-one verification.

Flow: Input Image -> Attack -> Face Detection -> Face Alignment -> PAD -> Face Processing -> Training (Deep NN-Parameter Learning) or Test

Deep Face Recognition:

- Euclidean Distance
- Angular Distance
- Loss Function
- Threshold
- Metric Learning

FACIAL FEATURES

Distinction level of feature depending on the resolution that you have. The resolution is defined by the IPD (Inter-Pupillary Distance) that is the distance between the two pupils of the eyes.

Level 1 features

Gross facial features easily observable (general geometry of the face and global skin color)

- Round/oval face
- Male/female
- Ethnicity
- Low resolution is required (<30 Inter-Pupillary Distance or IPD)

Level 2 features

Localized face information due to structure/position of face components (eyes, mouth, ...)

It's a set of points, which typically are located around the chin, around the mouth, the nose, the eye, the high. These points are facial landmarks, are called the facial landmark, and these are very useful for acute risk detection.

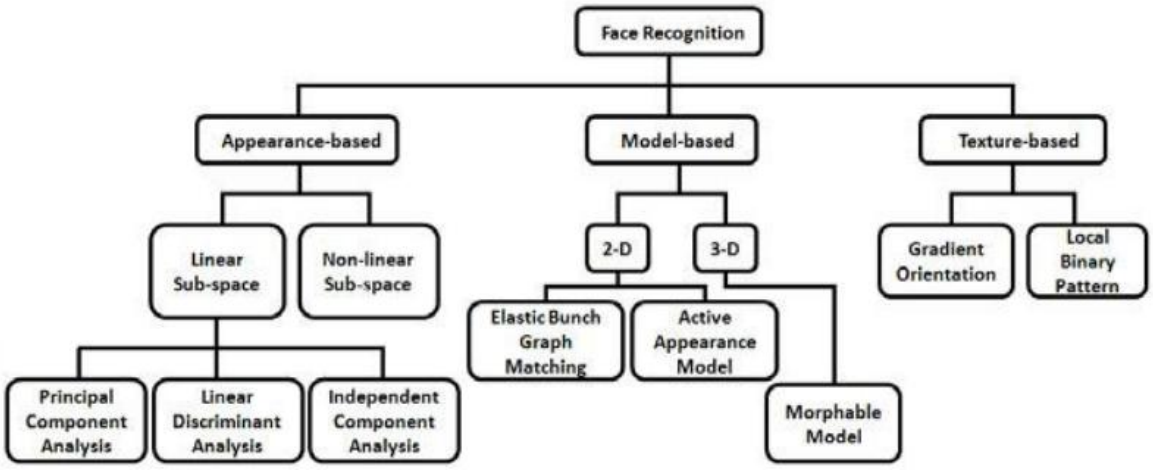
- Gabor, LBP, Shape
- Useful for accurate face detection
- Used to compensate deformation
- Requires higher resolution (30-75 IPD)

Level 3 features

- Unstructured, micro level features on the face (scars, freckles, skin discoloration, moles)
- Requires very higher resolution (>75 IPD)

TAXONOMY OF FACE FEATURES

Divided into 3 main classes: (da sapere)

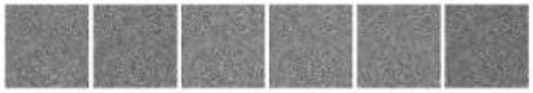


FACIAL FEATURES: GLOBAL (Appearance-based)

The idea: The basis of your face is basically the same and you can represent any face by composing those elements with different amounts (weights).

So, this concept is performed by performing what is the principal component of the composition. You compute S_T (*credo*), which is basically an array of values. And for each array of values, you can compute the scatter values. We do this for each person, and we build an N by M matrix, where N is the number of person in the room, and for each couple you have the covariance.

- One of the first approaches to be proposed was based on Principal Component Analysis (PCA) -> Eigenfaces



- PCA aims at identifying the most suitable basis to represent data
  - $W_{PCA} = \arg\max_W |W^T S_T W|$
- S_T : scatter matrix of pixel-wise covariances between different faces in the DB
- It's a transformation method. The PCA gives you a change of basis (*credo*).

Better alternatives:

- **Linear discriminant analysis (LDA)**
  - $W_{LDA} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|}$
  - S_B : between-class scatter matrix
  - S_W: inter-class scatter matrix
  - This maximization is performed by associating one of the sample that you have from one guy with one of the sample that you have from another guy. You want make them the most separate as possible.
  - I want that all the samples coming from the same person looks the same. This is done by maximize and you want to minimize the samples that comes from different classes.
- **Fisherfaces**
  - $W_{Fisher} = arg\ max_W \frac{|W^T W_{PCA}^T S_B W_{PCA} W|}{|W^T W_{PCA}^T S_W W_{PCA} W|}$

    - It’s a combination, you want maximize the sample from the same class and minimize the possibility of correlation between samples coming from two different classes.
- Independent Component Analysis (ICA) can be used as well

FACIAL FEATURES: LOCAL

Local aspects of the face.

Elastic Bunch Graph Matching (EBGM)

- Image graph representing a face: a geometrical structure consisting of various nodes connected by edges.
- The nodes are located at facial landmarks
- At each location a set of Gabor coefficients are used as features

Neural Networks (NN)

- To get local features
- Different architectures are used to generate a set of features characterizing people's identity and features

Tensorfaces

- Multilinear model characterizing different faces and expressions in a higher order space.

Manifolds

- Faces lie on manifolds; expressions, poses define submanifolds. Their analysis reveals the characteristics of the data distribution (used for dimensionality reduction)

Kernel methods

- Sometimes linear separation is not enough.

Correlation

FACE RECOGNITION STEPS

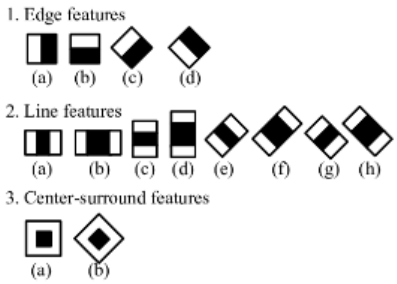
- **Face localization:** segments the face area from the background
- **Face alignment and normalization:** normalize the face geometrically to canonical coordinates.
- **Face processing:** compensate illumination, pose, viewpoints
- **Feature extraction:** extract salient information that is useful for distinguishing faces of different individuals and is preferably robust with respect to the geometric and photometric variations
- **Face comparison:** features are compared against one (verification) or many (identification) face references.

FACE DETECTOR

- A simple filter bank with learned weights applied across the image
- But with some notable performance-boosting implementation tricks… (you don’t need DeepLearning)
- Integral image representation and rectangle features
- Selection of a small but effective feature set with AdaBoost (an algorithm)
- Cascading simple detectors to quickly eliminate false positives.
- You have a set a filter value, then analyze the different facing and then you combine the results of each filter.

CASCADE DETECTION

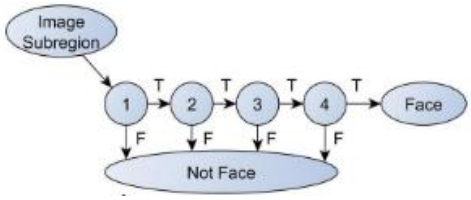
- Cascade classifier: it is a classifier that goes into multiple stages, and in each stage it used a set of filter that verify that in this rectangle there is a face.
- Process subregions of an image
- Test a hypothesis on each area
- If verified continue, otherwise exit



Feature types:

- Features corrispond to filters, they basically testing some charateristic of the image in the rectangle.

1. Edge features
2. Line features
3. Center-surround features

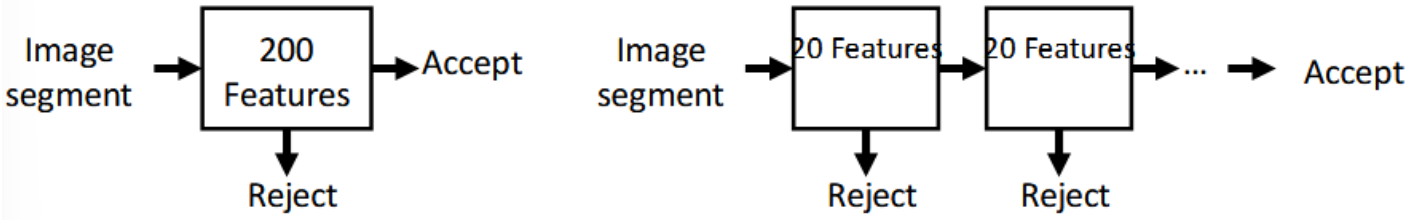


BOX FILTERS

- Faces can be sketched as a set of box filters
- Rectangular filters can be associated to different facial features depending on the type and shapes
- Better response if there are place in a specific location of the face. Maximum response = the filter is centered on the facial features that aims at modelling (depending on the type of filter (*credo*)).
    1. Edge Features
    2. Line Features
    3. Four rectangle Features

CASCADING DETECTORS

- Instead of applying all 200 filters at every location in the image, train several simpler classifiers to **quickly eliminate easy negatives**.
- Each successive filter can be trained on true positives and the false positives passed by the filters before it.
- You have multiple steps and you have to pass all the small checks in order to get to the end, this simplify the performance.
- The filters are trained to allow approximately 10% false positives.



CASCADE IMPROVEMENTS

The cascading features provide comparable accuracy, but ten times the speed.
This is how Viola Jones works (*something like that*).

HANDLING VARIATIONS

- Variation in the sense that the same face can look different depending on different factors.
- Handle pose, illumination, orientation and expression changes (occlusions as well)

- Heterogeneous face recognition (different media)

Different methods to deal with it:

Synthesis method

- A synthetic visible photograph from the alternate face image format (a synthetic face)

Feature-based method

- Encode face images using feature descriptors that are largely invariant to changes between the two domains (e.g., SIFT and LBP can be invariant from pictures and sketches - select the most discriminant features e.g. LDA)

Prototype-similarity method

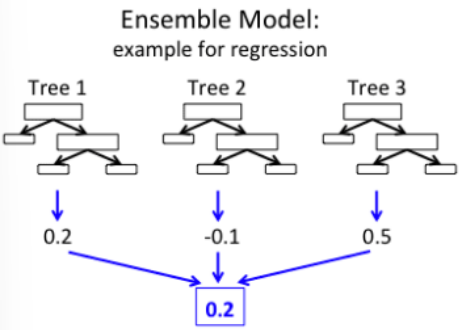
- Represent the face image as a vector of similarities to the images in a prototype database.
- Requires a set of images available in format A (source) and format B
- It's like, if you characterize the identity of the person with respect to the resemblance, within the two domains, you hope that the resemblance pattern that you have for the sketches is the same resemblance pattern that you have for the pictures.

FACE ALIGNMENT/NORMALIZATION

- Poses and expression are the major problem because they introduce variation in the images.
- Faces can be taken from different viewpoints.
- Different expressions/emotions can impair the FR systems. (take a picture for each expression)
- Given the expression of the person, I will re-bring the point in the correct position. (*credo*)
- Facial landmarks need to be detected and used to align the acquired image to a neutral expression/frontal pose
- Using morphing algorithms can be used to modify the expression.

ENSEMBLE OF REGRESSION TREES

It's a classifier. The tree take a subset of the pixel around the face and than it try to localize the position. You do multiple operation because every iteration refines the parts. The dlib library implements a shape predictor algorithm with an ensemble of regression trees (ERT).



- Examines a sparse set of input pixel intensities (i.e., the "features" to the input model)
- Passes the features into an Ensemble of Regression Trees (ERT)
- Refines the predicted locations to improve accuracy through a cascade of regressors

LANDMARK EXTRACTION

- Facemark: OpenCV API for facial landmark
- Other solution different from Dlib:
  1. **Kazemi solution** (Dlib, Facemark): Generate a cascade classifier that approximated face deformation.
  2. **AAM** (Facemark) - Pantic's approach: Fast optimization over AAM.
  3. **LBF** (Facemark) - Ren's approach: Use progressively-refining LBF.

FACE MODELLING

- **Active Appearance Model (AAM)** is a statistical model of the facial appearance generated by combining shape and texture variations.
- Here we have that the points of the landmark p_j (the local landmark) can be modeled as a mean value (which is p_μ) + a modification ($E_s w_{s,j}$).
- Naming p_j a fiducial point in image I_j, after applying PCA we have: $p_j = p_\mu + E_s w_{s,j}$ where is the generic shape and is the eigenvector matrix.
- As for texture, same logic, after warping a similar image, the pixel patch g_j can be modelled as (g_μ is the mean texture): $g_j(p_j) = g_\mu + E_g w_{g,j}$
- Compute $w_j = (w_{s,j}, w_{g,j})$ so that is mapped to a reference model g^m

$$\omega_j = \min_{\omega} | g_j(p_j) - g_j^m |$$

- You want optimize these two elements so that the appearance of the image around that point is very similar to the average appearance around that point.

DEEP LEARNING SOLUTIONS FOR DEFORMATION MODELLING

- Use of a cascade of convolutional layers -> CNNs
- We have several networks that localize different parts of the face and each part may be related to some set of points with respect the others.
- 3 stages (each stage used different type of network) by Sun et al.
- Other works distinct inner and outer landmarks
- Other approaches: 2 stages
- Stacked hourglass structures used as well
- Investigation on bias and variance has been carried on

FACE EMOTION RECOGNITION

- FER uses dlib features as well

Usage:

- **Security:** Prevent violence and improves the overall security of a place.
- **HR assistance:** Determines whether the candidate is honest and truly interested in the position.
- **Customer service:** Determine how satisfied is a client.
- **Differently-abled children:** Help autistic children interpret the feelings of people around them.
- **Audience engagement:** Audience's emotional responses.
- **Video Game Testing:** Gain feedback from the user.
- **Healthcare:** Know if a patient needs medicine or whom to prioritize.

FACE PROCESSING

- **One-to-many (or training data augmentation):** Generates many different patches/images from a single one changing rotations, pose, in order to increase the dataset.
- **Many-to-one:** Bring each face back to the canonical view (controlled conditions)
- But you need to represent your face using a feature. And therefore we have different solutions that typically now are using in learning, and goes back from 2013, up to the recent days.

FEATURE EXTRACTION/CLASSIFICATION

- **Problem:** Ot's really difficult to have a large dataset. If you want to train neural networks, you need to have lots of sample in order to be robust.
- **Solution:** Having a network that generalizes well the proportion of the face, the characteristics of the eyes, nose etc. so that it is possible to represent every face of every person on the Earth. The larger is the dataset better it is. Now day is possible using large network that are train for general detection and that perform a fine-tuning on the topic you want (*credo*).
- For most applications, it is difficult to include the candidate faces during the training stage ("zero-shot" learning task).
- Faces share a similar shape and texture: the representation learned from a small proportion of faces can be generalized.
- Datasets make a difference
    - Industry datasets (Google, Facebook, …) ~ $10^6 - 10^7$ data subjects
    - Academic datasets ~ $10^3 - 10^5$ data subjects
- Deep learning is nowadays widely used to generate a feature vector
- Backbone networks fine-trained on faces
- Multiple backbone can be used

Loss functions

- Euclidean
- Angular
- Softmax

ATTACK TO FR

Face presentation attack

- Impersonation attacks (mimic a specific character)
- Evasion attacks (don't want to be recognize)

Types:

- **Make-up Attacks:** Some specific make up that may deceive the recognition system; difficult to detect even for a human supervisor.
- **Photo Attacks:** Presenting a photograph of the genuine user (printed, displayed on screen); picture could have been taken with a digital camera or from internet. (here for defending you can check if the person moving or blinking)
- **Video Attacks (also called replay attacks):** The attacker replays a video of the genuine client using a digital device (e.g., mobile phone, tablet or laptop); more difficult to detect, as not only the face 2D texture is copied but also its dynamics.
- **Mask Attacks:** The presentation artefact is a 3D mask of the genuine client's face; depth cues and shadows are not useful in this case; less common than the previous two categories in research publications.

ES. FACE RECOGNITION ISSUES

1. Buolamwini, Gebru 2018: Gender shades
2. Champions League 2017 Final in Cardiff: 2470 possible suspects, errors 2297 (92%)

FAIRNESS (lo ha letto al volo pero lo lascio nel dubbio)

- **Group fairness:** Both samples from unprotected and protected groups have an equal probability of being assigned to a correct prediction.
- **Fairness through awareness:** Any two individuals with a certain similarity metric (defined for a particular task) should receive a similar outcome.
- **Fairness through unawareness:** Any sensitive attributes S are not explicitly used in the decision-making process.
- **Counterfactual fairness:** Classification remains the same in a counterfactual world where the individual belonged to a different demographic group.
- **Fairness in relational domains:** Considers social, organizational, and other relational connections in addition to individual attributes.

METHODS TO RECOVER FAIRNESS (questo invece l'ha approfondito un poco)

Pre-processing

- Creating a balanced set: Enlarge (new) dataset
- Data augmentation: Slightly modified samples (noise)
- Resampling: Oversampling, Undersampling
- Synthetic data generation: VAE, GAN
- Unawareness: Exclude sensitive attributes

In-processing

- Cost-sensitive learning: Penalize errors on minorities
- Domain-adapt. methods: Leverage knowledge from unbiased groups
- Disentangl. Methods: Remove sensitive info prior to decision

Post-processing

- Modify regions

LECTURE 21 (IRIS AND GAIT RECOGNITION)

IRIS RECOGNITION

- **Iris:** Regulates amount of light enter into the eye
- **Texture:** Due to an agglomeration of multiple anatomical entities (multilayered structure)
- **The eye structure:**
    - **Anterior border layer:** Increased density of chromatophores (i.e., pigment (color) containing cells).
    - **Posterior layer:** Typically the iris is referred to this layer. Two thick cells contains heavily pig pigmented epithelial cells - impenetrable to light.
    - **Muscle layer:** Contract and dilate the pupil.
    - **Stromal layer:** made of collagenous connective tissues and blood vessels (radial).

IRIS RECOGNITION

Infrared image of an eye features:

- Contraction Furrows
- Radial furrows
- Zig-zag lines

we can define the design of the iris in different areas, which are kind of concentric, so they start from the tubular, then they get farther and farther. We have that this iris is characterized by furrows, which are basically the equivalent of the iris of ridges. We have two main boundaries: the Pupillary Boundary and the Limbus Boundary, they defined the area we used.

Uniqueness:

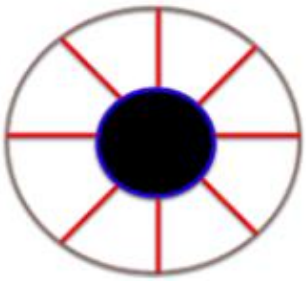
- Uniqueness of each iris is a consequence of the random morphogenesis of its textural relief during prenatal growth.
- Stable within a few months after birth.
- (Irises of monozygotic twins are different.)

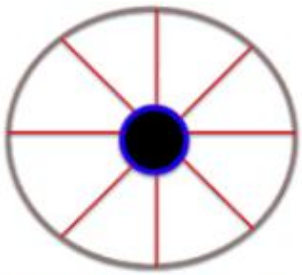
Color:

- Determined by the pigmentation (melanine granules, density of cells in the stroma).
- Not crucial for recognition: texture structure is a distinctive element. (we don't care, we care about the texture)

IRIS MUSCLES

- Iris size is controlled by **sphincter** (circular) and **dilator** (radial).

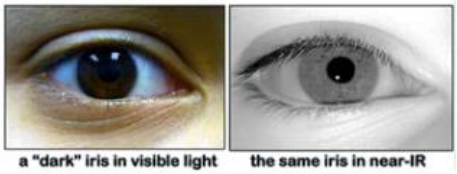




Reduce: sphincter contract, dilator relax

enlarge: sphincter relax, dilator contract

- Iris is usually illuminated in near-infrared band (visible 380-780 nm, near-IR 700-1400 nm).
- Example commercial iris sensor: 770 + 870 nm LEDs.
- Visible light vs Near-IR: Usable textures for dark iris in Near-IR (700-900nm).
- The color image of your eyes can create problem, we solve this using a Infrared camera. Infrared are reflected by the lower layer of the iris area, this mean the top layer (the layer with the color) is bypass.



STEPS OF AN IRIS RECOGNITION SYSTEMS

1. **Camera acquisition**
2. **Localization and Normalization**
3. **Image Enhancement**
4. **Feature extraction**
5. **Compare (with the Database):** Comparison implies computing a match score. A threshold decide if two images match or not (Yes/no).

ACQUISITION

- Usually implies a collaborative subject.
- It is very similar to a standard camera.
- Also works within a 6- to 12-meter radius.
- SEEK II allows military to scan iris using portable device.

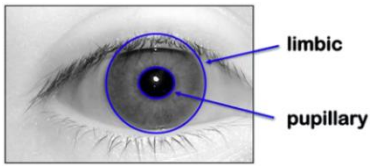
ACQUISITION AND SEGMENTATION

Acquisition

- Acquisition systems are usually made by a CCD sensor (monochrome image using near-IR light) and a near-IR illuminator (illuminate the eye on that specific frequency).
- Requires the cooperation of the subject.

Segmentation

- Localizes the spatial extent of the iris in the image of the eye by isolating it from other structures (sclera, the pupil, the eyelids, eyelashes): find inner (pupillary) and outer (limbic or limbus) boundaries. So you localize the two boundaries. For localizing the iris we assume that is a perfect donut.
- Eyelashes and eyelids can partially interrupt the contour.
- Crucial part in recognition!



SEGMENTATION

- Traditional segmentation uses the integro-difference operator to find the parameter of the limbic boundary:

$$\max_{(x_0,y_0,r)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{x_0,y_0,r} \frac{I(x,y)}{2\pi r} ds \right|$$

a. Then image pixel are integrated along a circle centered at (x_0, y_0) with radius r.
b. Then gradient is computed along the line connected to the center (operator $\partial/\partial r$).

c. Data are convolved with a Gaussian filter with radius r and scale σ (ensure that sharp edges corresponding to crypts, freckles, and furrows are reasonably blurred).

- So you're moving around the radius, so with increasing radius, and you are basically checking the value of the radius when you get the spark.
- $G_\sigma(r)$ is used for performing a smoothing. (sharp the edge, avoid the noise)
- Find the maximum in the blurred partial derivative with respect to increasing radius r, of the normalized contour integral of $I(x, y)$ along a circular arc ds.
- A similar approach is assumed for pupillary boundary.
- This maximization gives the center of the boundaries and the radius of the boundaries.

SEGMENTATION

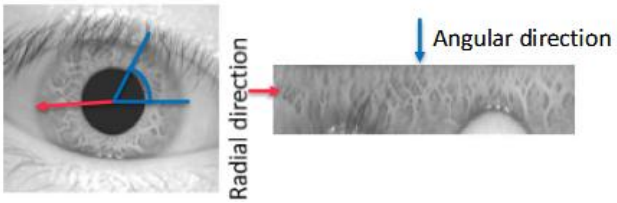
- Choose x_0, y_0, r that maximize:
    - Gaussian smoothing over of
        - Difference with respect to of
            - Average pixel intensity along (portions of) circle
- **How to find the center?**
    - Bounding box of the largest dark region might be a range to search for , with .
- **Non-circular pupillary and limbic boundaries:**
    - Fitting ellipses.
    - Fitting circles and adjusting using active contours
    - using "balloon" active contours.

NORMALIZATION

What you have done until now? You have basically localized your iris.

Now you move from Cartesian coordinates to spherical coordinates.

Unwrapping the iris: iris texture within the annular region from cartesian coordinates to pseudo polar coordinates via a rubber sheet model.



- Account for variations in pupil size.
- Common image domain.
- 2 norm. irises can be registered by translation.

Pixel belonging to eyelids and lashes are labelled with "0"s (segmentation mask).

Photometric transformation are followed to enhance the visibility and detectability of the iris.

You can exclude pixel with details that are not important (for example eyelashes). We do a bayer mask (*credo*) so we know which pixel consider and which not.

The iris image it is called Rubbership (*credo*) image.

ENCODING AND MATCHING

- Convert iris in a set of features.
- Traditional approach: Gabor filters

- Phasor are extracted (magnitude is not considered): each phasor is coded with 2 bits (depending on the quadrant of the complex plane where it is lying).
- Deep features used as well.
- **Matching**: with binary features hamming distance can be used (xor between binary elements + sum).

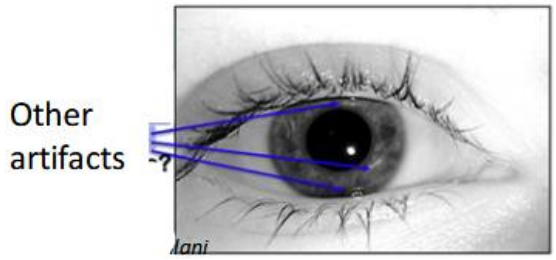
UNWRAPPING AND TILING

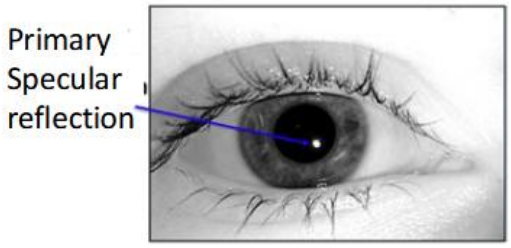
- Convert into pseudo-polar coordinates.
- **"Rubber sheet" model**: linearly stretching or compressing the imaged iris to a standard size frame.
- Texture filters can then be applied at a fixed grid on the rectangle to generate the same size iris code for any image.
- Binary mask tells which parts need to be avoided (because they are occluded) and which part needs to be kept.

IRIS MASK

How you generate the ask? The mask need to get rid of occlusion (pixel that are covered by elements that are not part of the iris)

- The "mask" occlusion due to:
  - Eyelids
  - Specular reflections
  - Eyelashes
  - Strands of hair
- Example: eyelids detection: search for a parabolic edge within the region defined by the outer circle (spline-fitting); searching for strong near-vertical edges in the segmented iris.

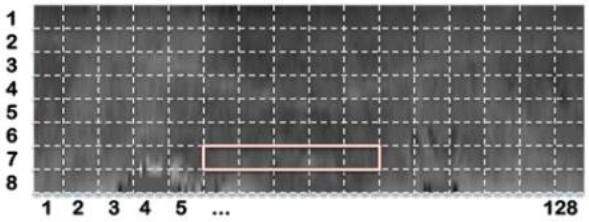




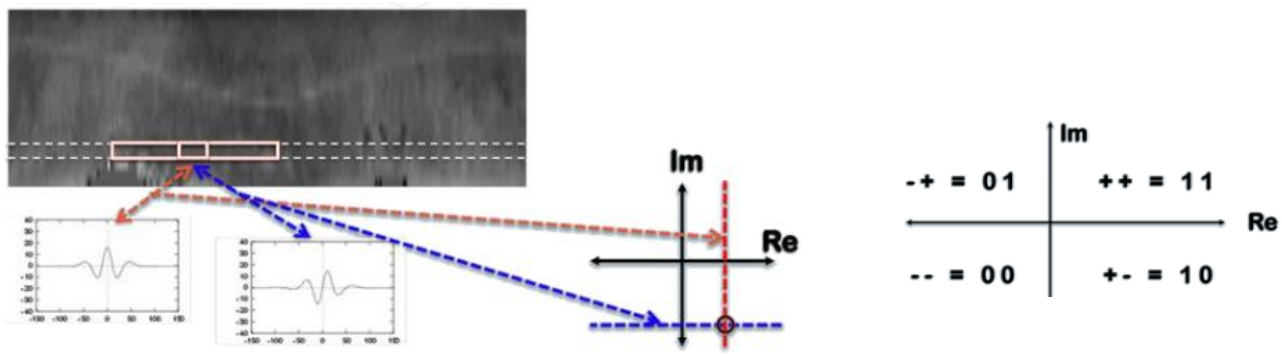
- Which of these elements effect the iris code (*credo*)? Primary specular reflection
- Primary specular reflection (in the pupil).
- Other specular artifacts can be more subtle.
- Other segmenter for hairs and eyelashes.
- You generate a segmentation, given this segmentation you generated your mask.

COMPUTING THE IRIS CODE (1/2)

- Daugman applies a Gabor filter on a MxN point grids.
- Gabor filter generates a couple of values (real and imaginary part of the filter response): point in complex plane.



- Then according to this value (real and imaginary) you get a two bit representation for each of these *tiles*. You can decide how many *tiles* you want to have.

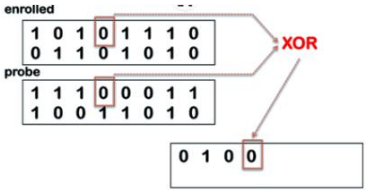


COMPUTING THE IRIS CODE (2/2)

- Possible relevance to human vision: "Excellent description of the 2D neural receptive fields found in the visual cortex" (Daugman).
- "Those same receptive fields that are so nicely modeled with Gabor filters may equally well be described in terms of kernels called 'difference of offset Gaussians'..." (Snyder).
- Good theoretical properties.
- The standard commercial iris code is 2,048 bits (plus 2048-bit mask).
    - 2,048 / 2 = 1,024 sample points.
    - 1,024 = 8 radii x 128 samples is a plausible sampling configuration.
    - Where 128 samples are angular direction, 8 samples are vertical direction.

FRACTIONAL HAMMING DISTANCE

- You want to compared with your dataset. How do you do? (we have binary strings)
- It is possible to compute the similarity with fractional Hamming distance, FHD
- We use XOR because we want a distance.
- Fraction of bits that differ, i.e., fraction of 1 bits
    - FHD = 0 exactly equal
    - FHD = 0.5 uncertain
    - FHD = 1 completely different

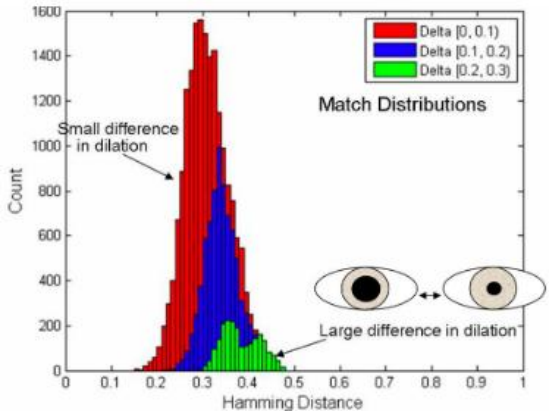


- Your fractional distance can be seen as a set of hand operations between the code of the first iris, the second code, and also a hand between the mask (*credo*). So I can perform the computation on those bits that are both available for both masks.
- In case there is a mask for each code (M1, M2)
    - $$\frac{[(M1 \text{ AND } M2) \text{ AND } (Code1 \text{ AND } Code2)]}{[M1 \text{ AND } M2]}$$
- Head tilt can change the FHD value
- FHD is computed for a range of circular shifts, and the min FHD is kept.
    - corrects the authentic distribution
    - increases time for matching
    - changes the impostor distribution.
- What is most affecting by this is the impostor, so the results that you have on irises that you do not want to authenticate.

PUPIL DILATION

If the pupil il dilated you have less useful bits.

- Dilation ratio = $\frac{\text{pupil radius}}{\text{iris radius}}$
- Normalization can compensate change in:
  - ambient lighting
  - emotional state
  - health
  - medications
  - decision-making !!



- Tou can see that the dilatation is changing the authentication distribution (*credo*)
- Increased difference in dilation shifts the authentic distribution. No change in impostor.
- Larger pupil dilation increases the mean of the authentic distribution.
- Larger pupil dilation decreases the mean of the impostor distribution.

WHO'S USING IRIS SCANNERS

- the U.S. military has used iris scanning devices to identify detainees in Iraq and Afghanistan
- The New York City Police Department installed B12 Technologies' mobile MORIS (Mobile Offender Recognition and Information System) in the fall of 2010.
- Rhode Island Department of Corrections, Orange County and Los Angeles County had plans to implement iris scanning technology.
- United Arab Emirates, one of the largest users of iris recognition for border control.
- Fujitsu NX F-04G, Vivo X5Pro, Samsung Galaxy S8
- Iris recognition devices are currently being installed in every sheriff's department along the U.S.-Mexico border.

MYTHS ABOUT IRIS RECOGNITION

- Iris characteristics are genetically determined.
- Using Laser scans.
- Scanners acquire the back of the eye.
- Iris and retinal scanning are the same.
- Iris can reveal or help diagnose medical conditions.
- Someone could gouge my eye out and thus steal my ID.
- Taking eye-dilation drops can let me register with a different identity.
- Lenses and sensors needed for iris recognition are too big to make it viable in mobile devices.
- Iris recognition are too power-hungry or compute-intensive for smartphones.
- Effectiveness change with color/age.

ADVANTAGES

- accuracy and reliability ~ten times more accurate than fingerprinting
- fingerprints are constantly exposed and susceptible to damage, the iris is naturally protected by the cornea (the eye's transparent front coating)
- seems to remain reliably unchanged for decades (though not necessarily for life).
- iris scans can be performed safely and hygienically at some distance from the eye.

IMPAIRING ELEMENTS

No biometrics is foolproof

- Medical conditions. (acute iris inflammation causes systems to fail)
- Iris recognition in crowd - inaccurate 1 to 10% of times
- poor subject presentation (closed eye, rotated iris)
- capture environmental problems (illumination, blur, ...)
- Image processing or storage (compression, corruption)
- Unusual characteristics of the individual (abnormal shapes)

Missing rate goes from 2% to 20%.

CONTACT LENSES

Typically system can check that you have contact lenses, if you have the system ask to take the contact lenses off.

Contact lenses do result in visible artifacts in iris images

Transparent lens / AccueVue cosmetic lens / Fresh Look lens.

- Evade detection: intentionally create a false non-match (easy to do)
- Create a synthetic identity (maybe)
- Impersonate a targeted identity (maybe, but even harder)

Detect contact lenses analyzing Fourier spectrum

- Clear lenses increase the FNMR slightly; a minor social impact issue.
- Textured lenses are a security issue; automatic detection needed.
- Detection of textured lenses seen in training data is a solved problem.
- Detection of new textured lenses is not a solved problem.

IRIS TEMPLATE AGING

- Longer time intervals generally make it more difficult to match samples to templates due to the phenomenon known as 'template aging'.
- changes in the biometric pattern, its presentation and the sensor.
- Initially thought to be immutable

IRIS CHANGE

- From medical literature of the same time (1994), we know that the iris changes functionally with increased age.
- Smaller average pupil size with age (so greater time lapse translates into greater average difference in dilation, degrades iris match scores.
- shape of the cornea changes.

OTHER ATTACKS

- Presentation attack (uses pictures): hacking of Samsung Galaxy S8 systems
- databases of iris biometric are a honeypot of sensitive, highly personal data that will be targeted by criminals.
- Data breaches and hacks are at an all-time high.

GAIT RECOGNITION (using in camera surveillance in China right now)

OVERVIEW OF GAIT ANALYSIS STRATEGIES

"Gait analysis is the systematic study of animal locomotion, more specifically the study of human motion, using the eye and the brain of observers, augmented by instrumentation for measuring body movements, body mechanics, and the activity of the muscles."

- Medical analysis and diagnostics
- Rehabilitation and sport training
- Product design
- Animation and CGI
- Security
- Forensic analysis

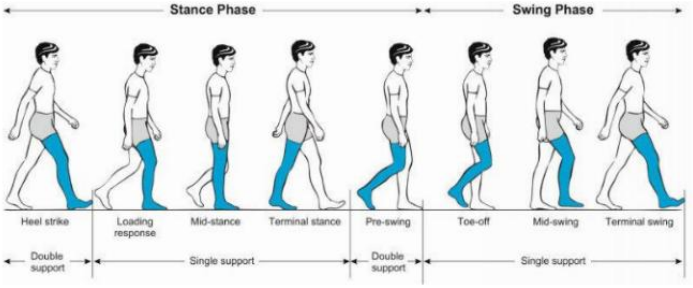
} **Biometrics**

Influenced by:

- Footwear,
- Fatigue,
- Use of knee brace or other rehabilitation devices,
- Sex,
- Speed,
- Disorder/Medical condition/Disease,
- Age,
- Directed gait,
- Location, lighting, impairing conditions,
- Camera,
- First trimester during pregnancy.

PHASES OF GAIT ANALYSIS STRATEGIES

- Human walking occurs in a specific pattern, including various stages: **gait cycle**, defined by two phases: **stance** and **swing**.
    - **Stance Phase**(feet touch the ground)**:** initial contact, loading response, mid-stance, terminal stance, pre-swing
    - **Swing Phase**(feet is left from the ground)**:** initial swing, mid-swing, terminal swing.



OVERVIEW OF GAIT ANALYSIS STRATEGIES

Different systems can be employed to acquire human motion

- **Wearable sensors:**
    - IMU (Inertial measurement unit): accelerometers, gyroscopic sensors, magnetometers, force sensors, extensometers
    - electromyography
    - Active markers (cinema use it whit special suits)
- **Non-wearable sensors:**
    - Based on pressure sensors

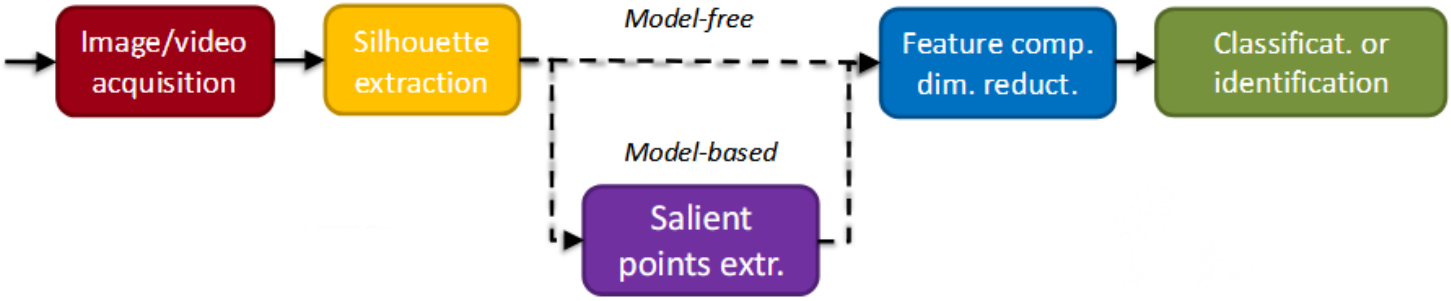
- **Camera based:**
  - Standard RGB camera
  - Multi-camera systems
  - Using visual markers
- **3D scanning:**
  - Depth camera
  - RF 3D sensors
  - LIDAR sensors (point clouds)

GAIT SIGNAL EXTRACTION

- **Pressure-based systems:**
  - Records timing, pressure, striking points and stance variations
  - Possible features: footprints, sampled transitional footprints
- **Sensor-based systems:**
  - Model human body as a set of points linked by segments (skeleton model)
  - Sensor must be placed in proper points
  - Time-series from point coordinates (absolute or relative, x,y,z, angles, …)
- **Vision-based systems:**
  - Extract silhouette of person walking
  - Fit skeleton or process silhouette directly

GAIT IDENTIFICATION SCHEME

Four main phases:



- 2 type of methods to perform analysis:
  - **Model-free:** we analyze the frames, you create the information directly on the silhouette value.
  - **Model-based:** you know you can simplify the proportion and the shapes of a human body by simple elements, so we have a salient point extraction
- Image/video acquisition (RGB, depth cameras, RF scans)
- Segmentation and extraction of people silhouette
- Extraction of salient points (joints and bones model)
- Computation of time series for each joint position/relative angles for bones
- Extraction of features from time series / dimensionality reduction
- Identification/Classification system: compare the feature in the database

SILHOUETTE EXTRACTION

- Conceptually similar to face/fingerprint segmentation strategy seen in the past
- Background subtraction methods are used to isolate people:
    - Motion compensation in video, correlation estimation, classification
    - Traditional methods: gaussian/mixture filters, kernels, LBP
    - DL-based methods: CNNs, Self-Organizing Background Subtraction (SOBS) or Maps (SOM), based on Bayesian models (BNN)

  → Then, people=blobs are clustered
- Other solution identify persons in the image (bounding box) + background information + silhouette extraction
- Bounding boxes found with traditional CV methods (Haar or HOG + classifier like SVM) or CNNs
- Pixels in the bounding boxes are labelled (person/not-person)

Issues:

1. Shadows
2. Threshold bg/fg
3. Moving objects in background
4. Compression artifacts

MODEL-FREE MODELLING

- Work directly on the silhouette
- Motion-Energy Image - MEI and Motion-History Image – MHI (given a silhouette you perform the short operation on the silhouette of the person walking (*credo*))
- Motion Silhouette Image - MSI
- Gait Energy Image - GEI (uses the average) and Gait History Image - GHI (History models the difference in time)
- Forward (f) and backward (g) Single-step History Image (fSHI and bSHI)
- Active Energy Image – AEI (h)
    - with bag (i)
    - with coat (j)
- Distance-based Features
- Centroid-based Features
- Learned features
- **The key:** there are multiple energy images connected to the silhouettes, which are obtained by combining multiple frames from a silhouette together, then you can apply neural network on this.

MODEL-BASED: FITTING SKELETON IN SILHOUETTES

- If you fit the skeleton you don't need the full image.
- Approaches can process multiple images/silhouette (single silhouette, multi-views, 3D poses)
- Approaches started back in '80s
- **Two classes of methods:**
    - using a 3D model: a 3D skeleton with some anthropometrics constraints (e.g., shape from silhouette)

    - process 2D frames from the camera: train a regressor to place the coordinates of joints given shape and color information
- Fitting a model into a silhouette: decompose silhouette into rigid parts and associate each part to a bone and connect each bone with different joints.
- DL-based regressor: PoseNet (localize joints from color images)

STEP PERIOD EXTRACTION

- Extract the signal corresponding to one step
- It can be related to vector intensity, acceleration, orientation
- **Basic SP pipeline:**
    1. Signal preprocessing: upsampling and filtering
    2. Template extraction: the signal relative to a step will be obtained using template matching (to make this independent from the orientation)
    3. Template matching: correlated template and signal to find minima (beginning and end of step). (every time you have high correlation you have one step)
    4. Reference system normalization: transform all the signals into a standard reference system for the whole duration of the step.
    5. Sample normalization: data points do not have the same length (e.g., walking speed can change) so reinterpolation is required to format data in the same way.

ADDITIONAL STEPS TO IMPROVE DETECTION

- filtering with very low cutoff frequency to retain only the most evident periodic signal
- template update function
- Classify each interval

SETTING FEATURES AND CLASSIFICATION

- Coordinates (or force values) are organized into temporal series
- Normalization is required; outliers removal
- Features are then extracted
    - Hand-crafted
    - SOM
    - CNN
    - RNN
    - dimensionality reduction (LDA, PCA)
- Final classification can be done with traditional ML methods (or DL)
    - SVM, DTW, HMM
    - CNN
    - DNN

EXAMPLE OF CLASSIFIER

- Take interval of samples x_i
- Build couples (x_i, y_i)
- Train a classifier e.g. CNN
- More complex classifiers, more accurate
- After feature extraction, you can apply more elaborate regularization and classification

- Authentication can be performed on One-Set or Closed-Set setups
- OneClass SVM for continual authentication

ISSUES IN GAIT IDENTIFICATION

- Identification of a person or a set of people.
- Estimation of some physical characteristics of persons (age, sex, ...): soft biometrics

Environmental issues (more evident in outdoor environments)

- Pose: solved by multi-camera systems
- Illumination: infrared systems
- Occlusions (e.g. subject carrying a backpack): thermal camera can solve

Viewing issues

- perspective with respect to the camera: attenuated by geometric transformation

Variations in user characteristics

- Users' gait can change due to different physical and psychological issues: re-enrolment

LECTURE 23 (AI FORENSICS)

VOICE RECOGNITION

- Often different types of interactions take place using voice only (contracts, bank communications, ...)
- Is it possible to discriminate whether a given audio signal is authentic?
- It is possible to infer other information concerning the state/emotional condition of the person.
- **Applications:**
    - Voice authentication (access control) and background checking (natural voice checking)
    - Speaker detection (surveillance)
    - Forensics speaker recognition (voice as evidence)
- **Two approaches:**
    - **Text-dependent:** users utter a specific passphrase (usually randomized)
    - **Text-independent:** mostly used for speaker detection and forensic speaker identification (challenging)

VOICE CHARACTERISTICS

Speech production is extremely complex.

Sociolinguistic factors (e.g., level of education, linguistic context and dialectical differences) Physiological issues (vocal tract length, shape and tissues, configurations of the articulatory organs)

Two different feature levels: high-level (linguistic) and low-level (acoustic)

High level characteristics

- Idiolectal (related to specific linguistic system)
    - depends education, society, family context, town of birth
    - environment (@work @home)
- Phonotactics (use of phone units and realizations)

Low-level

- Prosody. Features that provides speech with naturalness, full sense, and emotional tone
    - instantaneous energy
    - intonation
    - speech rate
    - unit durations
- Spectral related to individual articulatory actions and to the individual physiological configuration (Static and dynamic)

VOCAL TRACT

Vocal tract can be modelled as a set of filters

- Vocal tract frequency response
- Glottal pulse
- Speech signal

FILTERBANKS STRUCTURE

Filterbanks approaches are widely-used for speaker recognition.
It is a generic term which refers to the class of methods that process on multiple frequency bands of a given signal (subband processing).
Suppose that the input signal is x[n] with FFT X[i], the output is $Y[i] = \sum_{j=1}^{N} X[i]H[j]$

- i.e. output of filter is a weighted FFT
- Sub-band feature extraction
- Full-band feature extraction

MEL FREQUENCY CEPSTRAL COEFFICIENTS (MFCC)

Features are mostly extracted from frequency domain MFCC are among the most widely-used features for voice verification (and not only)

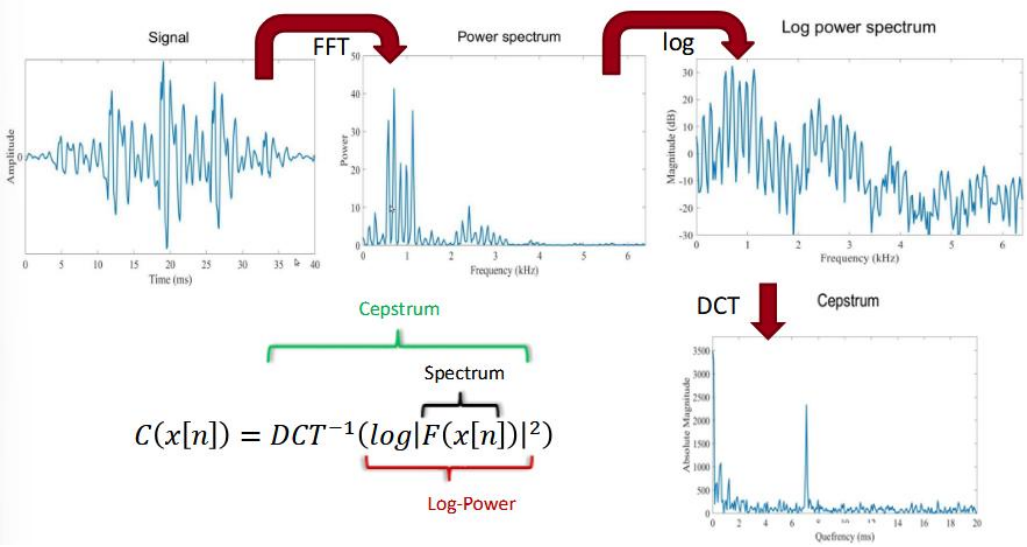
1. Segment is divided into overlapping segments
2. Frequency response is computed for each segment (with a hamming window h[n])

$$X(f) = \sum_{n=0}^{N-1} h[n]x[n]e^{-2\pi\frac{fn}{N}}, \quad f = 0, \dots, N-1$$

3. Compute the power of the spectrum $|X(f)|^2$
4. The output is then processed with a bank of filters centered uniformly on a Mel scale
5. Log of coefficients
6. Performs the DCT of log power coefficient as if it is a signal (DCT $\simeq$ FFT)
    - creates a more compact version, simplified FFT
    - creates real value coefficients

CESPTRUM EXAMPLE

Features are mostly extracted from frequency domain



COMMENTS ON MEL-CEPSTRUM COEFFICIENTS

Use first 12-13 coefficients
Computation of delta coefficients (1^{st} and 2^{nd} derivative, Δ and ΔΔ), e.g. $\Delta C_k(x[n]) = C(x_k[n]) - C(x_{k-1}[n-1])$
Total of 39 coefficients

Then processed by a variety of transforms (reduce their sensitivity to handset variations or extend their expressive power)

- computation of delta coefficients (and derivative)
- cepstral mean subtraction (CMS)
- coefficient stream filtering (Hermansky and Morgan, 1984)
- histogram-based normalization (Pelecanos and Sridharan, 2001)

Conditioned on word or phone identity obtained from an automatic speech recognizer (ASR) and modeled separately.

SPEAKER MODELING

Two approaches to estimate the class-conditional feature distribution

- Parametric (stochastic): fit a specific distribution/estimate parameters
- Non-parametric (template)

Most popular approaches for text-independent identification:

- Vector quantization
- Gaussian Mixture Model
- SVM
- Deep Neural Networks

OTHER BIOMETRICS

- Eyes - Retinal scan
- Eyes - Scleral vein
- Ear
- Vein
- Odour
- Finger and palm vein
- Headprint
- Signature
- Keystrokes
- DNA

FORENSICS AI

THE ADVENT OF ML

- Large development of AI tools during the last years
- New tools for manipulation of multimedia data
- Rise of generative AI: use PCs to create contents

New threats posed to cybersecurity

- AI Malware
- Attacks to smart assistant
- AI password guessing and CAPTCHA breaking
- AI-based encryption
- AI hacking and cryptocurrency trading

- Impersonation attack
    - voice cloning
    - face cloning
    - online game cheats
    - social account impersonation

WHAT ABOUT MULTIMEDIA?

- Social engineering at scale
- Content creation
    - Fake image creation Deepfakes
- Content parsing
- Robocalling
- Attacking image recognition in autonomous systems (cars, drones)
- Escaping image recognition systems
- Fraud in voice and face recognition
- Privacy violation

CONTENT PARSING AND ROBOCALLING

Detect the most relevant information while scanning the data on a server: parsing malwares
Named Entity Recognition (NER): identify the semantically significant parts and pieces of individual information (phone numbers, address)
Extract important sensitive information from documents
Similar to **robocalling**: profiling users in order to find out the most effective strategy to make the attack successful.

ATTACKING IMAGE RECOGNITION

Social Network Analysis and Propensities (SNAP): analyze personal social network behaviors to determine, e.g., the likelihood of a particular action being taken or the role of an individual.
Highly rely on image recognition: masking activity could prevent such matching.
Using adversarial attack for object detection: fooling face recognition or object detection.

GENDER BIAS AND PREJUDICES

- Cognitive bias
- Missing data
- Uncontrolled creation
- Missing backtracing
- Is this logic?
- Size does not imply diversity

"Feeding AI systems on the world's beauty, ugliness, and cruelty, but expecting it to reflect only the beauty is a fantasy." (Birhane and Prabhu)

CONTENT GENERATION

Rise of ChatGPT-like services

- Generating a full, human-sounding text of a simple title
- Turning the textual description of an application into working code
- Changing the writing style of a text while maintaining the content
- Passing the Turing test for a human-sounding chatbot

Create material for frauds

Also for images, video and 3D model Google's Imagen Video, DALL-E, MidJourney and Stable Diffusion.

Rise of deepfake

MIMICKING USERS: IMPERSONATION ATTACK

Profiler can build models that mimick the writing style of people: impersonate individuals in order to access bank account or keep fraudster accounts alive.

It is possible to do it with voice and faces as well: deepfakes.

Identity theft and fraud combined with other crimes.

FRAUDS ON VOICE AND FACING FOR ACCESS CONTROL

Synthetic image, video and audio sequence generation given and synthesis model that is tuned on a specific person.

- Audio deepfake to access bank account
- Video deepfake for defamation in political campaigns
- Revenge porn
- Accessing biometric protected premises

INTELLECTUAL PROPERTY VIOLATIONS

- Style can not be protected with copyright
- Possible frauds
- Damages for content creator

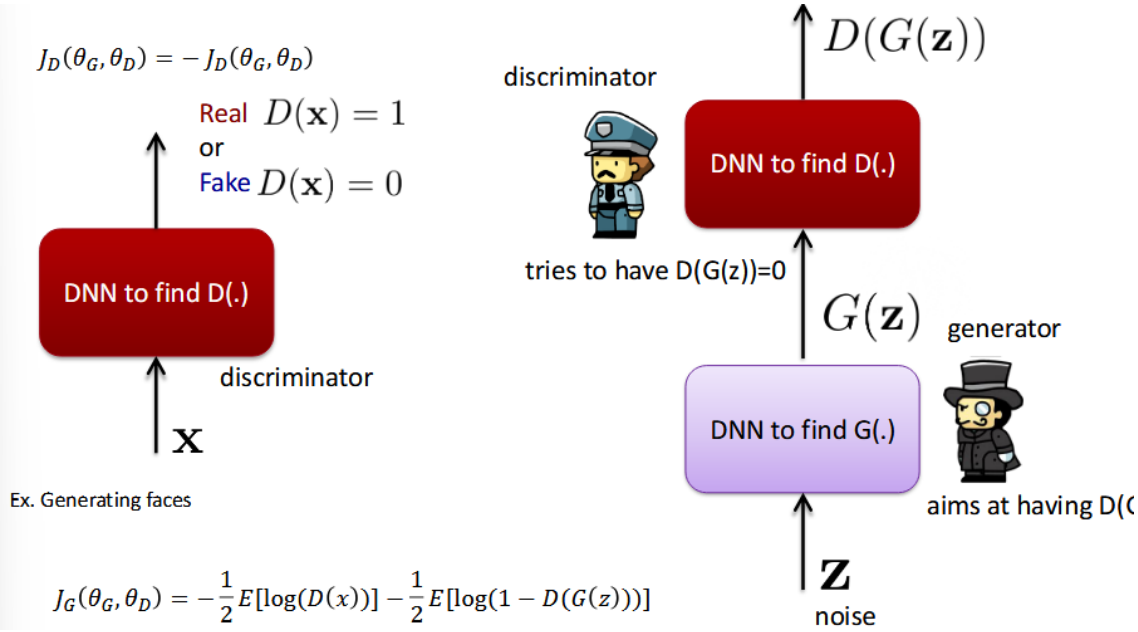
GENERATIVE AI

TAXONOMY OF GENERATIVE MODELS

- Deep generative models
    - Maximum Likelihood
        1. Explicit Density
            - Tractable density
                - Fully visible Nets
                - NADE
                - MADE
                - PixelRNN
                - Change of variable models
            - Approximate density
                - Variational (Variational Autoencoder)
                - Markov Chain (Boltzmann machine)

2. Implicit Density
    - Direct (GAN)
    - Markov chain (GSN)

GENERATIVE ADVERSARIAL NETWORKS (GANS)



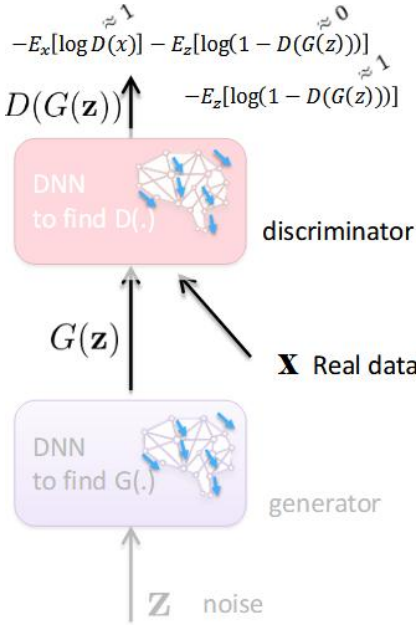
FACE CREATION USING GANS

Random noise -> Generator (Deconvolutional Network) -> Generated faces Real faces -> Discriminator (Deep Convolutional Network) -> Real or Fake

GAN TRAINING PROCEDURE

GAN training usually follows alternating procedure

1. Train discriminator D for one or more epochs (i.e., rounds on the whole dataset)
    - while training D only discriminator loss is used
    - weights of D are updated through backpropagation
    -
2. Generator trains for one or more epochs
    - sample random noise
    - generate fake samples
    - get discriminator output
    - compute loss
    - backpropagate through D and G to get gradients
    - use gradients to change generator weights
3. Repeat steps 1,2



PROBLEMS

Vanishing gradient Discriminator always guess; no possibility to learn directions for better parameters

Solutions: modified minmax, wasserstein loss

Mode collapse Generator produces only a specially-plausible output
Solutions: Wasserstein loss, unrolled GAN
Failure to converge *Solutions: adding noise, penalizing discriminator weights*

DIFFUSION
Systems that are not in thermodynamic equilibrium
Reverse the process of diffusion, i.e., diffusion process correspond to an information loss -> going back is recovering the lost information.
In case of an image, original information = clean image, diffused information = noisy image

DIFFUSION AS A CHAIN OF PROGRESSIVE STEPS (1/2)
Diffusion in images means adding noise
Noise has usually a Gaussian or normal distribution
Adding noise to pixel values alter the value of intensity
Diffusion process can be seen as a sequence of processing steps (Markov chain): every step depends only on the previous one.
Take an image and add a little of noise.
Use this to learn how to reverse this noise adding problem.

DIFFUSION AS A CHAIN OF PROGRESSIVE STEPS (2/2)
Adding a different noise, you are able to generate a different image like in GAN architecture.
This is like restoring the pixel back after diffusion.

CNN-BASED ARCHITECTURE
Reversing the process is done using a CNN (UNet)
Better performances than GAN