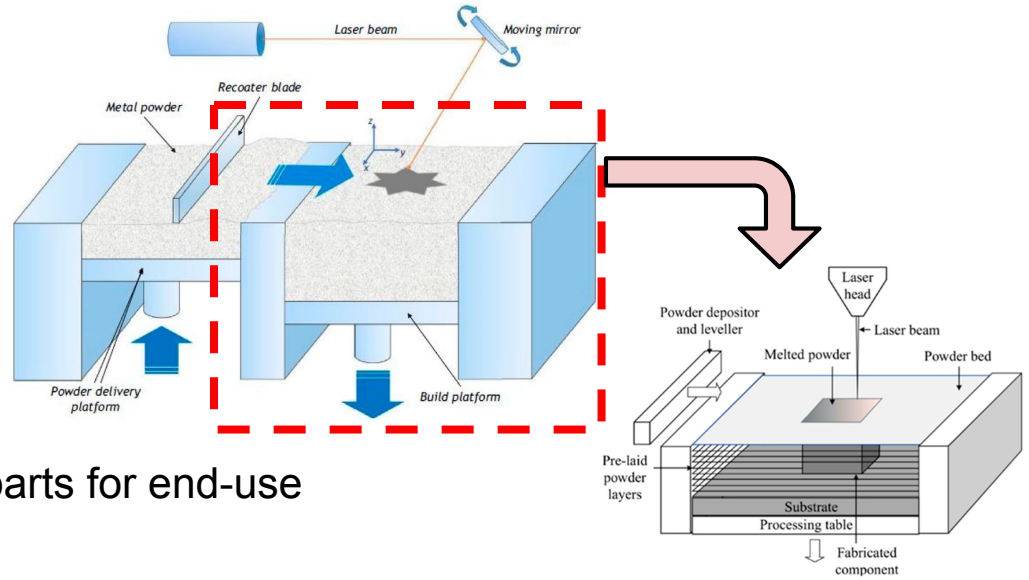
❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

(Dynamic Segmentation CNN, Luke Scime, ORNL)

❖ Powder-Bed AM Overview (Ch.1.1)



- ➢ Powder-Bed AM builds complex 3D components layer-by-layer
  - ■ Step 1. A metal powder layer is spread via a recoater on a bed
  - ■ Step 2. The cross-section is fused or binded using a laser/e-beam
  - ■ Step 3. The substrate is lowered by one layer thickness
  - ■ Step 4. Process repeats until the build is completed
- ➢ Initially used for prototyping, it now produces “near-net-shape” parts for end-use
  - ■ Ideal for low-to-medium volume production
  - ■ Well-suited for highly engineered or customized parts (e.g., Aerospace, Medical implants, Nuclear, Automotive)
- ➢ These industries require: Higher quality, Better pedigree traceability, and Lower rejection rates
  - ■ Traditional open-loop control and ex-situ inspection cannot meet these requirements
    - **Open-loop Control:** A control scheme in which the printing process proceeds without using sensor feedback, relying solely on pre-set process parameters (e.g., laser power, scan speed, hatch spacing) from start to finish
      - No monitoring or adjusting the process parameter during the build allowed
      - Anomalies that arise during the build cannot be corrected in real time, leading to higher variability in part quality
    - **Ex-situ inspection:** A post-process inspection method where the completed build is examined outside the machine (“ex-situ”)
      - Using CT scans, optical measurements, surface analysis, or metallographic examination after sectioning
      - Defects are checked only after the entire build → significant time and material waste
  - ■ We need something that allow real-time control to guarantee higher quality and traceability without wasting resources

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

(Dynamic Segmentation CNN, Luke Scime, ORNL)

❖ Powder-Bed AM Overview (Ch.1.1)

➢ Some anomalies occur at very short spatiotemporal scales during fusion or binding.
But, many are persistent and detectable in layer-wise imaging, which motivates real-time anomaly detection!

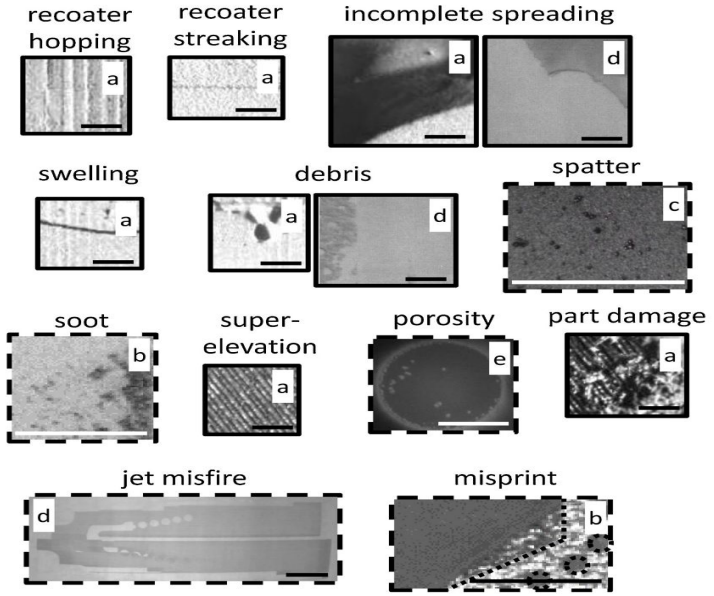
→ The focus of this work is the development of an algorithm for the layer-wise detection of these **surface-visible anomalies** via the real-time automated analysis of imaging data collected from powder bed metal AM processes

→ This work proposes a CNN-based framework to achieve this objective!

❖ Surface-Visible Powder Bed Anomalies (Ch.1.2)

➢ Most anomalies are visually detectable in layer images;
this work aims to automatically classify them **pixel-wise!**

- Recoater-related defects: Hopping/Streaking → Global & Regional pattern
- Powder spreading anomalies: Incomplete Spreading, Debris, Spatter/Soot
  → Regional & Local
- Part-geometry related defects: Swelling, Part Damage, Porosity
  → Regional & Local
- Path-related defects: Jet Misfire/Misprint → Regional & Local



❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM (Dynamic Segmentation CNN, Luke Scime, ORNL)

❖ Existing Work & Limitations (Ch.1.3)

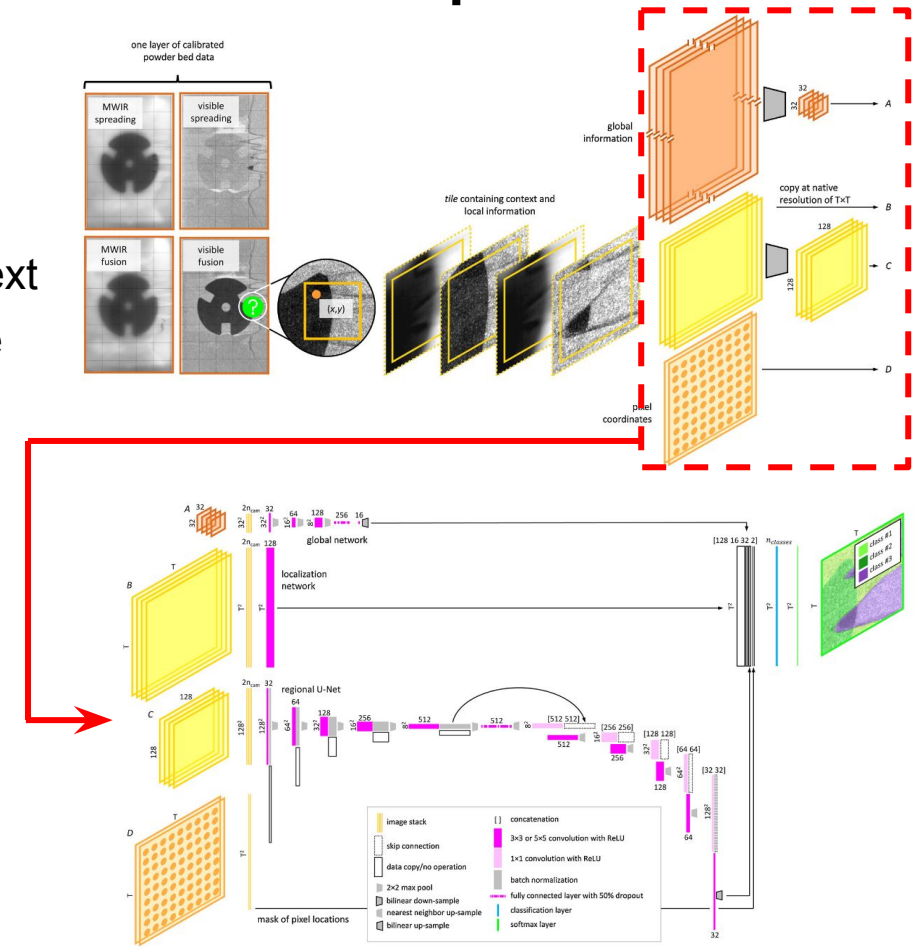
- ➢ Focusing on improvements in sensing technologies (Flash illumination, fringe projection, high-resolution scanning,...)
  - This can improve the raw data quality
  - But, it still requires robust automated data analysis to classify anomalies
- ➢ Data analysis
  - Traditional algorithms: Geometry comparison between printed layers and CAD-geometry
    - **Hand-crafted Feature Extraction:** Human manually segmentize & Compare them via Computer vision (Limited scalability)
    - Only few limited anomalies are detectable (e.g., hopping, streaking, swelling, debris?)
    - Micro-level spatial anomalies are not detectable → highly hinge the quality!
    - **Only one class can be detectable at a time**: Poor multi-class detection
  - ML (SVM, RF, Bayesian Mode, and etc.) / DL (CNN, U-Net) approaches
    - [ML] Still need features/targets which have to be hand-crafted & Only one or Small number of anomaly classes are detectable
    - [DL] Able to self-extract important features & Possible for multi-class detection, but
    - Highly risk for overfitting & Lack of transferability across machines (e.g., cat or dog?)
    - Only do patch-wise classification

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

(Dynamic Segmentation CNN, Luke Scime, ORNL)

❖ DSCNN (Ch.1.4)

- ➢ Take a multi-channel image as input
- ➢ Combine 3-legs (U-Net, Global/Local CNN) to aware multi-scale context
- ➢ Integrate Pixel-coordinate layer to use location-based prior knowledge
  - ■ Coordinate-aware segmentation (x,y channels)
- ➢ Use pixel-level data instead of patch-level
  - ■ One segmentation by hand in a single layer can expand the data a lot
- ➢ Include a mixed layers from different machines
  - ■ Could have all the classes, but imbalanced
  - ■ Can train a model that has robust performance across 6 AM systems
  - ■ Significant improvements over MsCNN in accuracy, localization, and speed



❖ Goals & Novelty of DSCNN (Ch.1.4)

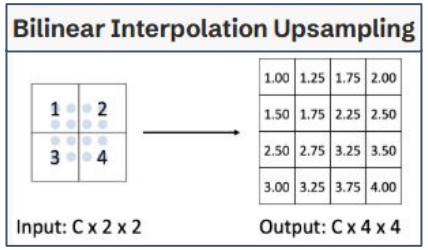
1. Real-time anomaly classification, even at high camera resolutions
2. Pixel-wise semantic segmentation at native image resolution
3. Rapid transfer learning between machines with limited training data
4. Natural support for multi-sensor fusion within the architecture

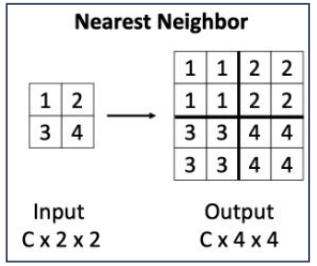
❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

(Dynamic Segmentation CNN, Luke Scime, ORNL)

❖ Experimental Settings (Ch.2)

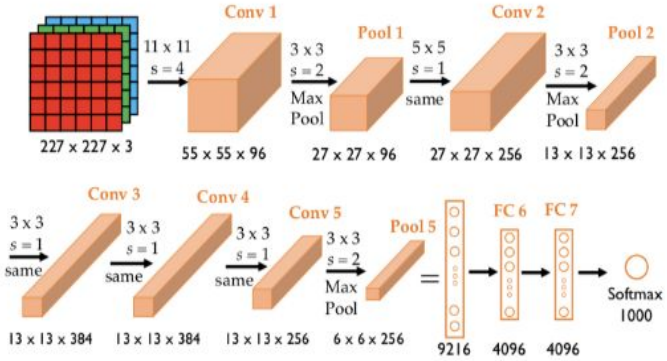
- ➢ DSCNN is validated on six different powder-bed AM machines across three technologies (L-PBF, EB-PBF, Binder Jetting), demonstrating strong generalizability
  - 6 machine types: 3 L-PBF, 1 EB-PBF, 2 Binder Jetting
- ➢ Each layer provides two images per camera—post-fusion and post-spreading—which capture thermal/morphological states before and after recoating
  - All images are calibrated for FOV distortion, Perspective distortion, Lighting correction via OpenCV 3.3.1 & Open3D 0.4
  - DSCNN input uses calibrated 8-bit images: 0~255 intensity range & grayscale
- ➢ Machine-Specific Configurations
  - EOS M290 (CMU): 1 MP visible-light camera, Full-bed FoV (250×250 mm), Side-mounted LEDs, 290 μm/pixel Resolution
  - ConceptLaser M2 (MDF): 5 MP visible-light camera, Full-bed FoV (240×240 mm), Top LEDs, 110 μm/pixel Resolution
  - Renishaw AM250 (MDF): 15 MP high-res Pixelink camera, Small FoV (30×30 mm), 20 μm/pixel Resolution (finest)
  - Arcam Q10 (MDF): 5 MP **NIR** camera, 70% of bed FoV (200×200 mm), Imaging via emitted light (no external lighting), High-temperature environment, 120 μm/pixel Resolution
  - ExOne M-Flex (MDF): 10 MP visible + 0.3 MP **MWIR** dual-camera, 70% of bed FoV (250×400 mm), 130 μm/pixel Resolution
  - ExOne Innovent (MDF): 20 MP visible camera, Very high resolution of 30 μm/pixel
- ➢ Images from low MP cameras are upsampled to match visible images → all images have same size
  - NN, Bilinear, Bicubic Interpolation, Transposed Convolution…
  - DSCNN takes multi-scale & multi-channel images as an input data
- ➢ Use CAD-geometry (design intent) images to augment final segmentation results by calibrating predictions
  - Do not use this as a part of training data (This is highly likely to cause over-fitting)





❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Overview of relevant knowledge & Intuition (Ch.3.1)



➢ Role of ML / DL in AM domain

- ML models are used to predict/classify QoI (e.g., anomaly presence and type)
  - Training requires ground-truth labels in the training data: often hand-crafted (i.e., representative data annotated by AM experts or other independent processes)
- DL: self feature extraction (often outperforming hand-crafted features) and learns the model
  - CNN is a special architecture that is well-suited for image data
    - It perform element-wise Convolution operations to extract local patterns and spatial relationship in global
    - It builds higher-level features across layers (“Feature map”)

➢ Goal of this work: semantic segmentation

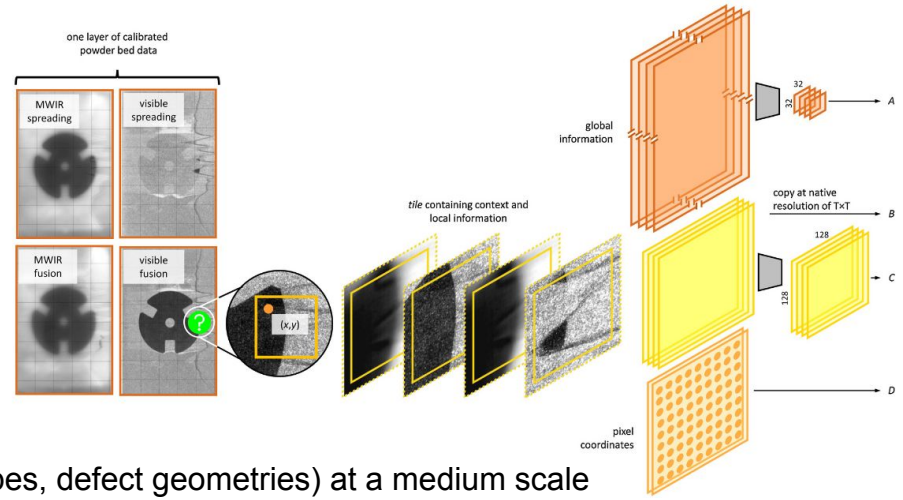
- Classify each pixel to one of several classes: powder, part, incomplete spreading, swelling, spatter, debris, and etc.
  - Pixel labels depend on the pixel’s own intensity and the context in its neighborhood
  - Therefore, the model must combine local and larger-scale contextual information
- Limitations of the previous approach: AlexNet-based MsCNN (Dec, 2018)
  - Encode multi-scale context by feeding three patches(or tiles) of different spatial scales into the **three RGB channels of AlexNet**
    - Apply ‘up-sampling’ in smaller patches to make them all in the same resolution
  - **In natural images, RGB channels are physically correlated at the same spatial location**
    - Most AlexNet features were learned on data where RGB channels align spatially
  - In the MsCNN, however, **different patch sizes** occupy each channel, breaking the assumption of channel-wise spatial correlation
    - When reusing AlexNet for multi-scale patches, the representation becomes suboptimal and **transfer learning is less effective**

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Overview of relevant knowledge & Intuition (Ch.3.1)

➢ Core DSCNN idea: three parallel legs for multi-scale context

- Design the architecture to explicitly separate and process information at 3 different size scales via 3 parallel(independent) “legs”
  - Global CNN leg **(stack A)**: 32×32 downsampled feature maps
    - Very shallow Neural Network layer
    - Captures large-scale conditions (illumination, major powder spreading anomalies, and feedstock-related behavior)
    - Its output is later broadcast back to each pixel as global context
  - Localization leg **(stack B)**: native-resolution tiles (T×T with padding)
    - Include highly local features around each pixel
    - Independent of resolution or camera configuration
    - Support cross-machine applicability
  - Regional U-Net leg **(stack C)**: 128×128 downsampled feature maps
    - Use a U-Net architecture to capture morphological information (e.g., part shapes, defect geometries) at a medium scale
    - Preserve approximate spatial location via skip connections



- Combining these legs, each pixel is classified using a multi-scale feature vector that encodes local, regional, and global info

➢ Tile-wise processing: GPU and receptive-field considerations

- DSCNN processes input images tile-wise:
  - Large powder-bed images (10 MP or more) are broken into tiles to fit into GPU memory and form mini-batches
  - This makes the method feasible on desktop-level GPUs
- Tile resizing also allows control over the effective receptive field without changing the network architecture:
  - Adjusting tile size changes how much of the physical build area a kernel “sees”
  - This is crucial for transferring models across machines with different resolutions and fields of view (Don’t change kernel size, pooling stride, etc.)

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Manual Ground Truth Data Labeling (Ch.3.2)

➢ Training a CNN requires a significant amount of ground truth data, typically 10^5 ~ 10^7 targets

- This was avoided in the previous work (patch-wise) by Transfer learning
  - Low-level features were learned using a large but generic dataset
  - Only the final layers of the CNN were learned using the smaller dataset of interests
- DSCNN can't leverage generic TL
  - Low-level features are not able to be learned through generic dataset since DSCNN predicts pixel-wise!
  - Therefore, it needs a huge dataset!

➢ All pixels could be labeled through manual annotation on multi-layer, multi-build, and multi-machine layer images

- Annotators labeled pixels using both calibrated post-spreading and post-fusion images
- All cameras' images used if multiple cameras exist in a machine → DSCNN uses the full input image stack per layer
- Guidelines for Effective Labeling
  - Ensure representation of all anomaly classes
    - Class balance not required / Minimum 100,000 labeled pixels per class recommended
  - Variability > Quantity: capturing rare events and diverse geometries is crucial
  - Avoid labeling from only a single build
    - Multiple builds introduce variation in lighting, powder condition, etc. → Necessary for robustness in the global network
  - When multiple classes apply to one pixel, Label should reflect the more severe anomaly (e.g., part damage > debris)
  - Ensure spatial coverage across the entire powder bed: Must label samples from all regions (center, edges, corners)
    - Pixel coordinates (x, y) are explicit network inputs
- Strong class imbalance in pixel-wise → Loss function was customized to reflect this imbalance
- Partial Labeling per layers allowed → Unlabeled pixels are excluded in the loss function

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Manual Ground Truth Data Labeling (Ch.3.2)

Table 2
Accounting of ground truth data used for training. The balance of the training data consists of unlabeled pixels.

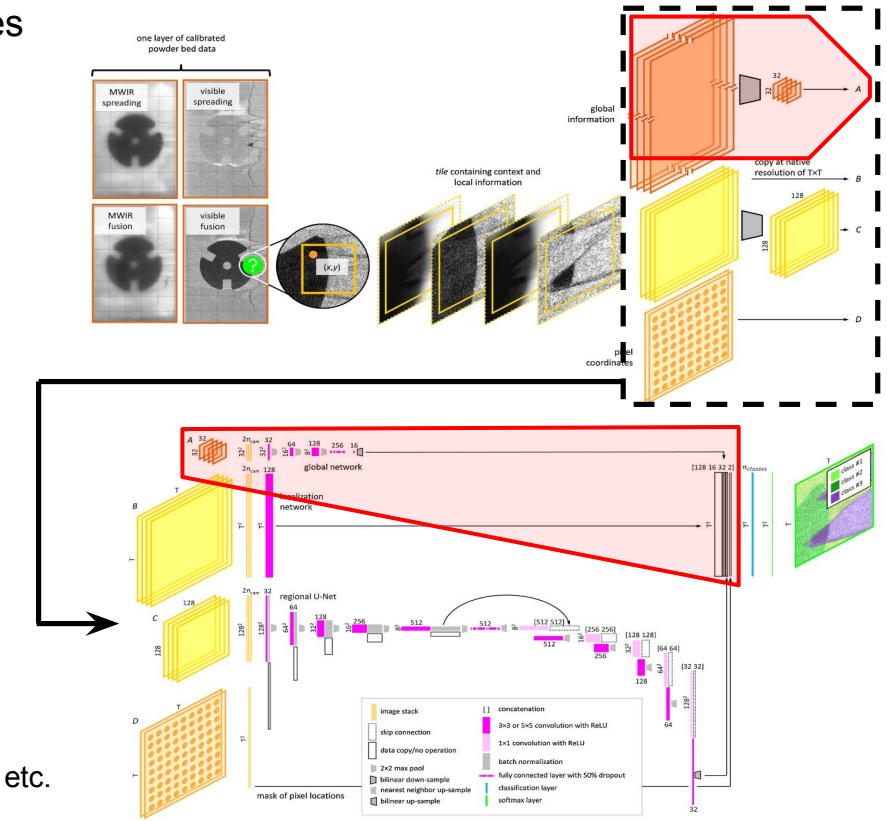
	EOS M290	ConceptLaser M2	Renishaw AM250	Arcam Q10	ExOne M-Flex	ExOne Innovent
Number of Builds	8	12	1	1	10	10
Number of Layers	37	23	3	4	14	13
Number of Pixels (10^6)	33	90	8	7	36	132
Anomaly Class						
Powder	83.2 %	71.8 %	73.5 %	44.7 %	63.5 %	80.5 %
Part	3.1 %	12.8 %	25.2 %	41.1 %	17.6 %	13.1 %
Recoater Hopping	1.3 %	–	–	–	–	–
Recoater Streaking	0.7 %	0.2 %	–	–	0.6 %	0.5 %
Incomplete Spreading	5.2 %	3.0 %	–	–	4.0 %	5.1 %
Swelling	0.1 %	0.1 %	0.1 %	–	–	–
Spatter	–	–	1.2 %	–	–	–
Soot	–	0.2 %	–	–	–	–
Debris	2.7 %	0.4 %	–	–	3.1 %	–
Super-Elevation	1.2 %	0.4 %	–	–	–	–
Part Damage	0.0 %[a]	–	–	–	–	–
Porosity		–	–	0.2 %	–	–

[a] The *part damage* class is not directly learned, it is only predicted according the heuristics outlined in Section 3.8.

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Construction of the Input Layer & DSCNN architecture & Pixel-wise classification (Ch.3.3~5)

- ➢ DSCNN is a two-blocked NN architecture
  - 1st block extracts multi-scale information from raw data (4 image stacks) → These are input layers!
  - 2nd block combines pixel-wise information of input data & process them through NN for classification
- ➢ Input layers: 4 image stacks that together encode multi-scale, multi-sensor, and spatial prior information
  - Multi-scale: Localized network + Globally downsampled layer per image + Regionally focused layer via U-Net
  - Multi-sensor(camera): post-spreading + post-fusion/binder-deposition images
  - Spatial prior information: normalized pixel coordinates
- ➢ Each layer image is split into tiles of size T × T (T: Hyper-parameter)
  - Reduces GPU memory (e.g., Mini-batch)
  - Allows flexible control of receptive field via tile resizing
- ➢ 3-legs architecture
  - Global Image Stack (A):
    - Load the full-resolution layer images (post-spreading + post-fusion images per camera)
    - **Downsample** them to 32×32 using bilinear interpolation
    - Input: 32 × 32 × 2n_cam
    - Output: **T×T×16 feature map**
    - Captures the **largest-scale, global context**, such as:
      - Global environmental variation like Illumination level changes (e.g., room lights on/off)
      - Powder bed flatness or large disturbances, Feedstock-level texture/color changes, and etc.

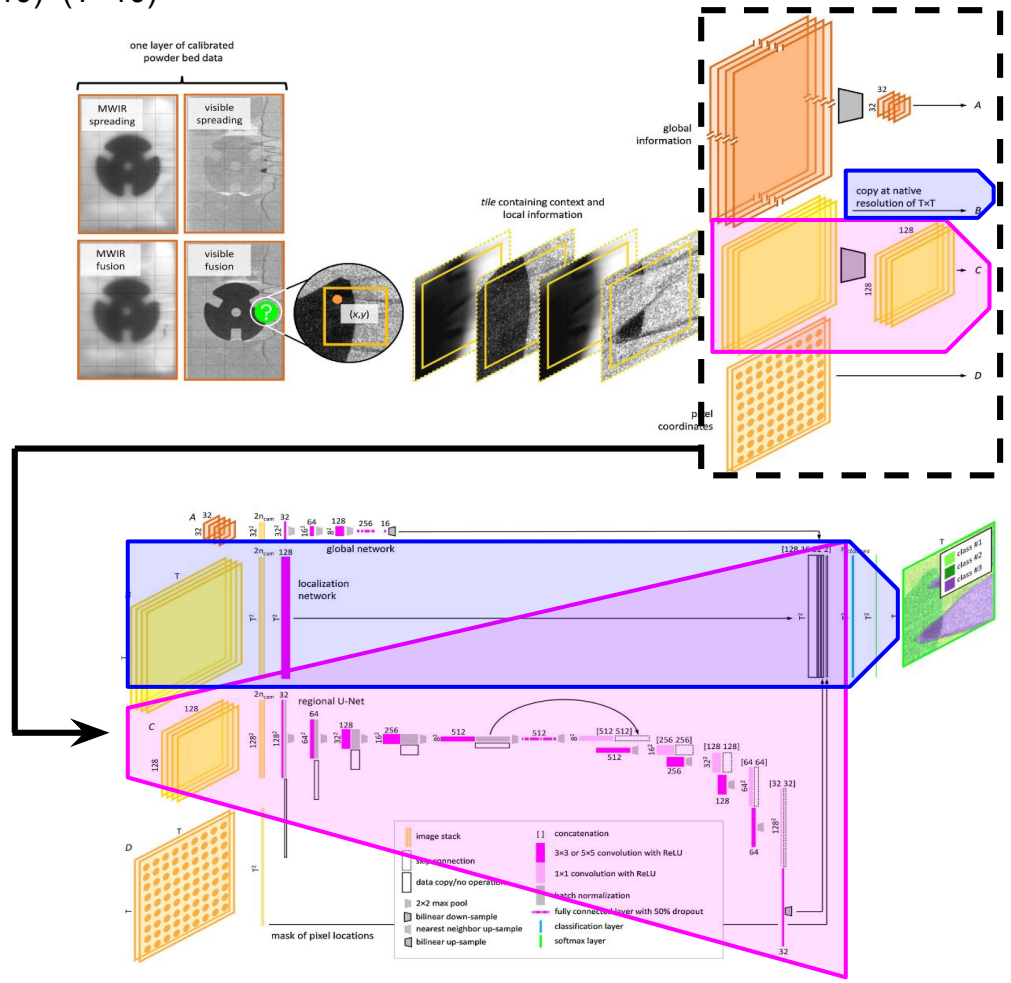


❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Construction of the Input Layer & DSCNN architecture & Pixel-wise classification (Ch.3.3~5)

➢ 3-legs architecture (Cont')

- Localization Image Stack (B):
  - **Retain one tile copy at native resolution (T×T)**, pad by 5 pixels → (T+10)×(T+10)
  - Stack post-spreading + post-fusion images → '2n_cam' channels
  - Input: (T+5) × (T+5) × 2n_cam
  - Output: **T×T×128 feature map**
  - Preserves **fine-scale, highly local information**, such as:
    - Local texture variations, Tiny spatter particles, Sharp boundary transitions
    - Very small defects
- Regional Image Stack (C):
  - **Resize** the native tile to 128×128 using bilinear interpolation
  - Apply 2-pixel padding → 132 × 132 × 2n_cam
  - Input: 132 × 132 × 2n_cam
  - Output: **T × T × 32 feature map**
  - Captures **medium-scale (regional) morphology**, such as:
    - Part edge geometry, Swelling, Recoater streak, Incomplete spreading
- Pixel Coordinate Stack (D):
  - Normalize each pixel (x, y) ∈ [−1, 1] relative to the powder bed center
  - Form a 2-channel map
  - Encodes location-dependent priors, such as:
    - Incomplete spreading often occurs near one side of the bed
    - Some defects exhibit spatial bias due to machine mechanics

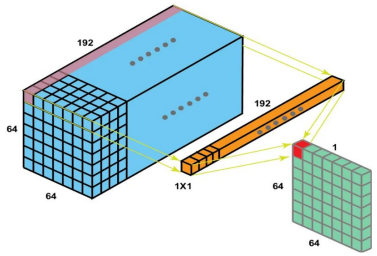


❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Construction of the Input Layer & DSCNN architecture & Pixel-wise classification (Ch.3.3~5)

➢ 3-legs architecture (Cont')

- After processing, all three legs output tensors with identical spatial resolution: T×T
  - They can be aligned perfectly and concatenated along the channel dimension, together with pixel coordinates **(T×T×178)**
  - Each pixel now has a 178-dimensional feature vector representing multi-scale context
- Pixel-Wise Classification Layer
  - Structure: '**in=178**' → '**out=n_classes=12**' via 1×1 Convolution
    - Output layers (stack A, B, C) of the three legs + pixel coordinates (stack D) are concatenated
    - Each pixel now has a 178-D feature vector, capturing local + regional + global + positional context
    - A **1×1 convolution (12)** converts the 178-D vector into **n_classes logits** per pixel
    - No ReLU & Softmax applied afterward: Outputs per-pixel pseudo-probabilities
    - This yields a per-pixel probability distribution across all anomaly classes

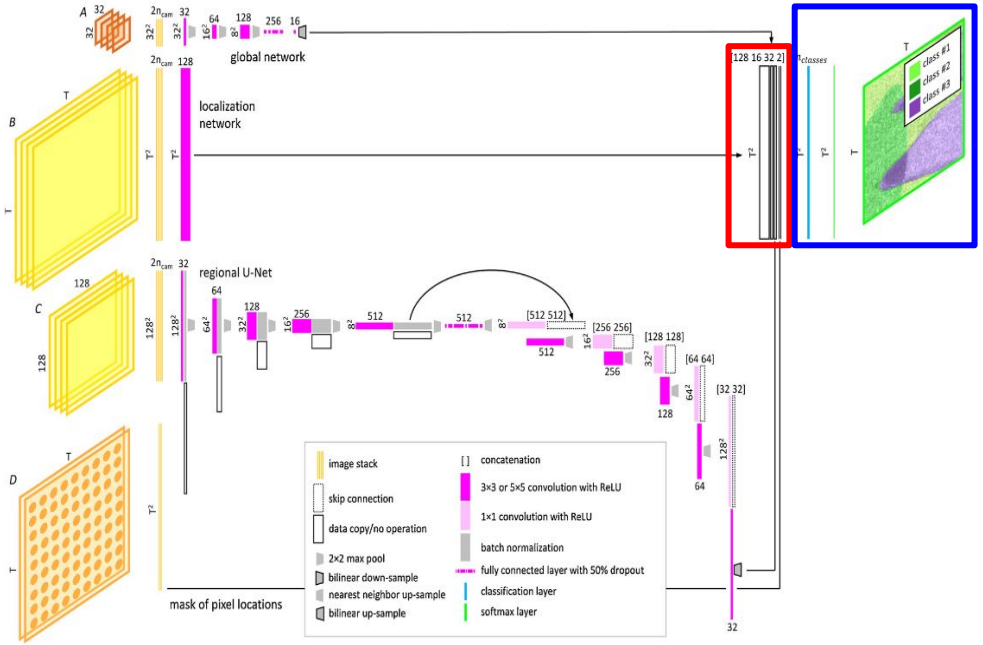


Final combination of multi-scale *features*. All units in pixels.

Layer	W_i	C_i	F	S	W_o	C_o
Concat	T	[128 16 32 2]			T	178
Classification	T	178	1	1	T	$n_{classes}$
Softmax	T	$n_{classes}$			T	$n_{classes}$

➢ The beauty of 3-legs architecture

- Reflect AM data characteristics
  - Defects are found **multi-scale** → need multi-scale encoder
  - Images are high resolution → tile strategy is essential
  - Machines have variable camera counts → dynamic channel handling
  - Location strongly affects defect likelihood → coordinate channels included
- Support robust transfer learning
  - Localization leg is fully convolutional → resolution-independent!
  - Tile size can be adjusted without breaking learned features
  - Feature extraction generalizes across machines



❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Training & Performance optimization (Ch.3.6~7)

➢ Challenges in powder-bed AM Data

- **Noisy & Inconsistent Manual Labels**
  - Pixel-wise labels are drawn manually → Potential Mis-labeling
  - Different builds, lighting, material batches, annotators → Inconsistent labeling
  - Label noise causes the model to learn contradictory patterns, degrading generalization across layers and machines
- **Extreme Class Imbalance**
  - Powder pixels dominate (40–80%), while rare defects (e.g., swelling, porosity, spatter, soot) comprise <0.1%

Table 2
Accounting of ground truth data used for training. The balance of the training data consists of unlabeled pixels.

	EOS M290	ConceptLaser M2	Renishaw AM250	Arcam Q10	ExOne M-Flex	ExOne Innovent
Number of Builds	8	12	1	1	10	10
Number of Layers	37	23	3	4	14	13
Number of Pixels (10^6)	33	90	8	7	36	132
Anomaly Class						
Powder	83.2 %	71.8 %	73.5 %	44.7 %	63.5 %	80.5 %
Part	3.1 %	12.8 %	25.2 %	41.1 %	17.6 %	13.1 %
Recoater Hopping	1.3 %	–	–	–	–	–
Recoater Streaking	0.7 %	0.2 %	–	–	0.6 %	0.5 %
Incomplete Spreading	5.2 %	3.0 %	–	–	4.0 %	5.1 %
Swelling	0.1 %	0.1 %	0.1 %	–	–	–
Spatter	–	–	1.2 %	–	–	–
Soot	–	0.2 %	–	–	–	–
Debris	2.7 %	0.4 %	–	–	3.1 %	–
Super-Elevation	1.2 %	0.4 %	–	–	–	–
Part Damage	0.0 %[a]	–	–	–	–	–
Porosity		–	–	0.2 %	–	–

[a] The *part damage* class is not directly learned, it is only predicted according the heuristics outlined in Section 3.8.

- The model could learn to always predict the majority class, achieving high accuracy but zero usefulness

  → **Need correction in loss-function to train the model!**

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Training & Performance optimization (Ch.3.6~7)

➢ Initialization Strategy

- AM images are extremely large → Unstable gradients at the beginning would cause exploding/vanishing gradients
  - CONV kernels ~ Normal(0, 0.05)
  - FC kernels ~ Normal(0, 0.004)
  - Stable early training improves convergence and prevents the model from falling into poor local minima

➢ Mini-Batch Construction (Tile-based; 15~50 tiles per layer)

- AM images are extremely large (100 times bigger than images used in typical CNN) and vary across machines
  → Hard to process a whole image layer as an input due to different dimension and GPU memory constraints
- Tile-based processing needed for Efficient GPU memory usage
  → Three-legs architecture was designed to support this by extracting multi-scale (local, regional, and global) information from each leg
  → Tiles preserve both local and regional context while keeping memory manageable
- Training rule: ≥10% labeled pixels required
  - Prevents batches dominated by unlabeled pixels, ensuring that gradients correspond to meaningful pixels
  - Unlabeled pixels masked out to prevents incorrect gradients when labels are missing

➢ Class Imbalance: Median Frequency Balancing Formula

- Median frequency balancing gives *more importance to rare defects* while stabilizing training
  - f_k = frequency of class k
  - Median({f}) = the median of all class frequencies

$$w_k = \frac{Median(\{f\})}{f_k} \Rightarrow E_k^{(weighted)} = w_k \cdot E_k$$

- DSCNN without balancing collapses to predicting powder only
- After balancing → even 0.1% defects could be learned
  - Rare class errors produce *large* loss → model learns them more strongly
  - Frequent class errors produce *small* loss → prevents majority-class domination

Suppose class frequencies are: {0.80, 0.15, 0.04, 0.01}.

Median ≈ 0.095

Powder: $0.095/0.80 \approx 0.12$ → small weight

Spatter: $0.095/0.04 \approx 2.38$ → larger weight

Porosity: $0.095/0.01 = 9.5$ → very large weight

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Training & Performance optimization (Ch.3.6~7)

➢ Hard Bootstrapping Loss

$$E_k = E(\{q\},\{t\}) = \sum_{k=1}^{n_{\text{classes}}} \left(\lambda t_k + (1-\lambda) z_k\right) \log_e(q_k + \varepsilon)$$

- Equation-Level Explanation
  - q_k: model's predicted pseudo-probability for class 'k' at a given pixel (softmax output)
  - $\{t_k\}$: one-hot human label. If the human says "class j", then t_j=1, others 0
  - $\{z_k\}$: one-hot model prediction, obtained by argmax on $\{q_k\}$. If model's top class is r, then z_r=1, others 0
  - $\lambda \in [0,1]$: trust coefficient. Controls how much we trust human vs. model
  - $\varepsilon = 10^{-8}$: tiny constant added for numerical stability when $q_k \to 0$

True probability distribution (one-shot)

$$H(p,q) = -\sum_{x \in classes} p(x) log\ q(x)$$

Your model's predicted probability distribution

- If λ=1: we trust the human completely → standard cross-entropy
- If λ<1: we partially trust the model's own prediction
  - Standard cross-entropy uses only t_k: $-\sum_k t_k \log q_k$
- Hard bootstrapping replaces t_k by blended target: $\tilde{t}_k = \lambda t_k + (1-\lambda) z_k$
  - **Case 1:** Human label is correct, model also agrees
    - Suppose 3 classes, human says class 2: t = (0,1,0)
    - Model predicts q = (0.1, 0.8, 0.1) → argmax is class 2 → z = (0,1,0)
    - Then, blended target: t = λ(0,1,0) + (1−λ)(0,1,0) = (0,1,0)
  - **Case 2:** Human label is wrong, model is confidently different
    - Human still says class 2: t=(0,1,0), but true class is actually 3
    - After some training, model predicts q = (0.1,0.1,0.8) → argmax is class 3 → z = (0,0,1)
    - Then, blended target: t = 0.7(0,1,0) + 0.3(0,1,0) = (0, 0.7, 0.3)
    - When the model is consistently confident against the human label, it gradually "pulls" the target away from the (possibly wrong) annotation
  - λ must be chosen moderately (e.g., 0.6~0.8) to be skeptical but not fully dismissive of human labels
    - If λ is too close to zero, model almost ignores human labels
- HB allows DSCNN to gradually "correct" inconsistent human labels using the pattern it learns from large volumes of data!

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Heuristics and Final Segmentation (Ch.3.8)

➢ This is a post-prediction calibration process

- The DSCNN produces pixel-wise predictions, but some AM anomalies cannot occur in certain locations
- Heuristic rules use domain knowledge (CAD geometry) to correct these impossible predictions
  - Debris on top of part → switch to part damage
    - Debris on top of part surface indicates the part surface was disrupted, meaning actual damage
  - If the DSCNN predicts “part / swelling” far away from expected part regions (>1100 μm), switch to “debris”
    - These anomalies *cannot exist* without fused material; if they appear in powder-only regions, the prediction must be wrong
- The authors considered adding CAD geometry as an input channel but rejected it
  - The model might overfit and rely on CAD rather than image evidence

❖ Transfer Learning (Ch.3.9)

➢ Each machine has different labeled class types and the amount of them is also different

➢ However, we can use learned feature representations from machine to machine

- Use machine #1’s abundant labeled data to help train a model for machine #2, which has limited labeled data
  - Step 1: Train DSCNN on machine-type #1 with random initialization
  - Step 2: Initialize a new DSCNN for machine-type #2 using learned parameters from machine #1
  - Only the final classification layer is randomly initialized because class sets differ
- Since DSCNN could learn class prediction from an abundant data, we can start from there to learn other less richer one!
  - Assumption: all these class types are distributed on the same manifold from a bigger perspective
- Tile resizing always occurs → small-scale details may be lost for high-resolution differences
  - Theoretically, transferring from 1 MP to >30 MP becomes very difficult

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ Results (Ch.4~5)

Anomaly Classification	EOS M290 MsCNN	EOS M290 DSCNN
Powder	−90.3 (2.5)	99.5 (1.6) 99.0 (1.4)
Part	- -	97.1 (9.2) 83.5 (17.0)
Recoater Hopping	−92.9 (90.9)	82.9 (10.0) 70.3 (35.1)
Recoater Streaking	−49.7 (74.0)	57.9 (8.8) 31.5 (41.2)
Incomplete Spreading	−74.1 (24.5)	92.4 (2.8) 95.7 (5.5)
Swelling	–	75.3 (29.5) 68.5 (57.0)
Spatter	–	–
Soot	–	- -
Debris	−82.7 (71.5)	72.1 (14.5) 67.3 (28.1)
Super-Elevation	−92.0 (54.4)	86.4 (5.0) 81.4 (5.2)
Part Damage	−90.5 (63.3)	−53.9 (49.7)
Porosity	–	–

➢ DSCNN shows better performance than MSCNN for pixel-wise classification

- For recoater hopping (1.3%) in EOS M290,
  - MsCNN true-positive rate (TPR) drops from 92.9% to 70.3% when moving to DSCNN, because MsCNN labels all pixels in a patch as positive if any anomaly exists inside
  - However, the false-positive rate (FPR) improves dramatically from 90.9% to 35.1%
    → This means DSCNN is much more precise spatially
- For large-scale incomplete spreading (5.2%), both TPR and FPR improve
  - TPR from 74.1% → 95.1%, FPR from 24.5% → 5.5%
  - For large-scale incomplete spreading (5.2%), both TPR and FPR improve
- DSCNN has much fewer mis-classification than MSCNN, with more accurate classification for large-scale anomalies

➢ Inter-Machine Transfer Learning improves FPR than non-transfer (initially & randomly trained) model

- Study case: Training a ConceptLaser M2 DSCNN with and without transfer from an ExOne Innovent DSCNN
  - After 10,000 mini-batches, the testing loss for the transfer-learned DSCNN is only 16% of the loss for the randomly initialized DSCNN
    → Transfer learning clearly speeds up learning and reaches a much better point in the same time
  - Class-wise average FPR is *better* (lower) for the transfer-learned model by 6.0% and 5.5%
    - Class-wise average TPR is *worse* for the transfer model by 7.6% and 5.4%, respectively
    - Early in training, a non-transfer model tends to over-segment anomalies
      → This inflates TPR but also massively inflates FPR which is bad!

	Without Transfer Learning	With Transfer Learning
Training Loss (no units)	1.35	0.22
Validation True Positive Rate	77.3 %	69.7 %
Validation False Positive Rate	44.9 %	38.9 %
Testing True Positive Rate	84.0 %	78.6 %
Testing False Positive Rate	38.0 %	32.5 %

  - **Transfer-learned model starts from a better structured representation → lower FPR but slightly lower TPR in early stages**
  - **As training continues, we can expect that TPR of the transfer-learned model to catch up/improve TPR while FPR stays low**
  - Loss/FPR-based comparison: Transfer-learned model is clearly beneficial, but simple TPR can mislead in this imbalanced & small-object setting

❑ Literature Review: Layer-wise anomaly detection and classification for powder bed AM

❖ **Results** (Ch.4~5)

➢ Skeptical (Hard-Bootstrapping) Loss Effect

- Ground-truth labels have inherent ambiguity: Class boundaries are not clearly defined, even for experts
  - Ambiguity exists between visually similar classes (debris vs soot, recoater streaking vs powder, etc.)
  - Human annotators make mistakes, especially in such borderline cases
  - Most mis-classifications occur near the boundaries
- Validation performance is higher for all classes when using skeptical loss (λ=0.9) than standard CE
  - Particularly large gains in TPR are reported for classes with subjective boundaries: (e.g., recoater streaking, incomplete spreading, swelling, soot, super-elevation)
  - Testing performance is nearly identical between standard and skeptical loss
    - The authors hypothesize that the validation dataset has richer variety than the test set

Anomaly Classification	ConceptLaser M2 standard	ConceptLaser M2 skeptical
Powder	99.0 (1.2) 98.5 (1.1)	99.3 (0.4) 98.6 (1.2)
Part	98.3 (3.8) 96.3 (6.6)	98.5 (2.6) 96.1 (6.2)
Recoater Hopping	–	–
Recoater Streaking	39.4 (38.6) 30.2 (80.7)	73.8 (20.4) 31.5 (79.0)
Incomplete Spreading	71.6 (5.1) 92.4 (3.9)	99.4 (1.1) 92.6 (4.0)
Swelling	65.7 (26.1) 60.4 (31.2)	79.2 (15.0) 58.9 (31.5)
Spatter	–	–
Soot	17.5 (74.9) 63.9 (39.6)	73.5 (31.0) 64.2 (38.5)
Debris	78.6 (20.6) 67.5 (29.1)	89.1 (12.2) 63.8 (27.4)
Super-Elevation	72.1 (64.5) 93.0 (17.5)	98.4 (11.8) 92.3 (11.7)
Part Damage	–	–
Porosity	–	–

➢ Qualitative Evaluation (EOS M290)

- Comparison of (a) MsCNN and (b) DSCNN predictions for a layer of EOS M290 powder bed imaging data overlaid on top the post-spreading image
- DSCNN shows better performance in a qualitative manner
  - Vastly improved localization of incomplete spreading, debris, recoater streaking, and recoater hopping
  - Less confusion between incomplete spreading and debris classes
  - DSCNN detects classes such as part and swelling that MsCNN did not handle