Model Selection

Michelle Marchesini Vanegas

Earlham College

February 27th

Outline

Model Selection

Model selection in computer vision is the process of choosing the most suitable machine learning model for a specific computer vision task from a set of candidate models. This crucial step precedes parameter estimation when the model is not known a priori and is essential for various machine vision applications.

Model Selection

The process of selecting an appropiate model includes:

- Evaluating different model architectures and algorithms specifically designed for visual data processing, such as convolutional neural networks (CNNs), object detection models, or image segmentation networks.
- Considering factors such as image resolution, dataset size, and computational requirements for real-time processing.
- Assessing model performance using metrics relevant to visual tasks, which may include accuracy, precision, recall, and mean average precision (mAP) for object detection.

Model Selection

- Balancing model complexity with performance, as computer vision models can be computationally intensive and may need to run on resource-constrained devices.
- Validating models using techniques like cross-validation or holdout sets, which is particularly important given the high dimensionality of image data.
- Considering the specific requirements such as real-time processing for video analysis or high accuracy for medical image diagnosis

Architecture Choice

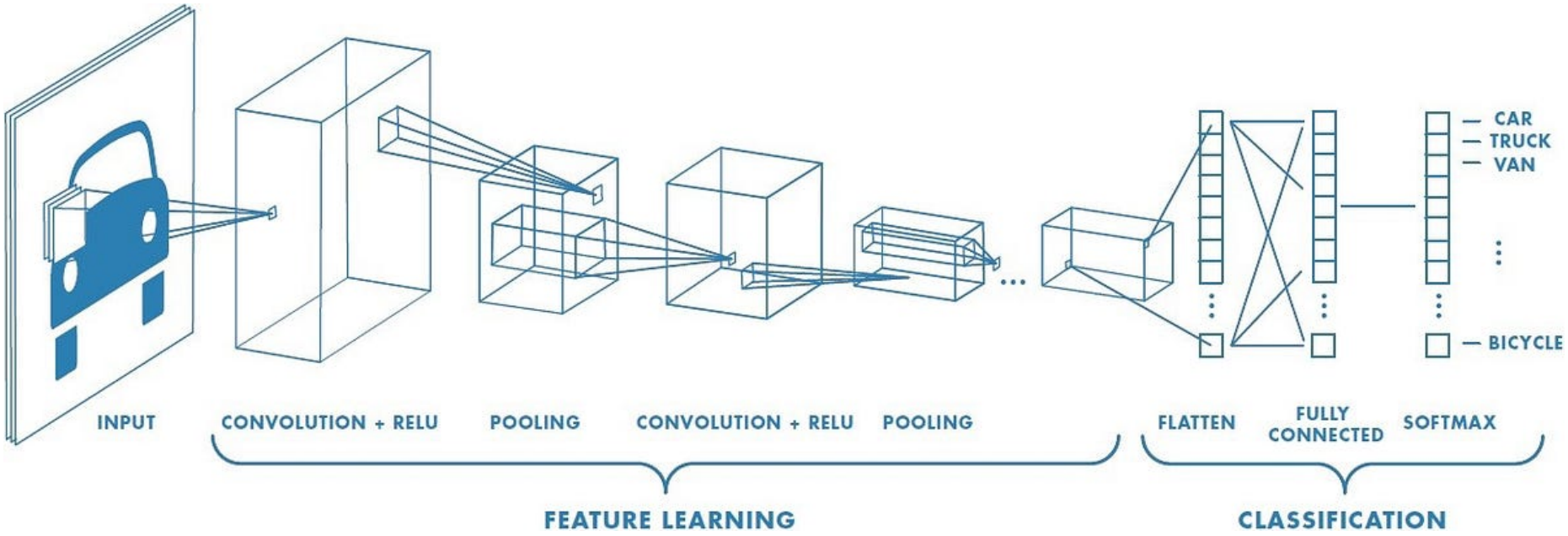
Architecture choice in deep learning refers to selecting the most appropriate neural network structure for a specific task. The main architectures are Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformers, each with distinct characteristics and applications.

Convolutional Neural Networks (CNNs)

CNNs excel in processing grid-like data, particularly images. They are characterized by

- Convolutional layers that apply filters to detect features
- Pooling layers that reduce spatial dimensions.
- Fully connected layers for final predictions

CNNs are widely used for: Image classification, object detection, Medical image analysis, and facial recognition.

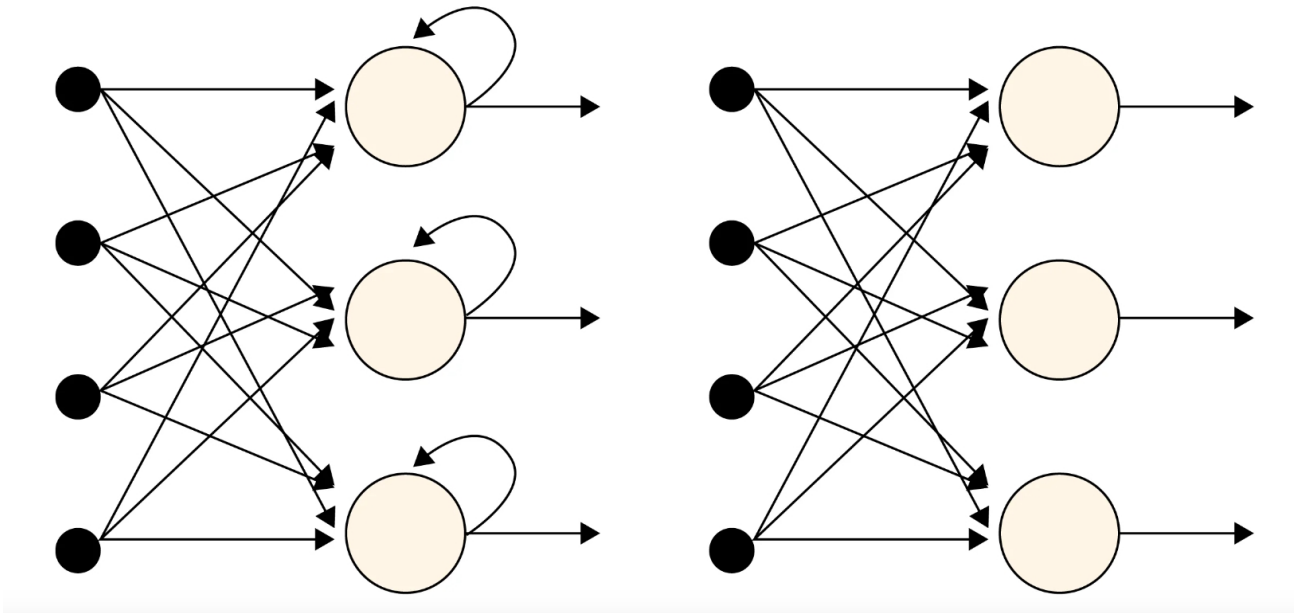


Recurrent Neural Networks (RNNs)

RNNs are designed to handle sequential data by maintaining an internal state or "memory". Key features include:

- Ability to process variable-length input and output
- Shared weights across time steps
- Capture of long-term dependencies in data.

RNNs are commonly applied to Natural language, processing tasks, speech recognition, time series analysis, and sentiment analysis.

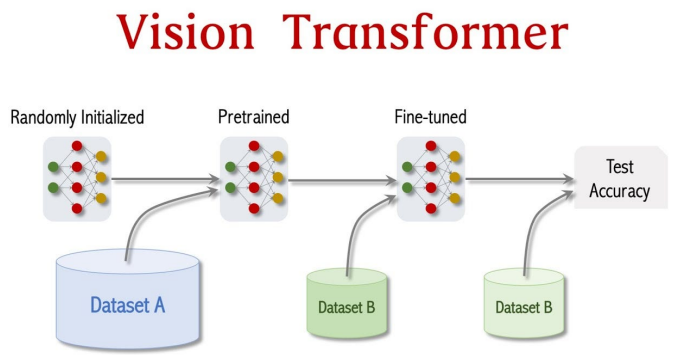


Transformers

Transformers are a type of neural network architecture designed to process sequential data, particularly excelling in natural language processing tasks.

- Self-attention mechanisms
- Parallel processing of input data
- Encoder-decoder architecture
- Positional encoding
- Scalability

Transformers excel in: Machine translation, text summarization, large language models (e.g., ChatGPT), image classification (Vision Transformers)



Architecture Selection Considerations

- **Data type:** CNNs for spatial data, RNNs for sequential data, Transformers for both.
- **Task complexity:** Transformers for complex, long-range dependencies; CNNs or RNNs for simpler tasks.
- **Computational resources:** CNNs and RNNs may be more efficient for smaller datasets, while Transformers often require more compute power.
- **Input size variability:** RNNs and Transformers handle variable-length inputs better than traditional CNNs.
- **Training data availability:** Transformers typically require larger datasets to reach optimal performance.

Pre-trained models

Pre-trained models are machine learning models, typically deep neural networks, that have been trained on large datasets to perform specific tasks. These models have already learned to recognize patterns and features from vast amounts of data, providing a foundation of knowledge that can be leveraged for various applications.
Pre-trained models are particularly useful in scenarios where:

- Labeled data is scarce or difficult to obtain
- Computational resources are limited
- Rapid prototyping or proof of concept is required
- Domain expertise is needed but not readily available

Key Characteristics

- **Two-step process:** Pre-training on large datasets, followed by fine-tuning for specific tasks.
- **Versatility**: They can be applied to various domains, such as natural language processing, computer vision, and object detection.
- **Transfer learning:** The ability to adapt knowledge from one domain to another, enabling rapid adaptation for distinct tasks.
- **Reduced development time:** They provide a head start in model development, significantly reducing the time and resources needed to build capable models.
- **Improved performance:** Often outperforming models trained from scratch, especially in tasks requiring deep data understanding.

Lightweight Models

Lightweight models are machine learning algorithms designed to operate efficiently in resource-constrained environments while maintaining good performance.
These models offer benefits such as improved efficiency, reduced latency, lower costs, enhanced privacy, and better scalability for deployment in resource-limited environments

- Reduced size and complexity compared to traditional models.
- Lower computational and memory requirements.
- Faster inference times and real-time processing capabilities.
- Suitability for edge computing, IoT devices, and mobile applications

Trade-Offs

- **Performance vs. Resource Usage**
- **Accuracy vs. Speed**
- **Generalization vs. Specialization**
- **Cost vs. Capability**
- **Flexibility vs. Efficiency**
- **Environmental Impact**

Hyperparameter Tuning

Hyperparameter tuning is the process of identifying and selecting the optimal hyperparameters for training a machine learning model. It aims to minimize the model's loss function, thereby maximizing its performance and accuracy.
Hyperparameter tuning is crucial because:

- It improves model performance and generalization.
- It helps prevent underfitting and overfitting.
- It can reduce computational costs and training time.

Common hyperparameters that often require tuning include learning rate, number of hidden layers, batch size, and regularization parameters.
As of 2025, hyperparameter tuning remains an essential part of the machine learning workflow, with ongoing research and development in automated tuning methods and efficient search strategies.

Key Aspects

- **Optimization techniques:** Grid search, Random search, Bayesian optimization
- **Cross-validation**
- **Gradual narrowing** .
- **Early stopping**
- **Automated solutions**

Common hyperparameters that often require tuning include learning rate, number of hidden layers, batch size, and regularization parameters.
As of 2025, hyperparameter tuning remains an essential part of the machine learning workflow, with ongoing research and development in automated tuning methods and efficient search strategies.