Digital Forensics and Biometrics

Open Exam Questions

Academic Year: 2024–2025

Course: Digital Forensics and Biometrics

Institution: *University of Padua*

Contents

Question 1 Definition of anomaly detection and the main types of anomalies

Anomaly detection is the task of identifying observations, events, or patterns in data that deviate significantly from what is considered normal or expected behavior. The central idea is that most data follows regular, repeatable patterns, while anomalies are rare and unexpected with respect to those patterns. In digital forensics and security, anomalies often correspond to malicious activities, intrusions, fraud, faults, or abnormal system behavior.

The definition of what is “normal” depends on the context and is usually learned from data through statistical models, machine learning techniques, or domain knowledge. Once a model of normal behavior is established, any observation that does not conform to it can be flagged as anomalous.

Anomalies are typically classified into three main categories.

The first category is **point anomalies**. A point anomaly occurs when a single data point is significantly different from the rest of the data. In this case, the observation is anomalous by itself, regardless of any additional context. For example, a sudden spike in network traffic, an unusually large financial transaction, or an unexpected packet loss can all be considered point anomalies. These are the simplest type of anomalies and are often addressed by classical outlier detection methods.

The second category is **contextual anomalies**. In this case, an observation is only anomalous when considered within a specific context, such as time, location, season, or operating conditions. The same value may be perfectly normal in one context and abnormal in another. For example, high electricity consumption may be normal during daytime but anomalous at night, or a large credit card expense may be expected during holiday periods but suspicious at other times. Contextual anomalies are particularly important in time-series data, where trends and seasonal components must be taken into account to distinguish true anomalies from expected fluctuations.

The third category is **collective anomalies**. A collective anomaly arises when a group or sequence of data points is anomalous as a whole, even if each individual point appears normal when considered in isolation. In this case, it is the pattern or structure of the data that is abnormal. Examples include abnormal sequences in system logs, irregular heartbeats in medical signals, or sudden changes in network behavior that persist over time. Collective anomalies are closely related to change detection, where the goal is to identify shifts in the underlying data-generating process.

Question 2 Anomaly detection evaluation techniques

Evaluating anomaly detection systems is more complex than evaluating standard classifiers, because anomalies are rare, often ambiguous, and may occur over time rather than as isolated points. For this reason, different evaluation techniques are used depending on the type of anomaly and the application.

A first family of evaluation methods is based on **binary classification metrics**. In this setting, each data point is labeled as either normal or anomalous, and the detection system produces a corresponding binary decision. Performance is evaluated using the confusion matrix, which counts true positives, false positives, true negatives, and false negatives. From these quantities, metrics such as precision, recall, accuracy, F1-score, and false positive rate are computed. Precision measures how many detected anomalies are actually true anomalies, while recall measures how many real anomalies are successfully detected. In anomaly detection, recall is often more critical than accuracy, because missing an anomaly can be far more costly than raising a false alarm, especially in forensic or security applications. Ranking-based metrics such as the area under the ROC curve are also commonly used when the detector produces anomaly scores rather than hard decisions.

Binary metrics, however, are not always sufficient, especially when dealing with time-series data and collective anomalies. In these cases, **change point detection metrics** are used to evaluate how accurately the system detects the moment when the behavior of the system changes. Rather than simply counting correct or incorrect detections, these metrics measure the temporal distance between the detected change point and the true change point. Typical measures include the average detection delay, mean absolute error, mean squared error, and root mean squared error. These metrics capture whether the system detects changes too early or too late and how large the detection error is in time. Some metrics also preserve the sign of the error to indicate systematic anticipation or delay.

A third important family of techniques is **window-based evaluation**. In many real-world scenarios, it is unrealistic to expect the detector to identify the exact time instant of an anomaly. Instead, a detection is considered correct if it occurs within a predefined time window around the true anomaly. Window-based evaluation assigns a true positive if the detector raises an alarm anywhere within this window and a false positive otherwise. This approach is particularly useful in real-time systems, where early detection should be rewarded more than late detection. Benchmarking frameworks such as the Numenta Anomaly Benchmark adopt this philosophy by assigning higher scores to early and accurate detections and penalizing false alarms.

Question 3 Main learning types in anomaly detection

Anomaly detection methods can be categorized based on the type and amount of supervision available during training.

In **supervised anomaly detection**, both normal and anomalous samples are labeled in the training data. The problem is treated as a standard binary classification task, where the model learns to discriminate between normal and anomalous behavior. Classical classifiers such as logistic regression, support vector machines, decision trees, or deep neural networks can be used. While supervised methods can achieve high accuracy when enough labeled data is available, they suffer from two major limitations: anomalies are often rare and diverse, making labeled datasets difficult to obtain, and class imbalance can strongly bias the learning process.

In **unsupervised anomaly detection**, no labeled data is available. Both training and test data may contain anomalies, but the model assumes that anomalies are rare and that normal data exhibits some underlying structure. The goal is to discover this structure and identify points that deviate from it. Unsupervised methods include clustering, density estimation, nearest-neighbor approaches, and dimensionality reduction techniques. These methods are widely used in practice because they do not require labeled anomalies, but they rely on assumptions about the distribution and concentration of normal data, which may not always hold.

Semi-supervised anomaly detection lies between these two extremes. In the most common setting, the training data contains only normal examples, while anomalies may appear at test time. The model learns a description of normal behavior and flags deviations as anomalies. Another variant involves having a small number of labeled examples from one class and a large amount of unlabeled data. Semi-supervised approaches are particularly suitable for forensic applications, where normal system behavior can be reliably recorded, but anomalous behavior is unpredictable and constantly evolving.

Question 4 Classes of classical anomaly detection algorithms

Classical anomaly detection algorithms can be grouped into several broad families based on the principles they use to model normality and detect deviations.

Statistical methods assume that normal data follows a known or learnable probability distribution. Once this distribution is estimated, anomalies are identified as observations with very low probability under the model. Examples include Gaussian models, mixture models, and time-series models such as autoregressive processes. These methods are mathematically well-founded but often struggle in high-dimensional or complex data.

Density-based methods detect anomalies by estimating how densely data points are packed in the feature space. Points that lie in regions of low density are considered anomalous. A well-known example is DBSCAN, which identifies dense clusters and treats points outside these clusters as outliers. Another example is the Local Outlier Factor, which compares the local density of a point to that of its neighbors, allowing the detection of anomalies in datasets with varying density.

Deviation-based methods focus on the influence of individual points on the overall data model. An observation is considered anomalous if removing it significantly alters statistical properties such as variance or model fit. These methods are related to the concept of influential points in statistics.

Neighbor-based methods rely on distances or similarities between data points. Anomalies are points that are far from their nearest neighbors or belong to very small or isolated clusters. These approaches are intuitive and effective in low to medium-dimensional spaces but can suffer from the curse of dimensionality.

Projection-based methods address high-dimensional data by projecting it into lower-dimensional subspaces where anomalies are easier to detect. Principal Component Analysis is a classical example, where anomalies are identified based on their reconstruction error or projection onto low-variance components. Random projection techniques can also be used to build efficient anomaly detectors.

Finally, **Bayesian network–based methods** model probabilistic dependencies among variables using graphical models. Anomalies are detected when observed data violates the expected conditional relationships encoded in the network. These methods are particularly useful when relationships between variables carry important semantic meaning.

Question 5 Support Vector Machines for anomaly detection

Support Vector Machines are margin-based learning algorithms that aim to find decision boundaries separating different classes with maximum margin. In the context of anomaly detection, SVMs can be used in both supervised and unsupervised settings.

When labeled normal and anomalous data are available, a standard two-class SVM can be trained to discriminate between the two. However, this approach is often impractical because anomalous samples are rare, heterogeneous, and difficult to label comprehensively.

For this reason, the most common use of SVMs in anomaly detection is the **One-Class SVM**. In this setting, the model is trained only on normal data and learns a decision boundary that encloses the majority of normal observations in feature space. Conceptually, the algorithm finds a region of minimum volume that contains most of the data, allowing for a small fraction of points to lie outside. At test time, any point that falls outside this learned region is considered anomalous.

Kernel functions allow the One-Class SVM to model complex, nonlinear normality regions. The choice of kernel and hyperparameters controls the tightness and shape of the boundary. One-Class SVMs are widely used for novelty detection and are particularly effective when normal behavior is well-defined and relatively stable.

Question 6 Deep Neural Networks and Deep Anomaly Detection

Deep Neural Networks are powerful function approximators composed of multiple layers of interconnected artificial neurons. Each neuron computes a weighted sum of its inputs, applies a nonlinear activation function, and propagates the result to the next layer. By stacking many layers, deep networks are able to learn highly complex and hierarchical representations of data, where lower layers typically capture simple patterns and higher layers encode increasingly abstract and semantic information.

Training a deep neural network requires defining a loss function that measures the discrepancy between the model's output and the desired objective, and then optimizing the network parameters using gradient-based methods such as stochastic gradient descent. The use of nonlinear activation functions and appropriate loss functions, such as cross-entropy for classification tasks, allows deep networks to efficiently learn even in very high-dimensional spaces. This makes them particularly effective when dealing with data types that are difficult to model using handcrafted features, such as images, videos, graphs, logs, or multivariate time series.

In the context of anomaly detection, deep learning is especially attractive because real-world forensic data is often high-dimensional, heterogeneous, noisy, and structured in complex spatial or temporal ways. Traditional anomaly detection methods rely on manually designed features and simple statistical assumptions, which often fail to capture the full complexity of such data. Deep neural networks, instead, can automatically learn informative representations directly from raw inputs and exploit spatial correlations, temporal dependencies, and nonlinear relationships that are characteristic of normal behavior.

Deep anomaly detection approaches can be broadly organized into three main paradigms.

The first paradigm uses deep neural networks as **feature extractors**. In this setting, the network is trained to map raw data into a latent representation that captures its most relevant characteristics. Anomaly detection is then performed separately using classical techniques such as one-class SVMs, distance-based methods, or density estimators applied to these learned features. This approach combines the expressive power of deep learning with the simplicity and interpretability of traditional anomaly detectors. However, since feature learning and anomaly scoring are decoupled, some information that could be useful for detection may be lost, and the learned features are not explicitly optimized for anomaly detection.

The second and most common paradigm focuses on **learning a representation of normality**. Here, the deep model is trained exclusively on nominal data in order to capture its regular patterns. The underlying assumption is that normal instances can be modeled accurately, while anomalous instances cannot. Autoencoders are a typical example: they are trained to reconstruct normal inputs through a compressed latent representation, and the reconstruction error is used as an anomaly score. Similar ideas are applied using generative models, predictability-based approaches for time series, and self-supervised learning techniques. In all these cases, anomalies are identified indirectly, through high reconstruction error, poor prediction accuracy, or low likelihood. This paradigm is widely used because it naturally fits the realistic scenario where only normal data is available during training.

The third paradigm consists of **end-to-end deep anomaly detection models**. In this case, the network is directly optimized to produce an anomaly score as output. Feature learning and anomaly scoring are jointly learned within a single model, and the loss function is explicitly designed so that normal data receives low anomaly scores while deviations receive high scores. These approaches aim to avoid the limitations of surrogate objectives, such as reconstruction or prediction error, and instead align the learning process directly with the anomaly detection task. While potentially more powerful, end-to-end methods are more complex to design and train and often require careful loss engineering or weak supervision signals.

Question 7 Autoencoders for anomaly detection

Autoencoders are one of the most widely used deep learning architectures for anomaly detection. An autoencoder is a neural network composed of two parts: an encoder, which maps the input data into a compact latent representation, and a decoder, which reconstructs the original input from this representation. The network is trained to minimize the reconstruction error between the input and its reconstruction.

In anomaly detection, autoencoders are typically trained using a **clean data setup**, meaning that the training dataset contains only nominal samples. During training, the autoencoder learns to efficiently compress and reconstruct normal patterns. At monitoring time, the trained autoencoder is applied to new data, and the reconstruction error is used as an anomaly score. The key assumption is that normal instances, being similar to those seen during training, will be reconstructed accurately, whereas anomalous instances will lead to larger reconstruction errors because they do not conform to the learned representation.

Many variants of autoencoders have been developed to address different data characteristics. Sparse autoencoders enforce sparsity in the latent representation, denoising autoencoders learn robustness to small perturbations, convolutional autoencoders exploit spatial correlations in images and videos, and LSTM-based autoencoders model temporal dependencies in sequential data. Variational and contractive autoencoders introduce regularization to improve generalization and avoid learning noisy or trivial representations. Graph autoencoders extend this idea to relational data.

Autoencoders are easy to implement and highly flexible, but they also have limitations. Since the objective function is designed for reconstruction rather than anomaly detection, the learned representation can be biased by rare but regular patterns or by outliers present in the training data. Moreover, setting a proper threshold on the reconstruction error is often non-trivial and application-dependent.

Question 8 Generative models and GAN-based anomaly detection

Generative models aim to learn the underlying data distribution of normal samples, allowing the generation of new instances that resemble the training data. Among these models, Generative Adversarial Networks, or GANs, have received significant attention in anomaly detection.

A GAN consists of two neural networks trained in opposition: a generator, which tries to produce realistic data samples from random noise, and a discriminator, which tries to distinguish between real data and generated samples. Through this adversarial process, the generator learns to approximate the distribution of the training data.

For anomaly detection, GANs are typically trained only on nominal data. At test time, the idea is to evaluate how well a given input fits within the learned normal data manifold. Since GANs do not directly provide likelihoods, anomaly scores are often computed by searching for a latent vector whose generated sample best matches the input. The reconstruction error, combined with the discriminator's response or intermediate feature differences, is then used as an anomaly score.

Several refinements of this idea exist, including architectures that introduce an explicit encoder to map inputs directly to latent space, or end-to-end designs that avoid costly optimization at test time. GAN-based approaches can model very complex data distributions and generate realistic samples, but they are notoriously difficult to train. Issues such as mode collapse, instability, and the possibility of generating samples outside the true normal manifold make anomaly detection based on GANs challenging and sometimes unreliable.

Question 9 Predictability-based anomaly detection

Predictability-based anomaly detection exploits the temporal structure of sequential data. The core idea is that normal behavior follows consistent temporal dependencies, which can be learned by a predictive model. Anomalies, instead, tend to violate these learned dependencies and therefore lead to poor predictions.

In this approach, a neural network is trained to predict the next observation in a sequence based on a temporal window of past observations. During training, only normal sequences are used, allowing the model to capture typical dynamics. At test time, the difference between the predicted instance and the actual observation is measured, and this prediction error is used as an anomaly score.

This paradigm is particularly effective for time series and video data, where motion patterns or temporal correlations are difficult to model manually. Models such as recurrent neural networks, LSTMs, convolutional architectures, or hybrid autoencoder-predictor networks are commonly used.

Predictability-based methods can capture both spatial and temporal regularities, but they are limited to sequential data and can be computationally expensive. Moreover, since they rely on surrogate objectives such as prediction accuracy, they may still be suboptimal for pure anomaly detection.

Question 10 Self-supervised anomaly detection

Self-supervised anomaly detection relies on the idea of creating artificial supervisory signals directly from the data, without requiring external labels. The model is trained to solve a surrogate classification task that forces it to learn meaningful features of normal data.

Typically, this is achieved by applying known transformations to the data, such as rotations, permutations, or other structured modifications, and training a classifier to recognize these transformations. The assumption is that normal instances will be consistent with the learned classification rules, while anomalies will produce uncertain or inconsistent predictions.

Anomaly scores can be derived in several ways, including prediction error, entropy of the output distribution, disagreement among multiple classifiers, or properties of the gradients. Self-supervised methods are flexible and can work in unsupervised or semi-supervised settings, but the choice of transformations is highly data-dependent. What works well for images may not be applicable to audio, text, or network traffic, and feature learning and anomaly scoring are still often separated.

Question 11 End-to-end anomaly score learning

End-to-end anomaly score learning aims to overcome the limitations of decoupled feature learning and scoring by jointly optimizing both. In this setting, the neural network is trained directly to produce an anomaly score for each input, and the loss function is designed to enforce a meaningful ranking between normal and anomalous instances.

These models often resemble one-class classifiers but do not inherit the weaknesses of classical distance-based or density-based metrics. Instead, they learn representations that are explicitly optimized for anomaly discrimination. Ranking-based losses, prior-driven models, and likelihood-based approaches fall into this category.

End-to-end methods are appealing because they align the learning objective directly with the detection task. However, they are more complex to design and may require assumptions about anomaly prevalence or weak supervision signals to guide training.

Question 12 Explainable anomaly detection and practical reliability

Explainability is a fundamental requirement for anomaly detection systems, especially in digital forensics and security. In these domains, detecting an anomaly is rarely sufficient on its own: analysts must understand *why* a certain observation has been flagged in order to assess its severity, verify whether it corresponds to malicious behavior, and decide on appropriate countermeasures. Without explanations, anomaly detection systems risk becoming opaque black boxes, reducing trust and limiting their practical usability.

Interpretability can be divided into **global** and **local** explanations. Global interpretability aims to describe the overall behavior of the model, identifying general patterns or rules that distinguish normal from anomalous behavior across the dataset. Local interpretability, instead, focuses on individual decisions, explaining why a specific instance has been assigned a high anomaly score. In forensic investigations, local explanations are particularly important because decisions often concern individual users, events, or transactions.

One class of explanation techniques is based on **feature importance**. These methods aim to identify which features contribute most to the anomaly score. Feature importance can be computed by observing how the anomaly score or model performance changes when features are removed, perturbed, or masked. This allows analysts to understand which variables drive anomalous behavior and to relate them to domain knowledge.

Another important class of explanations is **counterfactual explanations**. Counterfactuals answer the question: *"What is the smallest change to this instance that would make it no longer anomalous?"* By generating modified versions of an anomalous instance that fall back into the normal region, counterfactual explanations provide intuitive and actionable insights. They are useful both for analysts, who can identify the root causes of anomalies, and for system designers, who can refine detection rules or thresholds.

A widely adopted and theoretically grounded approach to explainability is **SHAP (SHapley Additive exPlanations)**. SHAP is based on cooperative game theory and attributes the anomaly score to individual features by computing their Shapley values. Each feature is treated as a "player" in a game, and the total anomaly score is distributed among features according to their marginal contributions across all possible feature subsets. SHAP has several important properties: **efficiency**, meaning that the sum of all feature contributions equals the total anomaly score; **symmetry**, ensuring that features contributing equally receive the same importance; **dummy**, meaning that irrelevant features receive zero contribution; and **additivity**, which allows explanations to be combined across models.

Because computing exact Shapley values is computationally expensive, practical implementations rely on approximations. **Kernel SHAP** is a model-agnostic method that approximates Shapley values through weighted linear regression over randomly sampled feature subsets. **Tree SHAP**, instead, is specifically designed for tree-based models and exploits their structure to compute exact or near-exact Shapley values efficiently. Both methods allow SHAP to be applied in real-world anomaly detection systems.

Explainability can also be applied to classical anomaly detection models. For example, in **Local Outlier Factor (LOF)**, explanations focus on understanding why a data point lies in a region of significantly lower local density compared to its neighbors. In **Isolation Forests**, feature importance can be derived from how often and how early specific features are used to isolate an instance in the trees: anomalous points tend to be isolated with fewer splits, and features used in early splits are considered more influential.

Beyond explanation techniques, practical strategies are essential to improve the reliability of anomaly detection systems. Visualization tools help analysts explore data distributions and clusters, human-in-the-loop approaches incorporate expert feedback into the learning process, careful threshold selection reduces false positives, and ensemble models increase robustness. Together, explainability and practical reliability mechanisms bridge the gap between algorithmic detection and real-world forensic analysis.

Question 13 Adversarial machine learning in anomaly detection

In real-world security and forensic scenarios, anomaly detection systems often operate in adversarial environments, where attackers actively adapt their behavior to evade detection. **Adversarial machine learning** studies how learning systems can be deliberately manipulated through carefully crafted inputs and how such systems can be made more robust against these attacks.

Adversarial settings are commonly characterized by the attacker's level of knowledge and intent. In **white-box scenarios**, the attacker has full access to the model architecture and parameters, while in **black-box scenarios**, the attacker can only observe the model's outputs. Attacks can also be **targeted**, aiming to force the model to produce a specific incorrect outcome, or **untargeted**, aiming to cause any misclassification or detection failure.

Several adversarial strategies are relevant to anomaly detection. **Poisoning attacks** target the training phase by injecting malicious samples or altering labels, thereby corrupting the learned model. **Evasion attacks**, which are the most common in practice, occur at test time and involve modifying inputs so that anomalous behavior is no longer detected. **Model extraction attacks** aim to reconstruct the detector or its decision boundaries by probing the system with carefully chosen queries.

Among evasion strategies, gradient-based methods play a central role. The **Fast Gradient Sign Method (FGSM)** is one of the simplest and most widely studied adversarial attacks. FGSM exploits the gradient of the loss function with respect to the input and perturbs the input in the direction that maximally increases the loss. Formally, the adversarial example is obtained by adding a small perturbation proportional to the sign of the gradient. The magnitude of the perturbation is controlled by a parameter that balances attack effectiveness and perceptual similarity to the original input. Despite its simplicity, FGSM is highly effective and computationally efficient, making it particularly dangerous in real-time systems. Variants of FGSM can be adapted for targeted or iterative attacks by adjusting the direction and magnitude of the perturbation.

More advanced adversarial strategies build upon this idea, using iterative optimization, saliency maps, or decision-boundary approximations to generate stronger attacks. These methods can produce minimal yet highly effective perturbations that are difficult to detect visually or statistically.

Defending against adversarial attacks in anomaly detection is challenging. **Adversarial training**, where models are trained using both normal and adversarially perturbed samples, can improve robustness but increases computational cost and may reduce generalization. **Ensemble methods** combine multiple detectors to reduce vulnerability to specific attack strategies. **Generative defenses** attempt to project inputs back onto the normal data manifold, filtering out adversarial perturbations. However, no defense is universally effective, and robustness often comes at the cost of increased complexity or reduced sensitivity.

Ultimately, adversarial anomaly detection can be seen as a non-cooperative game between attackers and defenders, where both sides continuously adapt their strategies. Designing robust anomaly detection systems therefore requires not only accurate detection models but also a deep understanding of adversarial behavior and defensive trade-offs.

Question 14 Biometrics: fundamentals, taxonomy, and operational modes

Biometric systems are authentication and identification systems that rely on intrinsic human characteristics to recognize or verify an individual's identity. Unlike traditional authentication methods based on something a user knows, such as a password, or something a user has, such as a card or token, biometrics are based on *who the user is*. For this reason, biometrics play a central role in modern security systems and digital forensics, where reliable identity attribution is crucial.

Biometric traits can be broadly classified into **physiological** and **behavioral** characteristics. Physiological traits are related to the physical structure of the human body and include fingerprints, facial features, iris and retina patterns, hand geometry, and DNA. Behavioral traits, instead, are linked to how a person acts or performs certain actions, such as voice patterns, signature dynamics, keystroke dynamics, or gait. In addition to classical biometrics, **soft biometrics** are often used to support identification. These include physical attributes such as height, gender, or hair color, behavioral tendencies, and adhered characteristics such as clothing or accessories. Soft biometrics are generally not distinctive enough to uniquely identify an individual, but they are useful for narrowing down the search space or supporting forensic investigations.

Soft biometrics are human characteristics that are not sufficiently distinctive to uniquely identify an individual on their own, but that provide useful contextual information. They include physical attributes such as height, gender, or hair color, behavioral traits like gait or posture, and adhered characteristics such as clothing or accessories. Soft biometrics are typically used to narrow down candidate sets, support surveillance and forensic analysis, and complement traditional biometric systems rather than replace them.

The **motivation for adopting biometric systems** lies in their ability to provide convenient and reliable authentication based on who a person is, rather than on credentials that can be forgotten, lost, or stolen.

From an operational perspective, biometric systems can function in two main modes: **verification** and **identification**. In identity verification, also known as 1:1 matching, a person claims an identity and the system verifies this claim by comparing the newly acquired biometric sample with the corresponding template stored in the database. Identification, instead, is a 1:N process in which the system compares the biometric sample against a database of many enrolled individuals to determine whether a match exists and, if so, whose identity it is. Identification is computationally more demanding and raises additional privacy and scalability concerns.

Because no single authentication method is perfectly secure, biometric systems are often deployed as part of **multi-factor authentication** schemes, where biometric traits are combined with knowledge-based or possession-based factors. This combination increases security by mitigating the weaknesses of individual authentication strategies.

Question 15 Security, attacks, and performance of biometric systems

Despite their advantages, biometric systems are vulnerable to a variety of security threats and attacks. One common class of attacks is **masquerade attacks**, where an attacker attempts to impersonate another user in order to gain unauthorized access or attribute actions to someone else. Closely related are **identity theft** attacks, in which an attacker enrolls in the system under a false identity, and **multiple identity** attacks, where the same individual gains access multiple times under different identities.

Traditional **trial-and-error attacks**, such as password guessing or offline dictionary attacks, are less applicable to biometrics but still relevant when biometric systems are combined with other authentication factors. Biometric-specific attacks include **replication attacks**, where an attacker creates a physical or digital copy of a biometric trait, and **digital spoofing or playback attacks**, where biometric data is intercepted during transmission and replayed to the system. In some scenarios, simply preventing correct identification can already cause significant disruption.

Various defense mechanisms are employed to mitigate these threats. Limiting the number of authentication attempts reduces the effectiveness of trial-and-error attacks, while increasing the sensitivity of verification thresholds can reduce false acceptances at the cost of usability. To counter replication attacks, biometric systems often incorporate **liveness detection**, which exploits characteristics that cannot be reproduced by static copies, such as temperature, blood flow, or three-dimensional structure. Digital spoofing attacks are addressed through secure communication channels using encryption and digital signatures.

The performance of biometric systems is evaluated by comparing similarity scores between biometric samples and applying a decision threshold. Because biometric measurements are inherently variable, even samples from the same individual acquired at different times rarely coincide exactly. Two key performance metrics are the **False Rejection Rate (FRR)**, which measures how often legitimate users are incorrectly rejected, and the **False Acceptance Rate (FAR)**, also known as the False Match Rate, which measures how often impostors are incorrectly accepted. The trade-off between these two errors is controlled by the decision threshold, and the point at which FAR equals FRR is known as the **Equal Error Rate (EER)**.

Biometric reliability is affected by several sources of variability and error. Aging, environmental conditions, sensor quality, and physical changes such as injuries can degrade biometric templates over time. For example, fingerprints may deteriorate due to occupation or trauma, facial recognition is sensitive to lighting and pose, and voice recognition is affected by illness or recording quality. These issues are typically mitigated through periodic re-enrollment, enrollment of multiple samples or traits, and controlled acquisition environments.

Question 16 Fingerprints as biometric traits: acquisition, processing, and recognition

Fingerprints are among the most widely used biometric traits due to their uniqueness, permanence, and long history of use in forensic investigations. They are employed in a wide range of applications, including criminal identification, border control, victim identification, and access control systems. Despite popular misconceptions fueled by media portrayals, fingerprint matching is a complex process that involves uncertainty, partial data, and potential human bias.

Fingerprint acquisition can be performed using various technologies. Optical sensors capture reflected light from the finger surface, capacitive sensors exploit differences in electrical conductivity between ridges and valleys, thermal sensors measure temperature differences, and radio-frequency or ultrasonic sensors probe subsurface structures of the skin. Each technology has its own advantages and limitations in terms of robustness, cost, and susceptibility to noise or spoofing.

Digitally, fingerprints are modeled as **two-dimensional grayscale images**, where intensity values represent ridge and valley structures. Fingerprint matching can be approached using **image-based** or **feature-based** methods. Image-based methods rely on global correlation between fingerprint images and are conceptually simple, but they are highly sensitive to rotation, translation, noise, and partial prints. For this reason, modern systems predominantly rely on feature-based approaches.

Feature-based fingerprint recognition relies on a **hierarchical representation** of fingerprint features. **Level 1 features** capture global ridge flow characteristics, such as ridge orientation and frequency, and include singular points like loops and deltas. Ridge orientation is typically estimated using gradient-based methods or Fourier-domain analysis, modeling local ridge patterns as sinusoidal signals. Singularities are detected by analyzing changes in the orientation field, often using the Poincaré index, and are used for fingerprint alignment and classification.

Level 2 features consist of **minutiae**, which are local ridge characteristics such as ridge endings and bifurcations. Minutiae-based recognition is the cornerstone of automated fingerprint identification systems because minutiae are highly distinctive and relatively stable. Extracting minutiae requires skeletonizing the ridge structure and typically demands higher-resolution images, around 500 dpi.

Level 3 features, such as pores and fine ridge details, provide additional discriminative power but require very high-resolution sensors and are less commonly used in large-scale systems.

Question 17 Minutiae extraction and representation in fingerprint recognition

Minutiae extraction is a fundamental step in fingerprint recognition, as minutiae constitute the primary features used by automated fingerprint identification systems. The process starts from a raw fingerprint image, which often contains noise, uneven illumination, and distortions due to skin conditions or sensor imperfections. To address these issues, fingerprint images are first enhanced to improve ridge–valley contrast and continuity. This enhancement phase typically relies on oriented filters, such as Gabor filters, that are tuned to the local ridge orientation and frequency, reinforcing true ridge structures while suppressing noise.

Once enhanced, the image is binarized, converting grayscale values into a binary ridge–valley representation. The binary image is then thinned through skeletonization, reducing ridge structures to one-pixel-wide lines while preserving their topology. Skeletonization is essential because it simplifies the detection of minutiae and ensures consistent geometric representation.

Minutiae detection is usually performed by analyzing the local pixel neighborhood in the skeletonized image. Ridge endings and bifurcations are identified by counting the number of ridge-connected neighbors around each pixel. However, this process is highly sensitive to noise and image artifacts. As a result, many detected minutiae are spurious, arising from broken ridges, noise, scars, or imperfections introduced during binarization and thinning.

To mitigate this problem, minutiae filtering is applied using heuristic rules based on spatial proximity, ridge length, orientation consistency, and distance from image borders. Only minutiae that satisfy these constraints are retained. Each minutia is then represented by its spatial coordinates, orientation, and type, following standardized formats such as those defined by ANSI/NIST or FBI specifications. This standardized representation enables interoperability across fingerprint recognition systems.

Question 18 Fingerprint matching as a correspondence problem

Minutiae-based fingerprint matching can be formulated as a **correspondence problem** between two sets of points: one extracted from the query fingerprint and one from the template fingerprint. Each minutia is described by its position, orientation, and type, and the goal of matching is to determine which minutiae from the query correspond to which minutiae in the template, if the two fingerprints originate from the same finger.

A simple and idealized formulation of this problem is known as **algebraic matching**. In this approach, it is assumed that each minutia in the query fingerprint corresponds to exactly one minutia in the template, and that matching can be decided locally using **tolerance regions**. After alignment, each query minutia is associated with a small tolerance box or region in which a corresponding template minutia must fall, both in terms of spatial position and orientation. A further simplifying assumption is that these tolerance regions do not intersect, so that matching is unambiguous.

While algebraic matching is conceptually simple and mathematically elegant, its assumptions rarely hold in real fingerprint data. In practice, fingerprint impressions are affected by noise, partial overlap, missing minutiae, spurious minutiae introduced during extraction, and skin distortion caused by pressure and elasticity. As a result, tolerance regions may overlap, true correspondences may be missing, and false correspondences may appear valid. These factors make fingerprint matching inherently uncertain and motivate the use of more robust strategies that go beyond local, one-to-one assumptions.

Question 19 Alignment and the Hough transform in fingerprint matching

Before meaningful correspondence between minutiae can be established, the two fingerprint representations must be brought into a common reference frame. This process is known as **alignment** and typically involves estimating a rigid transformation composed of translation and rotation. Without alignment, spatial proximity and orientation comparisons between minutiae would be meaningless.

One of the most widely used techniques for robust fingerprint alignment is the **Hough transform**. Rather than assuming known correspondences, the Hough-based approach works by generating many hypotheses about the possible transformation between the query and template fingerprints. Each hypothesis is derived from pairing a query minutia with a template minutia and computing the translation and rotation that would align them.

These transformation parameters are then mapped into a discretized **transformation space**, where each hypothesis casts a vote. If the two fingerprints originate from the same finger, many true minutiae correspondences will independently generate similar transformation parameters, causing votes to accumulate in the same region of the transformation space. The peak in this accumulator space identifies the most probable global alignment, that is, the transformation that is consistent with the largest number of minutiae pairs.

The key strength of the Hough transform lies in its robustness. Spurious minutiae and incorrect pairings generate votes that are scattered across the transformation space and therefore do not reinforce each other. Missing minutiae simply reduce the number of votes but do not prevent correct alignment as long as enough consistent pairs remain. For these reasons, the Hough transform is particularly effective in fingerprint matching scenarios characterized by noise, partial prints, and extraction errors.

Question 20 Pairing, pre-alignment, and refinement strategies

Once a global alignment has been estimated, the next step in fingerprint matching is **minutiae pairing**, that is, deciding which minutiae from the query and template should be considered corresponding. Pairing is typically performed by applying spatial and angular proximity constraints: minutiae are considered a match if they fall within predefined distance and orientation tolerances after alignment. The overall similarity between two fingerprints is then computed based on the number and consistency of matched minutiae, often normalized to account for partial overlap.

To improve efficiency and accuracy, **pre-alignment strategies** are sometimes used to generate good initial alignment hypotheses before full matching. Absolute pre-alignment relies on global reference structures such as singular points, while relative pre-alignment exploits invariant relationships between minutiae pairs, such as distances and relative angles.

Some approaches propose **local matching techniques** that compare minutiae using attributes that are invariant to global transformations such as translation and rotation. As a result, they are generally simpler, computationally less demanding, and more tolerant to strong non-linear distortions or partial fingerprint overlap.

Even after alignment and initial pairing, matching can be refined using more advanced optimization strategies. **Relaxation methods** iteratively adjust the confidence of minutiae pairs based on their mutual consistency: pairs that agree with many other pairs are reinforced, while inconsistent pairs are weakened or discarded. This allows the system to converge toward a globally consistent matching configuration.

Other approaches frame fingerprint matching as a combinatorial optimization problem. **Tree search methods**, inspired by operations research, explore a search tree of possible match assignments while pruning inconsistent or low-quality branches. **Energy minimization approaches** associate a cost or energy to each possible matching configuration, penalizing geometric inconsistency and rewarding agreement, and then seek the configuration with minimum energy.

Question 21 Beyond minutiae: ridge features, indexing, and scalability

Although minutiae are the cornerstone of fingerprint recognition, relying on them alone can be insufficient in challenging scenarios. Low-quality images, partial prints, or altered fingerprints may contain too few reliable minutiae for accurate matching. To improve robustness, additional features derived from ridge structure and texture are often incorporated.

Ridge-based features capture information such as local ridge orientation, frequency, and spatial relationships, providing complementary evidence when minutiae are unreliable. Texture-based representations, such as FingerCode, encode fingerprint information using filter responses over fixed regions, allowing coarse but robust matching. These features are particularly useful for indexing and candidate selection rather than final verification.

In large-scale identification systems such as national AFIS databases, exhaustive 1:N fingerprint matching is computationally infeasible. To address scalability, fingerprint indexing techniques are used to reduce the search space before detailed matching. A common approach is based on **minutiae triplets**, where groups of three minutiae form triangles described by feature vectors encoding their geometric properties. These features are ordered, hashed together with the fingerprint ID, and stored in a hash table. Given a query fingerprint, similar triangles are generated and looked up in the table, producing candidate ID lists. A voting mechanism then selects the most likely fingerprints for final matching, enabling efficient large-scale identification.

Real-world systems must also handle difficult cases, including latent fingerprints, intentional alterations, and low-quality impressions. Extensions to palm prints and the integration of deep learning–based feature extraction are increasingly used to enhance performance in such scenarios. Overall, effective fingerprint recognition systems combine minutiae-based matching with auxiliary features, indexing strategies, and scalable architectures to operate reliably in forensic and operational environments.

Question 22 Face recognition as a biometric modality: applications, advantages, challenges

Face recognition is a biometric technology that identifies or verifies individuals based on their facial appearance. It is widely employed in access control, identity verification, video surveillance, forensic investigations, social media indexing, and human–computer interaction. Unlike contact-based biometric traits such as fingerprints, face recognition can operate at a distance using non-contact sensors, making it particularly suitable for large-scale, unobtrusive, and covert acquisition scenarios.

One of the main advantages of face recognition is the availability of large legacy databases and the ease with which facial images can be acquired using standard RGB cameras. In addition to identity, facial images may convey supplementary information such as age, gender, ethnicity, and emotional state. At the same time, face recognition is highly sensitive to external and subject-related factors, including illumination and acquisition conditions, facial expressions, wardrobe and occluding elements, and dataset limitations, all of which can significantly degrade recognition performance.

Face recognition systems can employ different sensing technologies, including standard RGB cameras, infrared sensors, and three-dimensional scanners, each offering different trade-offs in terms of robustness and acquisition constraints.

A typical face recognition system follows a multi-stage processing pipeline. The first stage is **face localization**, which aims to detect and segment the face region from the background. This is followed by **alignment and normalization**, where the detected face is geometrically transformed into canonical coordinates to reduce variability due to pose and scale. **Face processing** techniques are then applied to compensate for photometric variations such as changes in illumination and viewpoint.

Subsequently, **feature extraction** encodes discriminative facial information into a compact representation that is suitable for distinguishing between different individuals. Finally, **face comparison** is performed by comparing a probe image either against a single reference template in a verification scenario or against a gallery of enrolled identities in an identification scenario, using similarity measures and decision thresholds. In modern systems, several of these processing stages may be merged into unified or end-to-end architectures.

Question 23 Facial features representation, extraction and detection

Facial features can be described at different levels of granularity. **Level 1 features** capture coarse, global characteristics such as overall face shape, gender, or ethnicity and can be extracted even from low-resolution images. **Level 2 features** encode the geometry and relative spatial arrangement of facial components such as eyes, nose, and mouth, and are commonly exploited for face alignment and deformation compensation. **Level 3 features** correspond to fine-grained local details such as scars, moles, and skin texture, which provide high discriminative power but require high-quality images and are more sensitive to noise and occlusions.

Feature extraction methods for face recognition can be broadly categorized into **appearance-based, texture-based, and model-based approaches**. Global appearance-based techniques, such as Eigenfaces, Fisherfaces, and PCA/ICA-based representations, model faces as points in a low-dimensional subspace learned from training data. Texture-based methods rely on local descriptors, including Gabor filters and Local Binary Patterns, which capture local intensity variations and are more robust to moderate illumination changes. Model-based approaches, such as Elastic Bunch Graph Matching and Active Appearance Models, explicitly represent both facial shape and texture and allow deformation modeling across expressions and poses.

Early face recognition systems relied heavily on **explicit face detection** using handcrafted features. A classical and influential approach represents faces as combinations of simple **rectangular (box) filters**, designed to capture characteristic contrast patterns associated with facial components, such as the darker eye region relative to the cheeks or the vertical structure of the nose. These box filters can be interpreted as edge, line, or center-surround features, depending on their configuration and orientation. A filter produces a maximum response when it is well aligned with the facial structure it is intended to model.

Efficient computation of box filter responses is achieved through the use of **integral images**, which allow the sum of pixel intensities over arbitrary rectangular regions to be computed in constant time, independently of their size. This makes it feasible to evaluate a very large number of candidate features at multiple scales and image locations.

Because the number of possible box filters is extremely large, **AdaBoost** is employed to automatically select a small subset of highly discriminative features and to train weak classifiers. Each weak classifier makes a simple decision based on a single feature, and AdaBoost combines them into a stronger classifier by iteratively focusing on hard-to-classify samples.

To achieve real-time performance, especially when scanning all subregions of an image at multiple scales, weak classifiers are arranged into a **cascade of classifiers**. Instead of applying all features to every image region, the cascade tests a sequence of increasingly complex stages. Early stages use very few features to quickly reject the majority of non-face regions, while later stages apply more features only to regions that have passed previous tests. Each stage is trained to allow a small fraction of false positives, ensuring that true faces are unlikely to be discarded prematurely. This cascading strategy yields detection accuracy comparable to that of a monolithic classifier using all features, while achieving speed-ups of an order of magnitude. Successive stages are trained on true positives and on the false positives propagated by earlier stages, progressively refining the decision boundary. Empirical evaluations show that cascaded detectors achieve similar ROC performance to full classifiers but at a dramatically reduced computational cost.

Beyond detection, face recognition systems must handle substantial **variability in pose, illumination, orientation, facial expression, and occlusions**. These variations motivate the use of alignment procedures, normalization techniques, and robust feature representations. In more challenging scenarios, **heterogeneous face recognition** is required, where probe and gallery images belong to different modalities, such as visible light versus infrared images, photographs versus sketches, or 2D versus 3D data.

Several strategies have been proposed to address heterogeneous recognition. **Synthesis-based methods** attempt to generate a synthetic image in a common modality. **Feature-based methods** extract descriptors that are largely invariant across domains, such as SIFT or LBP features, often followed by discriminant feature selection techniques like Linear Discriminant Analysis. **Prototype-based similarity methods** represent a face image as a vector of similarities to a set of prototype images available in both modalities, enabling comparison across heterogeneous domains.

Question 24 Face alignment and landmark extraction

Since face images can be taken from different viewpoints, alignment aims to reduce geometric variability by mapping facial images to a canonical pose and expression. This process relies on detecting **facial landmarks**, which are key points corresponding to facial anatomy, such as eye corners, nose tip, and mouth contour. The landmark is then used to align the acquired image to a neutral expression/frontal pose. Accurate landmark detection is crucial for normalization, expression compensation, and emotion recognition.

Several approaches have been proposed for landmark extraction, including regression-based methods using ensembles of regression trees, Active Appearance Models, Local Binary Features, and deep learning–based cascaded convolutional networks. These methods progressively refine landmark positions and can operate in real time under unconstrained conditions.

Question 25 Deep face recognition and feature learning

Modern face recognition systems are predominantly based on deep learning. Convolutional neural networks are trained on large-scale datasets to learn compact and highly discriminative facial embeddings. These embeddings generalize across identities and enable **zero-shot recognition**, where identities unseen during training can still be recognized.

Different backbone architectures and loss functions are used to improve discrimination, including Euclidean-based losses, angular margin losses, and modified softmax formulations. Deep features are compared using similarity measures, and thresholds are applied to make verification or identification decisions.

Question 26 Attacks, bias, and fairness in face recognition

Despite their high recognition accuracy, face recognition systems remain vulnerable to a variety of security threats and raise important ethical concerns. One major class of threats is represented by **presentation attacks**, in which an attacker attempts to impersonate a legitimate user by presenting an artificial or manipulated facial representation to the sensor. Simple photo attacks rely on printed images or digital photographs displayed on a screen and exploit the lack of liveness awareness in the system. More sophisticated video replay attacks use recorded videos of a target person, enabling the simulation of facial motion or blinking and making detection more challenging. Mask attacks exploit three-dimensional facial replicas that reproduce both geometry and texture, posing a significant risk for systems that rely on shape information. Makeup-based attacks alter facial appearance through cosmetics to either resemble another individual or evade recognition.

Beyond security, face recognition systems also raise significant **fairness and bias concerns**. Empirical evidence shows that recognition performance can vary substantially across demographic groups defined by attributes such as gender, age, or ethnicity, leading to unequal error rates and potentially discriminatory outcomes. These disparities often stem from imbalanced or unrepresentative training datasets, differences in acquisition conditions, or algorithmic design choices that unintentionally encode demographic information.

Several notions of fairness have been proposed to address bias in face recognition systems. **Group fairness** requires that individuals from protected and unprotected demographic groups have the same probability of being correctly classified, typically by equalizing error rates such as false acceptances or false rejections across groups. A complementary individual-centered notion is **fairness through awareness**, which states that individuals who are similar according to a task-specific similarity metric should receive similar outcomes, independently of sensitive attributes. By contrast, **fairness through unawareness** requires that sensitive attributes are not explicitly used in the decision process. However, this approach is often insufficient, as demographic information may still be implicitly encoded in facial features and learned representations. A more principled definition is **counterfactual fairness**, which requires that the system's decision for an individual remains unchanged in a hypothetical scenario where the individual belonged to a different demographic group, while all other characteristics are held constant. Finally, **fairness in relational domains** extends these notions by considering social or organizational relationships in addition to individual attributes, acknowledging that decisions may be influenced by relational structures.

Mitigating bias in face recognition can be achieved at different stages of the system pipeline. **Pre-processing strategies** attempt to balance datasets or modify input representations to reduce demographic correlations. **In-processing approaches** introduce fairness-aware constraints or regularization terms directly into the learning objective, encouraging the model to disentangle identity from sensitive attributes. **Post-processing methods** adjust similarity scores or decision thresholds after training to equalize error rates across groups.

Question 27 Iris recognition and analysis as a biometric trait

Iris recognition exploits the highly complex texture of the iris, which arises from the random morphogenesis of multiple anatomical layers during prenatal development. The iris consists of several layers, including the posterior epithelium, stromal layer, muscle layer, and anterior border layer. The resulting texture contains features such as concentric furrows, radial furrows, crypts, and zig-zag patterns.

The uniqueness of iris texture is established early in life and remains largely stable, even among monozygotic twins. Iris color, determined by pigmentation, is not crucial for recognition, as texture structure is the primary discriminative element.

An iris recognition system typically consists of several stages, starting with **image acquisition**, usually performed with near-infrared cameras and illuminators, it typically requires the cooperation of the subject. Accurate **segmentation** is critical and involves isolating the iris from surrounding structures such as the pupil, sclera, eyelids, eyelashes, and specular reflections.

Classical segmentation techniques rely on the **integro-differential operator**, which searches for circular or elliptical boundaries by maximizing intensity gradients along candidate contours. After segmentation, the iris is **normalized** using a rubber-sheet model that maps the annular iris region from cartesian coordinates to a fixed-size pseudo-polar representation, compensating for pupil dilation and size variations.

Once normalized, iris texture is encoded into a compact representation known as the **iris code**. Classical approaches use **Gabor filters** applied over a fixed grid to extract local phase information from the iris texture. The phase response is quantized into a binary representation, while magnitude information is discarded. A corresponding binary mask is generated to indicate occluded (by eyelids, eyelashes, ecc.) or unreliable regions (reflections, ecc.).

Commercial iris recognition systems typically produce iris codes of fixed length, enabling efficient storage and comparison.

Iris matching is performed by comparing two iris codes using the **fractional Hamming distance**, which measures the proportion of disagreeing bits between the codes, taking into account the corresponding masks. Because head rotation can cause circular shifts in the normalized iris representation, matching is performed over multiple shifts, and the minimum distance is retained.

Iris recognition is widely regarded as one of the most accurate and reliable biometric modalities, with performance often reported to be significantly higher than that of fingerprint recognition. Unlike fingerprints, which are constantly exposed to the environment and therefore susceptible to wear, cuts, and damage, the iris is naturally protected by the cornea, the transparent outer layer of the eye. This protection helps preserve the fine texture of the iris over time, contributing to its high stability. Empirical evidence suggests that iris patterns remain reliably unchanged for decades, although not necessarily for an entire lifetime. Moreover, iris acquisition can be performed in a safe and hygienic manner at a distance from the eye, without physical contact, making iris recognition well suited for high-security and large-scale applications.

Despite these advantages, iris recognition is affected by several **impairing factors** that can degrade performance. One important source of variability is **pupil dilation**, which causes non-linear deformation of the iris texture. Although normalization procedures compensate for dilation to some extent, large differences in pupil size can shift the distribution of genuine match scores and increase error rates. Over long time intervals, iris recognition systems may also experience **template aging**, due to gradual physiological changes such as pupil size reduction or variations in corneal shape.

A particularly relevant security concern in iris recognition is the use of **contact lenses**. Clear lenses slightly increase the false non-match rate and mainly pose a minor social impact issue. In contrast, **textured or cosmetic lenses represent a serious threat**, as they introduce artificial patterns that can alter or obscure the natural iris texture. Detection methods typically analyze image artifacts, for instance in the **Fourier spectrum**, where periodic structures introduced by textured lenses become evident.

Beyond contact lenses, additional factors may further impair iris recognition performance. These include medical conditions such as acute iris inflammation, poor subject presentation (e.g., closed eyelids or rotated irises), unconstrained acquisition scenarios such as recognition in crowds, capture environment issues like blur or inadequate illumination, image processing or storage artifacts such as compression or corruption, and unusual anatomical characteristics of the eye. As a consequence of these combined factors, real-world iris recognition systems may exhibit significantly higher missing rates compared to controlled acquisition conditions.

Question 28 Gait recognition and analysis as a biometric trait

Gait analysis is the systematic study of human walking patterns and has applications ranging from medical diagnostics and rehabilitation to security and forensic identification. In the biometric context, **gait recognition** aims to identify or verify individuals based on the way they walk, exploiting the fact that human locomotion exhibits person-specific patterns that can be observed at a distance and without subject cooperation. This makes gait particularly attractive in surveillance and forensic scenarios where other biometric traits may not be available.

Human walking follows a well-defined **gait cycle**, composed of two main phases: **stance** and **swing**. The stance phase includes initial contact, loading response, mid-stance, terminal stance, and pre-swing, while the swing phase is divided into initial, mid-, and terminal swing. These phases provide a temporal structure that can be exploited to extract periodic and discriminative gait features.

Gait acquisition can be performed using different types of sensors. **Wearable systems** rely on inertial measurement units, force sensors, or electromyography to capture acceleration, orientation, and muscle activity. **Non-wearable systems** may use pressure-sensitive floors to record footprints and stance dynamics. **Vision-based systems**, which are the most common in biometric applications, use RGB cameras, multi-camera setups, depth sensors, or 3D technologies such as LiDAR or RF sensors to capture walking sequences without physical contact.

A typical **gait recognition pipeline** consists of several stages. First, image or video data are acquired. Then, the walking subject is segmented from the background, often through **silhouette extraction**, which is conceptually similar to segmentation in face or fingerprint recognition. Background subtraction techniques range from traditional statistical models to deep learning–based approaches. Challenges at this stage include shadows, moving backgrounds, threshold selection, and compression artifacts, although depth and 3D data can significantly improve robustness.

After segmentation, gait information is represented using either **model-free** or **model-based** approaches. Model-free methods operate directly on silhouettes and motion patterns, without explicitly estimating body structure. Common representations include Motion Energy Images (MEI), Motion History Images (MHI), Motion Silhouette Images (MSI), and especially **Gait Energy Images (GEI)**, which average silhouettes over a gait cycle to capture both shape and motion information. Extensions such as Gait History Images or Active Energy Images aim to encode temporal dynamics more explicitly and handle variations such as carrying objects or clothing changes.

Model-based approaches, on the other hand, explicitly represent the human body as a set of joints and segments. These methods fit a 2D or 3D skeleton to the silhouette or directly estimate joint positions using regressors or deep learning models. From the resulting skeleton, time series of joint coordinates, angles, stride length, and cadence can be extracted. Model-based representations are generally more interpretable and can better handle viewpoint changes, but they are more sensitive to segmentation errors and require higher-quality data.

Once gait signals are extracted, **step period detection** and temporal normalization are often performed to isolate individual walking cycles. This involves signal preprocessing, template matching to identify steps, normalization to a common reference system, and interpolation to handle variations in walking speed. Additional filtering and adaptive template updates can further improve detection robustness.

Feature extraction then converts raw time series or images into compact and discriminative representations. This may involve hand-crafted features, dimensionality reduction techniques such as PCA or LDA, or learned representations obtained through CNNs, RNNs, or self-organizing maps. Finally, **classification or identification** is performed using traditional machine learning methods such as SVMs, Dynamic Time Warping, or Hidden Markov Models, as well as deep learning–based classifiers. Systems can operate in **closed-set** or **open-set** scenarios, and gait can also be used for **continuous authentication**, for example through One-Class SVMs.

Despite its advantages, gait recognition faces several challenges. Gait is influenced by many external and internal factors, including footwear, fatigue, carried objects, walking speed, injuries, medical conditions, and psychological state. Environmental conditions such as illumination, viewpoint, and occlusions further affect performance, particularly in outdoor settings. While multi-camera systems, infrared sensors, and geometric normalization can mitigate some of these issues, gait patterns may still change over time, requiring **re-enrolment** to maintain accuracy.

Question 29 Voice recognition and analysis as a biometric trait

Voice recognition is a biometric modality that identifies or verifies individuals based on characteristics of their speech signal. It is particularly relevant in scenarios where interactions occur using voice only, such as telephone banking, remote authentication, contractual agreements, surveillance, and forensic investigations. In addition to identity, voice signals may convey information about the speaker's emotional state, stress level, or health condition, which further increases their forensic relevance.

Voice-based biometric systems are commonly used for **voice authentication**, **speaker detection**, and **forensic speaker recognition**. Two main operational modes can be distinguished. **Text-dependent systems** require users to utter a specific passphrase, often randomized, and are typically employed in access control applications. **Text-independent systems**, which do not constrain the spoken content, are more challenging but essential in surveillance and forensic contexts.

Speech production is an extremely complex process influenced by both **physiological** and **sociolinguistic** factors. Physiological factors include the length, shape, and tissue properties of the vocal tract, as well as the configuration of articulatory organs. Sociolinguistic factors reflect education level, dialect, linguistic environment, and social context. As a result, speaker recognition relies on both **high-level linguistic features**, such as idiolect and phonotactic patterns, and **low-level acoustic features**. Acoustic features include **prosodic features**, such as energy, intonation, speech rate, and rhythm, as well as **spectral features**, which are closely related to individual articulatory behavior and vocal tract configuration.

From a signal processing perspective, the vocal tract can be modeled as a set of filters acting on a glottal excitation signal. This motivates the widespread use of **filterbank-based approaches**, which analyze speech across multiple frequency bands. Among these, **Mel Frequency Cepstral Coefficients (MFCCs)** are the most widely used features in speaker recognition. MFCC extraction involves segmenting the signal into overlapping frames, computing the power spectrum, filtering it using a Mel-scaled filterbank, applying a logarithmic transformation, and finally performing a Discrete Cosine Transform to obtain a compact representation. To improve robustness, MFCCs are often augmented with first- and second-order temporal derivatives, resulting in feature vectors of typically 39 coefficients.

Once features are extracted, **speaker modeling** aims to estimate the class-conditional distribution of these features. This can be achieved using **parametric approaches**, such as Gaussian Mixture Models, or **non-parametric and discriminative methods**, including vector quantization, Support Vector Machines, and deep neural networks. System performance depends on feature robustness, modeling accuracy, channel variability, and the amount of available training data, making forensic speaker recognition a particularly challenging task.

Question 30 AI forensics: generative models, threats, and forensic challenges

AI forensics aims to develop techniques to **detect, attribute, and analyze manipulated or synthetic content**, as well as to assess the reliability of digital evidence. This includes detecting deepfakes, identifying traces left by generative models, analyzing biases in AI systems, and understanding how dataset composition and cognitive bias influence model behavior. Issues such as gender bias, lack of diversity in training data, and missing data provenance further complicate forensic analysis.

The rapid development of machine learning and artificial intelligence has introduced powerful tools for multimedia analysis, but also new threats that significantly impact digital forensics. In recent years, **generative AI** has enabled the creation of highly realistic synthetic content, including text, images, audio, video, and 3D models. While these technologies have legitimate applications, they also pose serious risks to cybersecurity, privacy, and trust in digital evidence.

AI-based threats include large-scale **social engineering**, automated **content generation**, **robocalling**, and attacks against biometric systems such as voice and face recognition. In particular, **impersonation attacks** have become increasingly sophisticated, with models capable of mimicking an individual's writing style, voice, or facial appearance. Audio and video **deepfakes** can be used to access bank accounts, manipulate political campaigns, commit identity theft, or gain access to biometric-protected premises.

Modern generative models can be classified according to their underlying probabilistic assumptions. **Generative Adversarial Networks (GANs)** consist of a generator and a discriminator trained in opposition: the generator aims to produce realistic samples, while the discriminator learns to distinguish real from fake data. GANs have been widely used for realistic face synthesis but suffer from training instabilities such as vanishing gradients, mode collapse, and lack of convergence. More recently, **diffusion models** have emerged as a powerful alternative. These models generate data by learning to reverse a gradual noise-adding process, often modeled as a Markov chain, and typically achieve higher visual quality and stability than GANs.

The widespread availability of generative AI tools has profound forensic implications. Synthetic content can be used for **fraud**, **defamation**, **revenge porn**, and **intellectual property violations**, since artistic style itself cannot be protected by copyright. Moreover, adversarial attacks can be used to **fool image recognition systems**, including those deployed in autonomous vehicles or surveillance infrastructures, by introducing imperceptible perturbations or physical disguises.