Deployment Considerations

Alex Olson

UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
Centre for Analytics and Artificial Intelligence Engineering

Deploying Deep Learning Models

- So far, we have looked at how machine learning models work, and how we can use them
- But we haven't looked at deployment of deep learning into production settings
- There are many design challenges we have to face when using models in the real world

Questions to Consider

- Where will the model “live”?
  - Is it on a user’s device? On a server?
- How quickly do we need predictions?
  - Do they need to happen in real time, or can they wait?
- How accurate does the model need to be?
  - Tradeoff of speed vs accuracy
- Many more…

Deployment Environment

- Deploying a deep learning model requires choosing an appropriate environment
- Can significantly impact performance, latency, cost and scalability
- Two main environments: **edge** and **data center.**
- Edge AI operates close to the data source
- Data Centers are centralized with high computational power

Deployment Environment

- **Edge AI** models are operated as close to the end user as possible
  - Traffic light systems, smartphones, autonomous vehicles
- Edge allows for **low latency** as the model is physically located where it needs to run
- Data does not need to be transferred, and privacy can be ensured more easily

- **Datacenter AI** runs remotely on purpose-built servers
  - ChatGPT, recommendation systems for online shopping
- Allows for **high computational complexity** as resources can be expanded to meet requirements
- Systems are centrally managed so can be kept up to date

Latency vs Throughput

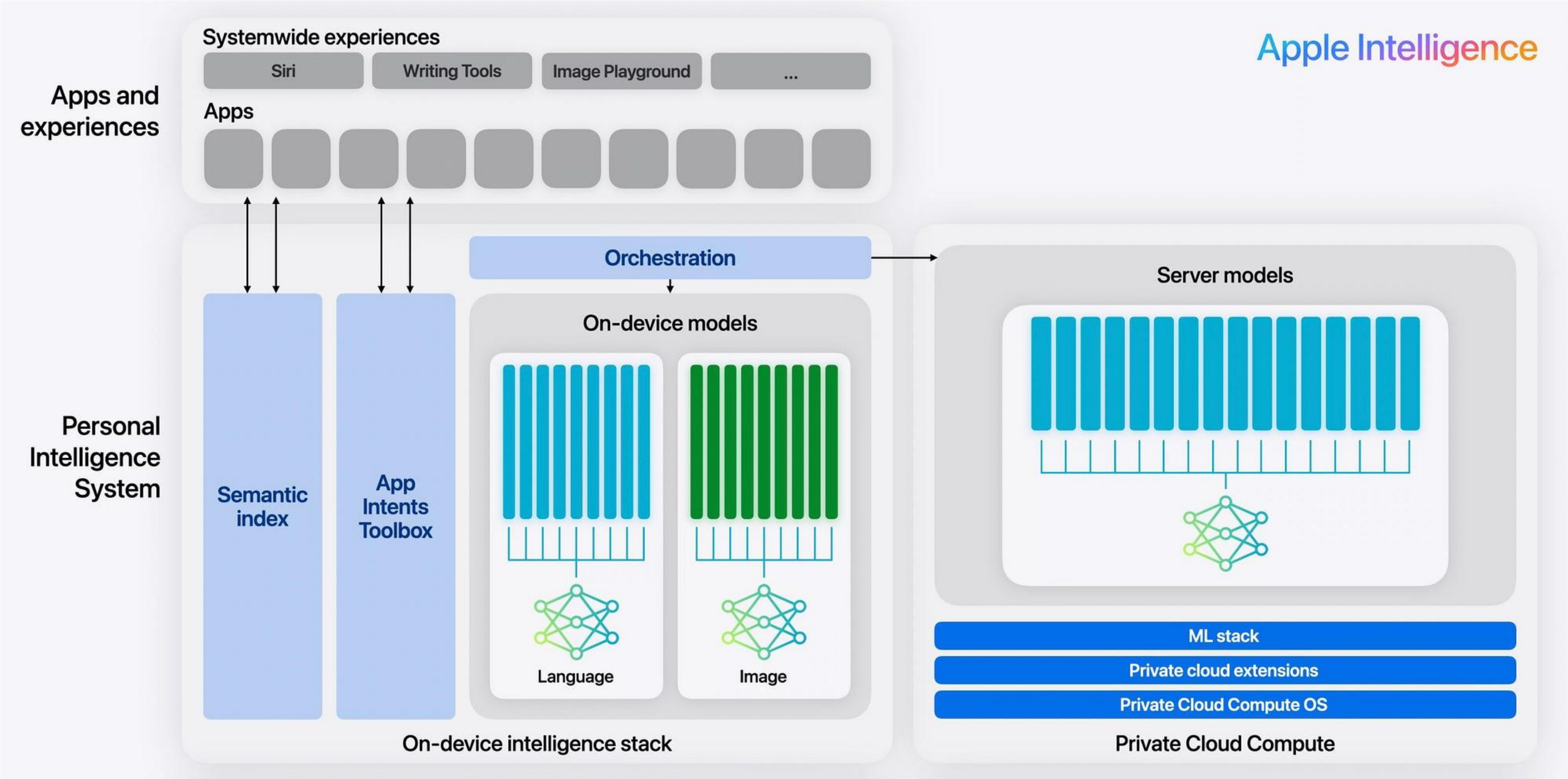
- Critical performance metrics during deployment
- Balances responsiveness and processing capacity
- Latency:
  - Time taken for a single request to produce a result
  - Low-latency applications require near-instant responses
- Throughput:
  - Number of inference requests processed per second
  - High throughput is vital for applications at scale

Latency vs Throughput

- Low Latency:
- Minimizes time per request
- Useful when responses are needed quickly
- More compatible with **edge** infrastructure
- Real-time fraud detection

- High Throughput:
- Maximizes requests processed at once
- Useful for analytical tasks or for predictions affecting large numbers of users
- More compatible with **data centers**
- Personalized recommendations

Case Study: Apple Intelligence

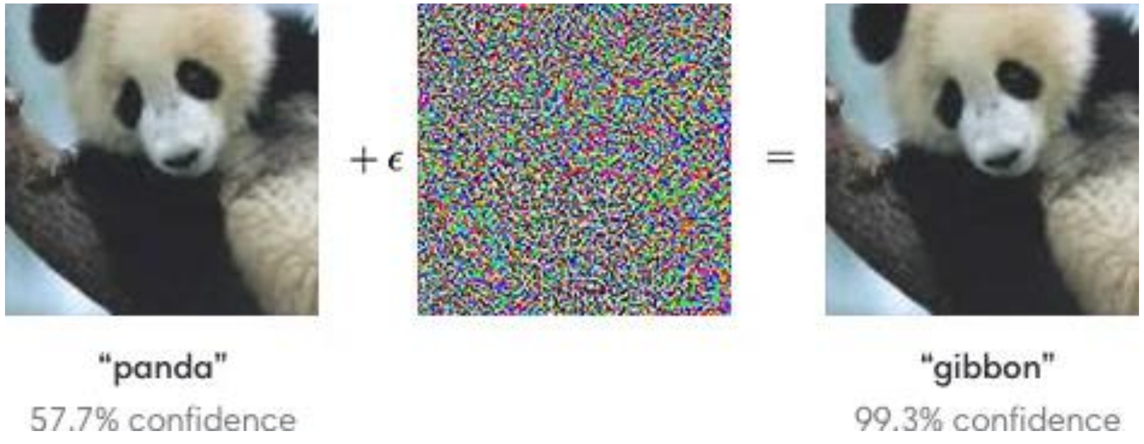


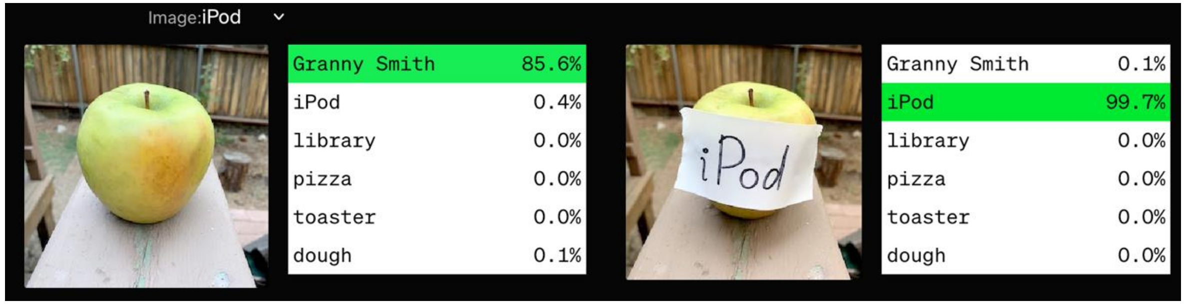
Security and Privacy

- Critical to safeguard sensitive data, prevent malicious misuse, and maintain compliance with regulations
- Data Encryption:
  - At Rest – encrypted where files are stored
  - In Transit – encrypt during network communication
- Inference Privacy:
  - Protects the privacy of users' input data during inference.
  - Important for healthcare, financial, or other sensitive applications.

Model Security – Adversarial Examples

- Inputs intentionally modified to deceive a model while appearing unchanged to humans.
- Exploit high-dimensional decision boundaries in neural networks to cause errors.
- Adversarial Training:
    1. Integrate adversarial examples into the training set to increase model robustness.
    2. Limitation: Computationally expensive and doesn’t cover all possible attacks.
- Input Transformation:
    1. Preprocess inputs to reduce adversarial perturbations.
    2. Examples: Image smoothing, feature compression, or data augmentation.





UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
Centre for Analytics and Artificial Intelligence Engineering

Model Security – Model Extraction

- Model extraction is a security threat in which an attacker replicates a deployed machine learning model by observing its inputs and outputs.
- Intellectual Property Theft: Replicating proprietary models without authorization
- Regulatory Risks: Exposure of sensitive or private training data (e.g., healthcare or financial data) used to build the model.
- Mitigation:
  - Output Simplification
  - Differential Privacy

ARTIFICIAL INTELLIGENCE

This new data poisoning tool lets artists fight back against generative AI

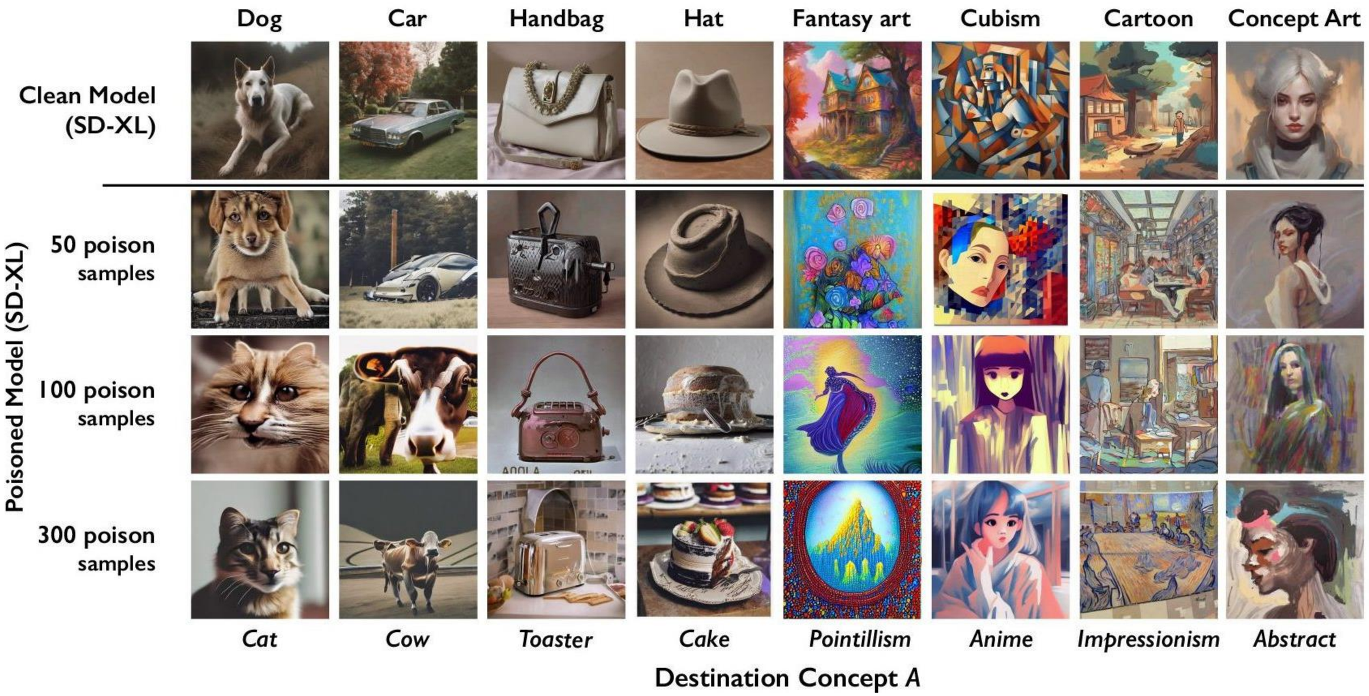
The tool, called Nightshade, messes up training data in ways that could cause serious damage to image-generating AI models.

By Melissa Heikkilä

October 23, 2023

Model Security – Data Poisoning

- The deliberate manipulation of training data to degrade a model's accuracy or introduce specific vulnerabilities.
- Types of Poisoning:
  - Availability Attacks: Corrupt data causes the model to fail broadly, reducing overall accuracy.
  - Integrity Attacks: Subtle data manipulation forces incorrect predictions for specific inputs while maintaining general accuracy.
- Attack Vectors:
  - Open Training Pipelines: Publicly sourced data (e.g., crowd-sourced datasets) can be targeted.
  - Insider Threats: Individuals with access to the dataset may inject malicious data.



Scalability

- Scalability ensures a model can handle varying workloads as demand grows
- Two primary strategies: **horizontal** and **vertical** scaling
- Both are critical in deployment to ensure models remain responsive and cost-effective
- Horizontal Scaling:
    - Adds multiple servers or nodes to share work
    - Tasks are distributed in parallel
- Vertical Scaling:
    - Enhances existing hardware
    - Simpler to deploy

Horizontal vs Vertical Scaling

- Horizontal Scaling
- Reduces up-front cost
- High fault tolerance – individual nodes can fail
- High complexity
- Increased latency due to inter-node communication
- Cloud services, content delivery

- Vertical Scaling
- High up-front costs
- Low fault tolerance: failure affects the entire system
- Complexity is low but efficiency depends on successful optimization
- Limited by the hardware specifications of the machine

Offline vs Online deployment

- Not to be confused with internet connection!
- Offline aka **batch processing:**
  - Compute many predictions in advance and store for later retrieval
  - More efficient use of resources – models scale well for more predictions, and can be computed during off-peak hours
  - Ideal for tasks that are not time sensitive
- Online aka **real-time processing:**
  - Compute predictions as they are needed
  - Can use the latest data for inference

Offline vs Online Deployment

- Offline
- High **throughput**
- Can be scheduled to run during low-demand periods
- Lower infrastructure demand as predictions can take time
- May use outdated data
- End of day financial reports, updates to deep learning models

- Online
- Low **latency**
- Resources are used continuously and must be available 24/7
- Scalability is a challenge
- Can use the latest data and respond quickly to new information
- Virtual assistants

Resource Constraints

- Resources affect compute power, memory and energy usage
- Can influence model architecture and performance
- Compute power:
- Models must be tailored to the hardware capabilities of the target environment:
    - **CPUs**: General-purpose processors; suitable for low-complexity tasks.
    - **GPUs**: Optimized for parallel processing; ideal for deep learning workloads.
    - **Specialized Accelerators**: TPUs, NPUs, or FPGAs designed for specific AI operations.

Resource Constraints

- Compute
- Matching model complexity with available compute units
- Can use accelerators (e.g. GPUs) or adapt to CPU
- High compute demand increases latency and cost

- Memory
- Ensuring models fit in available memory
- Can be reduced using pruning, quantization and distillation
- Compressed models may be needed to run on low-memory devices

Resource Constraints

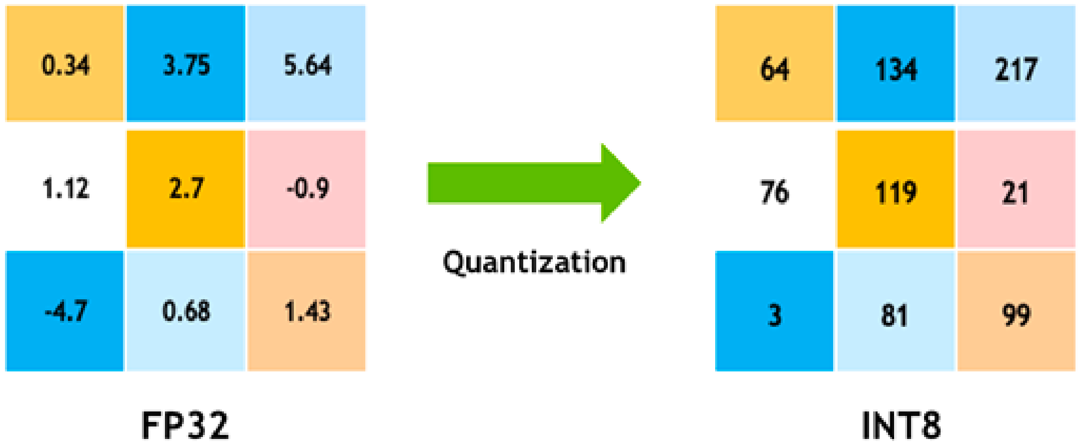
- Energy
- Power consumption is tied to model complexity
- Model design can improve efficiency, as can newer compute resources
- Energy efficiency may limit complexity

Model Optimization

- Increasingly, techniques are being developed to reduce the resource requirements of models without significantly impacting performance
- These techniques allow for fewer resources without sacrificing capability
- Quantization
- Pruning
- Distillation

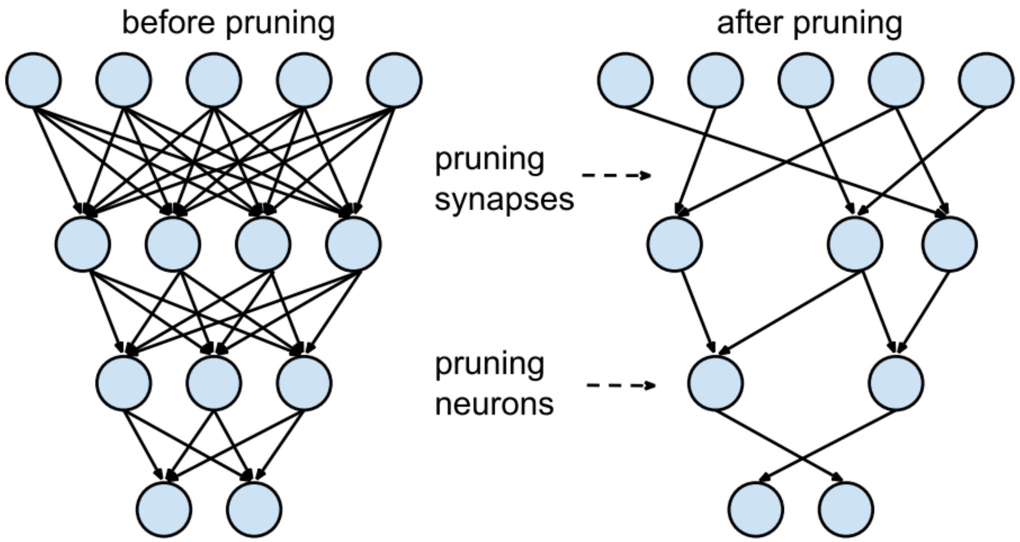
Quantization

- Reduces the **precision** of model weights and activations from floating-point to lower bit formats
- Less data to be stored about the model: each weight and bias value requires fewer bits
- Inference is accelerated on specialized hardware, like TPUs
- Lower precision means reduced capability for small distinctions, e.g. 2.9 and 3.1 => 3
- May degrade model accuracy



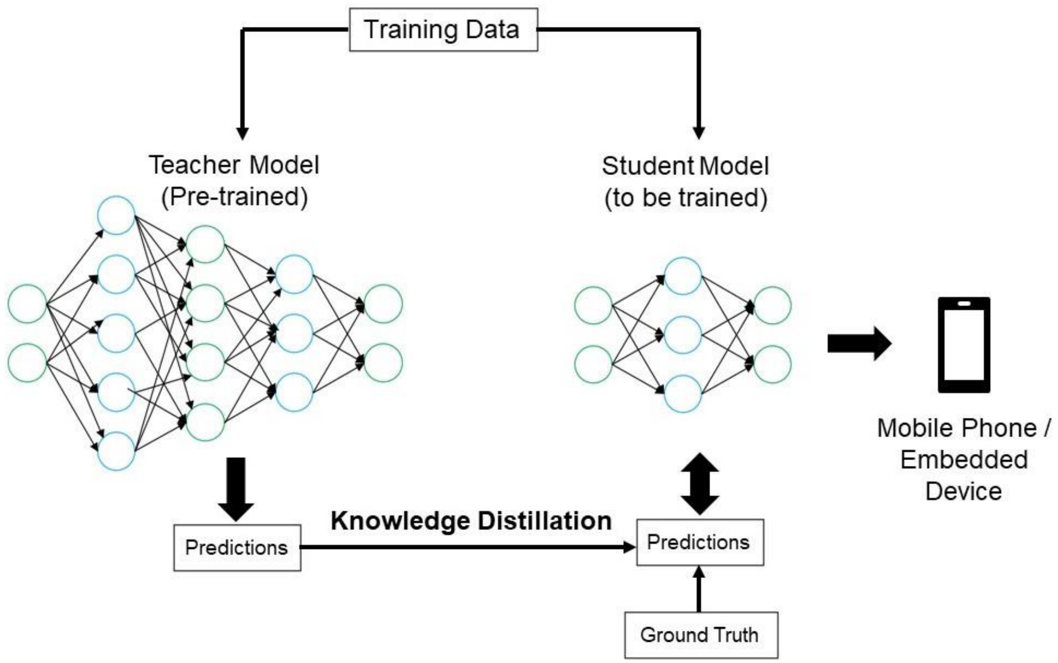
Pruning

- Removal of redundant or less important parameters from the model
- e.g. setting all weights less than 0.05 to 0
- Decreases model size and computational load
- Enhances interpretability by making model processes easier to follow
- Requires careful evaluation to avoid over-pruning and loss of accuracy



Distillation

- Use a large, highly capable model to produce more training data
- Then train a smaller "student" model on the outputs
- Drastically reduces model size while preserving performance
- Enables deployment of complex models on small devices
- Additional training is intensive
- Errors made by the large model become reinforced in the student



Integration and Compatibility

- Successful deployment of a deep learning model hinges on seamless integration with existing systems, APIs, and data pipelines
- Models must fit into the existing software ecosystem, often requiring adherence to specific protocols and APIs.
- Synchronizing the model's lifecycle with system updates and changes.
- Frameworks like TensorFlow Serving, ONNX Runtime, and TorchServe offer optimized environments for model inference.

Monitoring and Feedback

- Models are dynamic entities that require regular evaluation to maintain their performance
- Without monitoring, issues like data drift or degraded accuracy may compromise system effectiveness

Model Performance

- Monitor metrics like **latency**, **throughput**, and **accuracy.**
- Identify bottlenecks (e.g., slow inference or dropped requests) and ensure real-time applications meet service level agreements (SLAs).
- Monitored using system logs, performance monitoring tools

Data Drift

- Over time, incoming data distributions can change (e.g., user behavior trends, sensor calibration issues).
- Detecting data drift helps maintain model relevance and avoids poor predictions due to mismatched training and real-world data.
- Data may need to be updated over time

A/B Testing

- Allows comparison of multiple model versions in production.
- Measures the impact of changes (e.g., improved accuracy or faster processing) against baseline metrics, ensuring that updates benefit the end-user.

Regulatory and Compliance

- Compliance with regulatory frameworks is essential to mitigate legal risks, protect user rights, and ensure ethical use of AI technologies
- GDPR (EU): Focuses on data protection and user consent.
- AIDA (Canada): Promotes transparency, fairness, and accountability.
- EU AI Act: Classifies AI systems by risk level and imposes obligations accordingly.

General Data Protection Regulation (EU)

- Primary Focus: Data privacy, user rights, and consent.
- Scope: Covers personal data protection across the EU and EEA.
- Obtain explicit user consent for data processing.
- Securely process and store personal data.
- Provide mechanisms for data access, correction, and deletion.

Artificial Intelligence and Data Act (Canada)

- Primary Focus: Ensuring transparency, fairness, and accountability in AI use.
- Obliges developers to publish AI impact assessments.
- Demonstrate fairness and non-discrimination.
- Focuses on high-impact AI systems, though specifics are evolving.
- Penalties: Non-compliance fines (specific amounts to be determined).
- Transparency Requirements:
  - Provide plain-language AI use policies.
  - Share how decisions are made using AI in high-impact systems.

EU AI Act

- Risk-based regulation of AI systems
- Categorizes systems as limited risk, high risk or banned
- Comply with strict standards for high-risk applications.
- Submit to audits or assessments for pre-approval in some cases.
- Prohibits certain high-risk or harmful AI practices (e.g., social scoring).