Section 4: Convolutional Neural Networks (CNNs)

Advanced Image Processing for Solar Panel Defect Detection

Dr Bibin Wilson & Prof Anand Singh

September 10, 2025

Indian Institute of Technology Bombay

Section Overview

CNN Fundamentals

Convolution Operation: Mathematical Foundation

2D Discrete Convolution:

$$(I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) \cdot K(m,n)$$

where:

- I: Input image
- K: Kernel/Filter
- (i,j): Output position

Key Properties:

- **Parameter Sharing**: Same kernel across image
- **Translation Equivariance**: Feature detection anywhere

Output Dimensions:

$$O = \frac{I - K + 2P}{S} + 1$$

where:

- I: Input size
- K: Kernel size
- P: Padding
- S: Stride

Implementation Details:

`04_cnn_solar_advanced.ipynb`

Manual convolution & visualization

Common Filters and Edge Detection

Edge Detection Kernels:

Sobel X (Vertical Edges):

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Sobel Y (Horizontal Edges):

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Laplacian (All Edges):

$$\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$$

Solar Panel Applications:

- **Crack Detection**: Edge filters
- **Hot Spot**: Gaussian blur + threshold
- **Dust/Dirt**: Texture analysis
- **Cell Boundaries**: Grid detection

Filter Visualizations:

`04_cnn_solar_advanced.ipynb`

Applied to solar panel images

Pooling Layers

Max Pooling:

$$y_{ij} = \max_{(m,n)\in R_{ij}} x_{mn}$$

Average Pooling:

$$y_{ij} = \frac{1}{|R_{ij}|} \sum_{(m,n)\in R_{ij}} x_{mn}$$

Global Average Pooling:

$$y_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} x_{c,i,j}$$

Benefits:

- Dimension reduction
- Translation invariance
- Computational efficiency
- Overfitting reduction

Modern Alternatives:

- Strided convolutions
- Dilated convolutions
- Adaptive pooling

CNN Architectures Evolution

Classic Architectures Overview

LeNet-5 (1998):

- 7 layers, 60K parameters
- Conv → Pool → Conv → Pool → FC
- Handwritten digit recognition

AlexNet (2012):

- 8 layers, 60M parameters
- ReLU activation
- Dropout regularization
- GPU training

VGGNet (2014):

- 16-19 layers, 138M parameters
- 3×3 convolutions only

GoogLeNet/Inception (2014):

- 22 layers, 5M parameters
- Inception modules
- Multiple kernel sizes
- 1×1 convolutions

ResNet (2015):

- 50-152 layers, 25M-60M parameters
- Skip connections
- Batch normalization
- Identity mappings

All Architectures:

`04_cnn_solar_advanced.ipynb`

ResNet: Residual Learning

Residual Block:

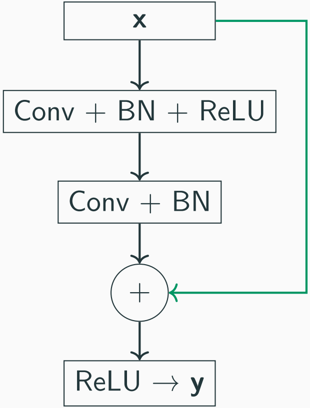
$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}$$

Identity Shortcut:

- No extra parameters
- No computational complexity
- Gradient highway

Projection Shortcut: When dimensions change:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}$$



Inception Module

Multi-Scale Feature Extraction:

- 1×1 convolutions (dimensionality reduction)
- 3×3 convolutions (medium features)
- 5×5 convolutions (large features)
- 3×3 max pooling (context)

Computational Efficiency:

- 1×1 conv before expensive ops
- Reduced parameters
- Parallel processing

Architecture Benefits:

- Captures multiple scales
- Network decides optimal path
- Efficient computation
- Better gradient flow

Inception Implementation:

`04_cnn_solar_advanced.ipynb`

Complete module with solar applications

Solar Panel Defect Detection

Solar Panel Defects Classification

Common Defect Types:

- **Cracks**: Micro/macro fractures
- **Hot Spots**: Overheating cells
- **Dust/Soiling**: Surface contamination
- **Delamination**: Layer separation
- **Discoloration**: EVA browning
- **Snail Trails**: Silver paste issues

Detection Challenges:

- Variable lighting conditions
- Multiple defect types
- Small defect sizes
- Class imbalance

CNN Architecture Design:

- Feature extraction backbone
- Multi-scale processing
- Attention mechanisms
- Class-weighted loss

Data Augmentation:

- Rotation (panel orientation)
- Brightness/contrast (lighting)
- Random crops (defect location)
- Synthetic defect generation

Complete Pipeline:

`04_cnn_solar_advanced.ipynb`

Transfer Learning for Solar Panels

Pretrained Backbones:

- ResNet50/101 (ImageNet)
- EfficientNet (Better accuracy/speed)
- MobileNet (Edge deployment)
- Vision Transformer (State-of-art)

Fine-tuning Strategies:

1. Freeze backbone, train classifier
2. Unfreeze top layers gradually
3. Full network fine-tuning
4. Discriminative learning rates

Domain Adaptation:

- ImageNet $\rightarrow$ Solar panels
- RGB $\rightarrow$ Thermal imaging
- Visible $\rightarrow$ Electroluminescence

Performance Metrics:

- Accuracy: Overall correctness
- Precision: Defect identification
- Recall: Defect coverage
- F1-Score: Balanced metric
- mAP: Multi-class performance

Advanced CNN Techniques

Attention Mechanisms in CNNs

Channel Attention (SE-Net):

$$\mathbf{s} = F_{ex}(F_{sq}(\mathbf{U}))$$

$$\tilde{\mathbf{U}}_c = \mathbf{s}_c \cdot \mathbf{U}_c$$

Spatial Attention:

- Focus on relevant regions
- Suppress background
- Improve localization

CBAM (Combined):

- Channel attention $\rightarrow$ Spatial attention
- Sequential refinement
- Minimal overhead

Self-Attention:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Benefits for Solar:

- Focus on defect regions
- Handle multiple defects
- Improve small defect detection
- Better interpretability

Attention Implementations:

`04_cnn_solar_advanced.ipynb`

Grad-CAM: Visual Explanations

Gradient-weighted Class Activation:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

$$L_{Grad-CAM}^c = ReLU\left(\sum_k \alpha_k^c A^k\right)$$

Process:

1. Forward pass to get prediction
2. Compute gradients of class score
3. Weight feature maps by gradients
4. Apply ReLU to get heatmap

Applications:

- Defect localization
- Model debugging
- Trust building
- Feature importance

Extensions:

- Grad-CAM++
- Score-CAM
- Layer-CAM
- Integrated Gradients

Visualization Tools:

04_cnn_solar_advanced.ipynb

Model Optimization for Deployment

Quantization:

- FP32 $\rightarrow$ INT8 ($4\times$ smaller)
- Dynamic vs Static
- Quantization-aware training
- Minimal accuracy loss

Pruning:

- Structured (channels/filters)
- Unstructured (weights)
- Magnitude-based
- Gradual pruning

Knowledge Distillation:

- Teacher-student framework
- Soft targets
- Feature matching
- Attention transfer

Edge Deployment:

- ONNX export
- TensorRT optimization
- Mobile frameworks
- Real-time constraints

Optimization Pipeline:

`04_cnn_solar_advanced.ipynb`

Practical Implementation

Data Pipeline and Augmentation

Efficient Data Loading:

- Multi-worker loading
- Prefetching
- Memory pinning
- Cache optimization

Augmentation Strategy:

- Geometric: Rotation, flip, crop
- Photometric: Brightness, contrast
- Advanced: MixUp, CutMix
- Domain-specific: Synthetic defects

Class Imbalance:

- Weighted sampling
- Focal loss
- SMOTE for images
- Cost-sensitive learning

Validation Strategy:

- K-fold cross-validation
- Stratified splits
- Time-based splits
- Geographic splits

Training Best Practices

Learning Rate Scheduling:

- Warmup phase
- Cosine annealing
- OneCycle policy
- ReduceLROnPlateau

Regularization:

- Dropout (spatial/standard)
- Weight decay
- Data augmentation
- Label smoothing
- Stochastic depth

Mixed Precision Training:

- FP16 computation
- FP32 master weights
- Loss scaling
- 2-3$\times$ speedup

Monitoring:

- TensorBoard logging
- Gradient norms
- Weight distributions
- Activation statistics

Training Pipeline:

04_cnn_solar_advanced.ipynb

Real-time Inference System

Pipeline Components:

1. Image preprocessing
2. Model inference
3. Post-processing
4. Result aggregation
5. Alert generation

Optimization Techniques:

- Batch processing
- Async inference
- Model caching
- GPU utilization

Performance Metrics:

- Throughput (images/sec)
- Latency (ms/image)
- Memory usage
- Power consumption

Deployment Options:

- Cloud (scalable)
- Edge (low latency)
- Hybrid (optimal)
- Drone-mounted

Summary: CNNs for Solar Applications

Key Concepts:

- Convolution operations
- CNN architectures evolution
- ResNet and Inception
- Transfer learning
- Attention mechanisms
- Visual explanations

Solar Applications:

- Defect detection
- Classification pipeline
- Real-time monitoring
- Predictive maintenance

Advanced Techniques:

- Model optimization
- Grad-CAM visualization
- Edge deployment
- Production systems

Complete Implementation:

`04_cnn_solar_advanced.ipynb`

Next: RNNs & LSTMs for Time Series

`Section 5 & 05_lstm_energy_advanced.ipynb`