A Strategic Approach to Pixel-Level Radiograph Analysis: Developing a Machine Learning Proof of Concept for Source Identification and Energy Estimation

1. Introduction: Framing the Challenge and Opportunity

This report outlines a strategic plan for developing a Machine Learning (ML) Proof of Concept (PoC) focused on the analysis of simulated radiograph images. The primary objectives are to perform pixel-level semantic segmentation of these 256x256 grayscale images to identify neutron sources, gamma sources, noise, and potentially other relevant classes (e.g., different material interactions, scatter). Concurrently, the PoC aims to estimate the energy associated with these identified pixels or regions. This document is intended to guide the ML team, particularly those new to the field, through the process of understanding the possibilities, familiarizing themselves with common methods, and aligning with desired project outcomes.

A significant advantage for this project is the availability of high-fidelity physics simulation data accompanied by the corresponding simulation parameters. This controlled dataset provides a unique opportunity to understand model behavior and potentially integrate physics-based knowledge, either directly or indirectly, into the ML models.[1] This is particularly beneficial when addressing the primary constraint of this PoC: a limited amount of labeled data. The scarcity of labeled examples will fundamentally shape the strategy, necessitating the exploration of techniques beyond standard supervised learning, such as data augmentation, transfer learning, semi-supervised learning, and potentially active learning to maximize the utility of the available data.[3]

It is important to frame this PoC as an exploratory endeavor rather than the development of a production-ready system. The goals are to: (a) determine what is technically achievable with the current dataset and ML techniques; (b) provide the team with hands-on experience and familiarity with the ML workflow; and (c) generate insights that can inform the Product Manager (PM) about potential outcomes, limitations, and future development directions.

While the current project utilizes simulated data, it is worth noting that models trained exclusively on simulations can sometimes face challenges when applied to real-world data, a phenomenon often referred to as the "simulation-to-reality gap".[7] Although the high-fidelity nature of the simulations used here may mitigate this issue, it remains a pertinent consideration for any future aspirations to deploy models on actual experimental radiographs. The PoC can, however, establish a valuable baseline understanding of feature importance derived from simulations, which can later be compared with features learned from real data, should it become available.

Furthermore, the simulation parameters themselves, while a rich source of information, must be handled thoughtfully. If used directly as input features, there's a potential for the model to develop an over-reliance on them. Should real-world scenarios involve parameter ranges or combinations not adequately covered by the simulations, the model's performance could be compromised.[1] Therefore, a preliminary step involves understanding the distribution and comprehensiveness of these simulation parameters to ensure they adequately represent the phenomena of interest.

2. Fundamentals for the ML Team: Core Concepts and Terminology

Embarking on a machine learning project requires a foundational understanding of its core principles and the specific domain of application. This section aims to provide the team with essential knowledge regarding ML in image analysis, the characteristics of the radiograph data, and fundamental ML terminology.

2.1. Introduction to Machine Learning in Image Analysis

Machine learning, particularly deep learning, has significantly advanced the capabilities of image-based analysis.[8] At its core, ML involves training computer systems to learn patterns from data to make predictions or decisions without being explicitly programmed for each specific task. In scientific imaging, including applications similar to medical imaging, ML models can decipher complex patterns, identify subtle features, and perform quantitative analysis at a scale and speed often unachievable by manual methods.[9]

2.2. Supervised Learning and Semantic Segmentation

This PoC will primarily utilize **supervised learning**, a paradigm where the model learns from data that includes both input examples (the radiograph images) and their corresponding known outputs (pixel-level labels and energy values).[11] The specific task is **semantic segmentation**, which involves assigning a class label (e.g., "gamma," "neutron," "noise") to every individual pixel in an image.[12] This is distinct from image classification, which assigns a single label to an entire image, or object detection, which identifies objects with bounding boxes. Semantic segmentation provides a dense, pixel-wise understanding of the image content, which is crucial for localizing sources and characterizing their properties.

2.3. Understanding Radiograph Data: Neutron vs. Gamma Signatures

The radiograph images in this project depict interactions of neutron and gamma radiation with matter. Understanding the fundamental differences in these interactions is key to interpreting the images and guiding the ML model development.

Neutrons and gamma rays interact with materials in distinct ways:

- **Neutrons** primarily interact with the atomic nuclei. This makes them particularly sensitive to light elements, especially those rich in hydrogen (e.g., plastics, water, organic compounds), and certain isotopes with high neutron absorption cross-sections, such as gadolinium (often used as a contrast agent).[15] Neutrons can penetrate dense materials like metals more easily than X-rays or gamma rays but are strongly attenuated by hydrogenous materials.
- **Gamma rays** (and X-rays, which are electromagnetically similar) interact predominantly with the electrons of an atom. Their attenuation is largely dependent on the material's density and atomic number (Z).[10] Dense, high-Z materials (like lead or steel) strongly absorb gamma rays, while lighter, low-Z materials are more transparent.

These differing interaction mechanisms lead to unique visual signatures in radiographs:

- **Neutron Radiographs:** Areas containing hydrogen-rich substances (e.g., plastics, water, oils, explosives like gunpowder, or even food contaminants like ketchup or butter as demonstrated in fuel injector imaging) will appear opaque or show high contrast because neutrons are significantly scattered or absorbed by hydrogen.[15] Conversely, many dense metallic components might appear relatively transparent to neutrons. The use of gadolinium tagging, for example, makes porous ceramic fragments within a turbine blade highly visible in neutron images due to gadolinium's high neutron cross-section.[15]
- **Gamma/X-ray Radiographs:** Dense materials (e.g., metals) will appear opaque (often white or bright in typical displays), effectively blocking the radiation. Lighter materials, including plastics and organic substances, will appear more transparent (darker), allowing more radiation to pass through.[10] For instance, in a firearm, the metallic components would be prominent in an X-ray, while the gunpowder might be invisible; a neutron image, however, could clearly show the gunpowder.[15]

The ML model's task will be to learn these distinct attenuation patterns to differentiate between neutron-induced signatures, gamma-induced signatures, and background noise. The "toy radiographs" are expected to simulate these varying visual characteristics. The patterns are not arbitrary; they are direct visual manifestations of underlying physical processes. A basic understanding of this physics can aid the team in interpreting model behavior and potentially in guiding feature engineering if simulation parameters are used directly as inputs.

Table 1: Comparison of Neutron and Gamma Ray Signatures in Radiographs

Feature	Neutron Radiation	Gamma/X-ray Radiation
Primary Interaction Mechanism	Nuclear scattering and absorption (interaction with atomic nuclei) [16]	Photoelectric effect, Compton scattering, pair production (interaction with atomic electrons) [10]
Sensitivity to Elements	High sensitivity to light elements (esp. Hydrogen), specific isotopes (e.g., Boron, Cadmium, Gadolinium) [15]	Sensitivity increases with atomic number (Z) and density [10]
Behavior with High-Z Materials	Generally good penetration (e.g., through lead, steel) [15]	Strong attenuation (materials appear opaque) [10]
Behavior with Low-Z/Hydro. Mat.	Strong attenuation (materials like plastics, water, organics appear opaque) [15]	Generally good penetration (materials appear transparent) [10]
Typical Appearance of Sources	Regions of high opacity where hydrogenous materials or neutron-absorbing materials are present.	Regions of high opacity where dense materials are present.
Example Applications from Research	Imaging munitions (gunpowder), detecting ceramic fragments via Gd-tagging, visualizing water uptake in roots, identifying contaminants in fuel injectors (ketchup, butter) [15]	Standard medical imaging (bones), industrial inspection for dense material defects, imaging metallic structures [10]

2.4. Neural Network Basics

For those new to ML, neural networks can be thought of as systems inspired by the human brain, composed of interconnected nodes or "neurons" organized in layers. They learn by adjusting the strength of these connections.

- **Layers:**
  - **Input Layer:** Receives the initial data (e.g., the pixel values of a 256x256 grayscale image).
  - **Hidden Layers:** Perform computations and feature extraction. In image analysis, **Convolutional Neural Networks (CNNs)** utilize specialized convolutional layers that apply filters to input images, effectively learning to detect patterns like edges, textures, and more complex shapes hierarchically.[9] The "depth" of a neural network refers to the number of hidden layers.
  - **Output Layer:** Produces the final prediction (e.g., a class label for each pixel, an energy value).
- **Activation Functions (e.g., ReLU):** After each neuron's computation, an activation function is applied. Functions like ReLU (Rectified Linear Unit) introduce non-linearity into the model.[19] This is crucial because real-world data and the patterns within it are rarely simple and linear; non-linearity allows the network to learn much more complex relationships.
- **Loss Functions:** A loss function quantifies how "wrong" the model's predictions are compared to the actual ground truth values during training.[20] The goal of training is to minimize this loss.
  - For **Semantic Segmentation** (a pixel-level classification task):
    - **Cross-Entropy Loss:** A standard loss function for classification tasks. It measures the difference between the predicted probability distribution over classes and the true distribution.[12]
    - **Dice Loss:** Particularly useful for segmentation tasks, especially with imbalanced classes (e.g., when source regions are small compared to the background). It directly measures the overlap between predicted and true segmentation masks.[13]
  - For **Energy Prediction** (a regression task):
    - **Mean Squared Error (MSE):** Calculates the average of the squared differences between predicted and true energy values. It heavily penalizes large errors.[20]
    - **Mean Absolute Error (MAE):** Calculates the average of the absolute differences. It is less sensitive to outliers than MSE and is interpretable in the original units of energy.[20]
- **Optimizers (e.g., SGD, Adam):** Optimizers are algorithms that adjust the internal parameters (called weights and biases) of the neural network in response to the loss calculated.[22] They essentially guide the "learning" process by iteratively

updating the weights to try and minimize the loss function. Adam is a popular and often effective optimizer that adapts the learning rate for each parameter, generally requiring less manual tuning than simpler optimizers like Stochastic Gradient Descent (SGD).[11]

2.5. Model Training and Evaluation

- **Training:** This is the iterative process where the model is repeatedly shown examples from the training dataset. In each iteration, the model makes predictions, the loss function calculates the error, and the optimizer updates the model's weights to reduce this error.
- **Overfitting vs. Underfitting:** These are common challenges in model training.[11]
    - **Overfitting:** Occurs when the model learns the training data too well, including its noise and specific idiosyncrasies. Such a model performs excellently on the data it was trained on but fails to generalize to new, unseen data.
    - **Underfitting:** Occurs when the model is too simple to capture the underlying patterns in the data. It performs poorly on both the training data and unseen data.
- **Validation and Testing:** To develop a reliable model and accurately assess its performance, the data is typically split into three sets:
    - **Training Set:** Used to train the model.
    - **Validation Set:** Used during training to tune hyperparameters (like learning rate or model architecture choices) and for early stopping (stopping training when performance on the validation set starts to degrade, to prevent overfitting).[24]
    - **Test Set:** Held out until the very end of development. It is used only once to provide an unbiased evaluation of the final model's performance on unseen data.

A clear and consistent understanding of this terminology is foundational. It forms the common language for discussing progress, challenges, and strategies within the team, ultimately contributing to the PoC's success by ensuring everyone can meaningfully participate in the development and interpretation process.

3. Strategy for Proof of Concept (PoC) Development: An Iterative Approach

The development of a Machine Learning Proof of Concept, especially with a team new to the field, benefits greatly from an iterative methodology. This approach allows for incremental progress, learning from each cycle, and adapting the strategy based on empirical results.

3.1. The Iterative ML Development Cycle for a PoC

Machine learning development is rarely a linear process; it is inherently iterative.[26] Each iteration typically involves a cycle of:

1. **Requirements & Planning:** Defining the goals for the current iteration.
2. **Analysis & Design:** Analyzing available data, selecting appropriate models and techniques.
3. **Implementation:** Coding the data processing, model training, and evaluation logic.
4. **Testing & Evaluation:** Running experiments and assessing performance against defined metrics.
5. **Review & Refinement:** Interpreting results, identifying shortcomings, and planning the next iteration.

For this PoC, an initial iteration might focus on a simplified version of the problem, such as segmenting only one type of source (e.g., neutron) with a basic model architecture. Subsequent iterations can then build upon this by adding more classes, incorporating energy prediction, and exploring more advanced techniques. This iterative experimentation is a core tenet of MLOps (Machine Learning Operations) principles.[27] Active learning, if employed later, also follows a similar iterative cycle of model training, querying, labeling, and retraining.[6]

3.2. Data Understanding and Preprocessing

Before any model training, a thorough understanding and careful preprocessing of the data are essential.

- **Initial Data Exploration:** The first step involves visually inspecting a sample of the radiograph images alongside their corresponding simulation parameters. This helps in building intuition about the data: How do neutron and gamma sources typically appear? Are there obvious visual distinctions? What do "noise" regions look like? Are there any apparent correlations between simulation parameters and image features?
- **Handling 256x256 Grayscale Images:** The input images are 256x256 grayscale. Standard preprocessing steps include:
  - **Normalization:** Pixel intensity values (typically ranging from 0 to 255 for uint8 grayscale images [28]) should be scaled to a smaller, consistent range, such as or [-1, 1]. This is crucial for the stable training of neural networks, as it ensures

that all input features have a similar distribution and magnitude, preventing issues with gradient updates.[29]

- **Resizing/Cropping:** It's important to confirm that all images strictly adhere to the 256x256 dimension. If variations exist (though unlikely for simulated data), a consistent resizing or cropping strategy must be applied.[28]
- **Denoising:** Simulated data can sometimes be "too perfect" or, conversely, might include specific types of simulated noise. If the simulations are very clean, adding a small amount of realistic noise (a form of data augmentation, see Section 5) might help the model generalize better to potentially noisier real-world data, if that is an eventual goal. If the simulations inherently include noise that is not representative of the target phenomena, denoising techniques could be explored.[29] Methods for unmixing signals and denoising have been applied to X-ray analysis.[32]
- **Contrast Enhancement:** Techniques like Histogram Equalization or Contrast Limited Adaptive Histogram Equalization (CLAHE) can enhance the visibility of subtle features in radiographs by redistributing pixel intensities.[29] This could be beneficial for distinguishing faint sources or fine details. However, these methods should be applied judiciously, as they can also amplify existing noise. The impact of contrast enhancement should be experimentally validated. The way images are normalized, potentially denoised, or contrast-enhanced constitutes a form of feature engineering for image data. These steps can significantly influence which features the model learns, especially for the subtle signatures expected in radiographs. The simulation parameters themselves could potentially guide or personalize these preprocessing choices. For instance, if a simulation parameter indicates a very low-intensity source, a contrast enhancement technique tailored for low-signal scenarios might be more effective than a generic one.

- **Integrating Simulation Parameters as Input Features:** The availability of simulation parameters is a key asset. Several strategies can be explored to leverage this information:
  - **Direct Input:** Flattened simulation parameters can be concatenated with image features at a suitable point in the neural network architecture. For example, they could be injected into the bottleneck of a U-Net or combined with features just before the final classification/regression layers.
  - **Conditional Processing:** Parameters could be used to conditionally alter parts of the model or the data flow.
  - **Physics-Informed ML (PIML) Principles:** While implementing full Physics-Informed Neural Networks (PINNs) that incorporate differential equations into the loss function [1] might be too advanced for an initial PoC with

a novice team, the underlying principle of using physics knowledge is valuable. For example, if a simulation parameter defines an expected energy range for a source, the loss function for the energy prediction task could be weighted or constrained based on this information. Incorporating physics knowledge can act as a form of regularization, guiding the model towards physically plausible solutions and potentially improving learning efficiency, especially with limited labeled data.[1] While some studies show ML applied to radiograph analysis [33], they don't explicitly detail using simulation parameters as direct input features in the provided contexts. Non-negative matrix factorization (NMF) has been used for unmixing signals in X-ray data, which could be relevant if sources overlap and parameters define their characteristics.[32]

3.3. Establishing a Strong Baseline Model

A crucial step in any ML project is to establish a strong baseline model. This provides a benchmark against which all subsequent improvements and more complex approaches can be measured.[35]

- **Start Simple:** The initial baseline should be relatively uncomplicated. For segmentation, a standard U-Net architecture [12] is an excellent choice. For energy prediction, a simple regression head attached to the U-Net or a separate small network operating on segmented regions can be used.
- **Train on Limited Labeled Data First:** Train this simple model using only the small, existing labeled dataset without employing advanced techniques like extensive augmentation or transfer learning initially.
- **Key Baseline Considerations:**
  - **Optimizer:** Adam is often a good default choice.[23]
  - **Learning Rate Schedule:** A decaying learning rate schedule (e.g., cosine decay or step decay) can help in finer convergence.[35]
  - **Batch Size:** Choose an appropriate batch size that fits into memory and provides stable gradient estimates.
  - **Pre-trained Weights:** If transfer learning is planned for later iterations, the baseline might initially use random weight initialization to purely assess learning from the project's own data. Alternatively, a baseline *with* standard pre-training can also be established early.

The performance of this baseline model will be highly informative. If even a simple model shows some capability in distinguishing sources or predicting energy, it's an encouraging sign. If it performs poorly, it underscores the severity of the data scarcity problem and highlights the critical need for the techniques discussed in Section 5.

This iterative feedback is central to the PoC process.[26]

3.4. MLOps Principles for a PoC (Introduction)

While a full-scale MLOps pipeline is beyond the scope of a PoC, adopting basic MLOps discipline from the outset is vital for efficiency, reproducibility, and learning, especially for a new team.[27]

- **Experiment Tracking:** Systematically log every experiment. This includes details such as:
  - The specific version of the dataset used (raw, preprocessed, augmented).
  - The model architecture and its configuration.
  - Hyperparameters used for training (e.g., learning rate, batch size, number of epochs).
  - Evaluation metrics achieved on the validation and test sets. Tools like MLflow [36] or TensorBoard can automate some of this, but even well-organized spreadsheets or text files are invaluable.
- **Data Versioning:** Keep track of different versions of the datasets as they evolve through preprocessing and augmentation.
- **Model Versioning:** Save trained model weights and their corresponding configurations. This allows for revisiting previous models and reliably reproducing results.

Adopting these practices early on, even in a simplified manner, builds good habits within the team. It prevents wasted effort from re-running forgotten experiments and ensures that insights gained from each iteration are well-documented and can effectively inform subsequent steps.[27] The ability to "fail fast and learn quick" and ensure reproducibility are key benefits.[36]

4. Core ML Task: Pixel-Level Segmentation and Energy Estimation

The central tasks for this PoC are to identify different radiation sources at a pixel level (semantic segmentation) and to estimate the energy associated with these sources (regression). This section delves into the primary model architectures and approaches for these tasks.

4.1. Semantic Segmentation for Source Identification

The goal is to classify each pixel in the 256x256 grayscale radiograph as belonging to "neutron source," "gamma source," "noise," or other potential classes.

- Deep Dive: U-Net Architecture
  The U-Net architecture is exceptionally well-suited for biomedical and scientific

image segmentation tasks, including the one at hand.12 Its design effectively balances feature extraction at multiple scales with precise localization.

- **Encoder-Decoder Structure:** The U-Net consists of two main paths [12]:
    - **Contracting Path (Encoder):** This path progressively downsamples the input image through a series of convolutional and pooling layers. Each downsampling step increases the receptive field of the neurons, allowing the network to capture broader contextual information and learn more abstract features.
    - **Expansive Path (Decoder):** This path symmetrically upsamples the feature maps from the encoder's bottleneck. Through transposed convolutions (or upsampling followed by convolutions), it gradually reconstructs a full-resolution segmentation map where each pixel is assigned a class label.
- **Skip Connections:** A hallmark of the U-Net is its use of skip connections.[12] These connections concatenate feature maps from the contracting path directly to the corresponding layers in the expansive path. This is crucial because it allows the decoder to reuse high-resolution spatial information from earlier layers that might have been lost during the downsampling in the encoder. For images like radiographs, where subtle details (e.g., the precise boundary of a source, fine textures indicating material type [15]) are important for accurate classification, these skip connections are vital for producing sharp and accurate segmentation masks.[12]
- **Suitability for 256x256 Grayscale Images:** U-Net was originally developed for biomedical images, which are often grayscale and of moderate size, making it a natural fit for the 256x256 radiographs in this project.[12]
- **Final Layer:** The network typically culminates in a 1x1 convolutional layer followed by a softmax activation function. The 1x1 convolution maps the multi-channel feature maps from the final decoder stage to a number of channels equal to the number of target classes (e.g., neutron, gamma, noise). The softmax function then computes the probability of each pixel belonging to each class.[12]

* Alternative/Advanced Segmentation Models (Brief Mention):
  While U-Net is an excellent starting point for a PoC with a novice team due to its widespread success, relative simplicity, and effectiveness 38, other architectures could be considered in later iterations or if U-Net encounters specific limitations:
    - **DeepLabV3:** Known for its use of atrous (dilated) convolutions and Atrous Spatial Pyramid Pooling (ASPP), which allow it to capture multi-scale contextual information effectively without significantly increasing parameters.[4] This could be beneficial if distinguishing sources requires understanding

broader context within the radiograph.
  - **SegNet:** Another encoder-decoder architecture that uses pooling indices from the encoder to perform non-linear upsampling in the decoder, aiming to preserve boundary details.[4]

For the initial PoC, focusing on mastering U-Net is recommended.

4.2. Energy Prediction (Regression Task)

Alongside identifying the type of source, the PoC aims to predict the energy associated with these sources. This is a regression task, as energy is a continuous value.

- **Approach:** The ground truth for this task will likely come from the simulation parameters that define the initial energy of the simulated sources or the energy deposited. It is crucial to clarify precisely what "energy" the model should predict: is it the energy of the incident particle, the energy deposited in the pixel, or some other derived quantity? This definition will directly influence how the ground truth is formulated and how the model is designed.
- **Integration with Segmentation:**
  - **Pixel-wise Regression:** One approach is to predict an energy value for every pixel classified as a neutron or gamma source. This could be achieved by adding a regression head to the U-Net decoder. Instead of (or in addition to) outputting class probabilities, this head would output a continuous energy value per pixel.
  - **Region-based Regression:** A potentially simpler initial approach would be to first segment the sources. Then, for each identified source region (a collection of pixels), features from that region (e.g., average pixel intensity from the U-Net's feature maps, size, shape) could be pooled and fed into a separate, smaller regression model (e.g., a few fully connected layers) to predict a single energy value for the entire region.

4.3. Multi-Task Learning (MTL) for Combined Prediction

Given that source identification (segmentation) and energy estimation (regression) are inherently related (a specific source type will have an associated energy), a Multi-Task Learning (MTL) approach is a natural consideration.[18]

- **Concept:** MTL involves training a single model to perform multiple tasks simultaneously. The underlying idea is that the tasks can benefit from shared representations learned by a common part of the network.[18]
- **Architectural Considerations:**
  - A common MTL setup features a **shared backbone** (e.g., the U-Net encoder)

that learns features relevant to all tasks. This backbone is then followed by **task-specific heads** or decoders.[18] For this PoC, this would mean a U-Net encoder, a segmentation decoder head, and an energy regression head.
    - More sophisticated MTL architectures, like KMNet [40] for remote sensing or MT-CP [41] for dense visual tasks, explore advanced concepts such as multi-scale feature fusion and explicit cross-task coherence mechanisms. While these specific architectures might be too complex for an initial PoC, the fundamental principles of shared feature learning and dedicated task outputs are directly applicable.
- **Loss Function:** The overall loss function in an MTL setup is typically a weighted sum of the individual task losses. For instance: Ltotal=w1·Lsegmentation+w2·Lenergy_regression. Balancing these task losses (i.e., choosing appropriate weights w1 and w2) is critical. If one task's loss dominates, the model might prioritize that task to the detriment of others. Dynamic loss prioritization schemes exist but add complexity.[41]
- **Benefits:**
    - **Improved Performance and Generalization:** Tasks can act as implicit regularization for each other, leading to more robust and generalizable features.[18]
    - **Parameter Efficiency:** An MTL model typically has fewer parameters than training separate models for each task.
- **Challenges:**
    - **Task Balancing:** As mentioned, appropriately weighting the task losses can be challenging and often requires careful tuning.
    - **Negative Transfer:** In some cases, if tasks are too dissimilar or the architecture is not well-designed, one task might hinder the learning of another.

For a novice team, a staged approach might be most effective: first, develop a strong segmentation-only model. Second, develop an energy prediction model (perhaps operating on ground-truth or predicted segments). Finally, explore combining these into an MTL framework. This allows for incremental learning and complexity management.

Table 2: Overview of ML Models for Pixel-Level Tasks

Model/Approach	Primary Task(s)	Key Architectur	Common Loss	Pros for Radiograph	Cons/Challenges for

		al Features	Functions	Analysis	PoC
U-Net [12]	Semantic Segmentation	Encoder-decoder, Skip connections	Cross-Entropy, Dice Loss [13]	Excellent for precise localization of sources, handles limited data well with augmentation, good for grayscale images, skip connections preserve fine details.	May require tuning for optimal performance with specific radiograph characteristics.
U-Net with Regression Head	Semantic Segmentation & Pixel-wise Energy Reg.	U-Net base, additional regression output head (e.g., linear activation) from decoder features	Combined Segmentation Loss (Dice/CE) + Regression Loss (MSE/MAE) [20]	Predicts energy at a fine-grained level.	Increased model complexity; balancing segmentation and regression losses can be tricky. Energy definition per pixel needs clarity.
Basic MTL U-Net [18]	Semantic Segmentation & Energy Regression	Shared U-Net Encoder, Separate Decoder (Segmentation) & Regression Head	Weighted sum of Segmentation Loss + Regression Loss [41]	Potential for improved generalization and parameter efficiency by learning shared representations.	Requires careful balancing of task losses; risk of negative transfer if tasks are not well-aligned or architecture isn't optimal. Adds

					hyperparameters.
DeepLabV3 (Advanced) [4]	Semantic Segmentation	Atrous convolutions, ASPP module	Cross-Entropy, Dice Loss	Strong context aggregation, good for objects of varying scales.	More complex than U-Net, might be overkill for initial PoC with a novice team.

5. Addressing Data Scarcity: Techniques for Limited Labeled Data

The limited availability of labeled data is a central challenge in this PoC. Relying solely on a small labeled dataset for supervised learning is likely to result in models that overfit (i.e., perform well on the training data but poorly on unseen data) and lack generalization capabilities.[24] This section outlines several techniques to mitigate this issue by maximizing the information extracted from both the limited labeled data and the potentially larger pool of unlabeled simulation data.

5.1. Data Augmentation

Data augmentation involves artificially expanding the size and diversity of the *labeled* training dataset by creating modified versions of the existing labeled images and their corresponding masks/energy values.[3] This helps the model learn to be robust to various transformations and reduces overfitting.

- **Geometric Transformations:**
  - **Flipping:** Horizontal and/or vertical flips of the images and masks. For radiographs, horizontal flips are generally safe. Vertical flips might be applicable depending on the symmetry and nature of the imaged objects.[3]
  - **Rotation:** Rotating images and masks by small angles (e.g., ±5 to ±15 degrees) can simulate slight variations in object orientation. Excessive rotation should be avoided as it might create unrealistic scenarios for radiographs.[3] Some contexts, like X-ray imaging, may have specific constraints on rotations.[43]
  - **Cropping:** Randomly or centrally cropping patches from the images and corresponding masks. This forces the model to learn from partial views.[3]
  - **Scaling (Zooming):** Resizing images and masks to simulate objects appearing at different distances or sizes.[3]
  - **Translation (Shifting):** Shifting images and masks horizontally or vertically.[3]

- **Intensity and Noise-based Augmentations:**
    - **Brightness and Contrast Adjustments:** Randomly altering the brightness and contrast of the images can help the model generalize to different imaging conditions or exposure levels.[3]
    - **Noise Injection:** Adding random noise (e.g., Gaussian noise, salt-and-pepper noise) to the images simulates imperfections in detectors or variations in signal transmission.[3]
    - **Elastic Deformations:** Applying small, non-rigid deformations to the images and masks. This technique is particularly effective for medical and biological images as it simulates the natural variability and "squishiness" of tissues.[12]
- **Advanced Augmentation (Consider for later iterations):**
    - **Generative Adversarial Networks (GANs):** GANs can be trained to synthesize entirely new, realistic radiograph images and their corresponding segmentations.[3] This is a more complex approach but can be very powerful for significantly expanding the dataset, especially if conditioned on simulation parameters to generate targeted variations. Numerous studies in medical imaging leverage GANs for data augmentation.[45]

Libraries such as Albumentations [43], Augmentor [43], or built-in functions in TensorFlow (tf.image [43]) and PyTorch (torchvision.transforms) provide convenient tools for implementing these augmentations.

5.2. Transfer Learning

Transfer learning is a powerful technique where a model pre-trained on a large dataset (often for a different but related task) is used as a starting point for training on a smaller, target dataset.[4]

- **Concept:** The rationale is that the pre-trained model has already learned general-purpose features (e.g., edges, textures, basic shapes from images) from the large dataset. These learned features can be beneficial for the new task, even if the domains are different.
- **Benefits:**
    - Often leads to better performance, especially when the target dataset is small.
    - Can result in faster convergence during training because the model starts with well-initialized weights rather than random ones.[4]
- **Strategy for Segmentation (e.g., U-Net):**
    - Select a U-Net architecture that has a standard CNN backbone (e.g., ResNet, VGG, EfficientNet) for its encoder part.[4]
    - Initialize the encoder with weights pre-trained on a large image dataset like

ImageNet (for general visual features) or, ideally, a large dataset of medical or scientific images if available.
  - **Fine-tuning:** The pre-trained weights are then "fine-tuned" on the specific radiograph dataset. This can involve:
    - Training only the decoder part of the U-Net while keeping the pre-trained encoder weights frozen (at least initially). This is suggested as potentially effective for medical image segmentation.[7]
    - Unfreezing some or all of the encoder layers and training them with a smaller learning rate than the randomly initialized decoder layers.
- **Challenges:**
  - **Domain Mismatch:** The effectiveness of transfer learning can be reduced if the source domain (e.g., natural images from ImageNet) is vastly different from the target domain (radiographs). The benefits are often more pronounced when transferring from a more similar domain.
  - **Task/Data Dependence:** Improvements in accuracy are not guaranteed and can be marginal, particularly if the tasks are very different or if the target dataset, though small, is highly unique.[7] The largest gains are typically seen for challenging tasks with small target datasets.[7]

5.3. Semi-Supervised Learning (SSL)

Semi-supervised learning techniques leverage the information present in a large pool of *unlabeled* data in conjunction with a small amount of labeled data.[5] This is highly relevant for the current PoC, given the availability of simulated radiographs without corresponding pixel labels.

- **Common Techniques:**
  - **Pseudo-Labeling (Self-Training):**
    1. Train an initial model on the small labeled dataset.
    2. Use this model to make predictions (generate "pseudo-labels") on the unlabeled dataset.
    3. Select the pseudo-labels where the model is most confident (e.g., pixels with high softmax probabilities).
    4. Add these confidently pseudo-labeled samples to the labeled training set.
    5. Retrain the model on the combined labeled and pseudo-labeled data. This process can be iterated. This is a conceptually simple yet often effective SSL method.[47]
  - **Consistency Regularization:** The core idea is that the model's predictions for an unlabeled sample should remain consistent even if the sample is slightly perturbed (e.g., through data augmentation).

        - **Mean Teacher:** A popular consistency regularization method. It involves two models: a "student" model (the one being trained) and a "teacher" model. The teacher model's weights are an exponential moving average (EMA) of the student model's weights. The student model is trained to minimize a consistency loss, which penalizes differences between its predictions on augmented unlabeled data and the teacher's predictions on the same augmented unlabeled data.[5] This approach has shown success in medical image segmentation.[47]
- **Relevance:** SSL methods are particularly appealing for medical and scientific imaging where unlabeled data might be plentiful but expert annotation is a bottleneck.[48]

5.4. Self-Supervised Learning (SSL - for Pre-training)

Self-supervised learning is another paradigm for learning from unlabeled data, typically used for pre-training a model to learn meaningful representations before fine-tuning it on a downstream supervised task with limited labels.[50]

- **Concept:** SSL creates "surrogate" or "pretext" tasks where the labels are derived automatically from the unlabeled data itself. The model learns to solve these pretext tasks, and in doing so, learns useful features about the data.
- **Examples of Pretext Tasks for Images:**
    - **Image Colorization:** Training a model to predict the color version of a grayscale image.
    - **Image Inpainting:** Training a model to fill in missing or masked-out patches of an image.
    - **Jigsaw Puzzle:** Training a model to reassemble shuffled patches of an image into their correct configuration.
    - **Rotation Prediction:** Training a model to predict the degree of rotation applied to an image.
    - **Contrastive Learning (e.g., SimCLR, MoCo, BYOL):** These methods train a model to learn representations such that augmented versions ("views") of the same image are close together in the embedding space, while views from different images are far apart.[52]
- **Benefit:** SSL allows the model to learn domain-specific features directly from the unlabeled radiograph data before it encounters any manual labels. This can lead to better initialization for the supervised fine-tuning phase compared to random initialization or even standard ImageNet pre-training if the domain is very different.

5.5. (Optional/Advanced) Active Learning

If there is capacity for acquiring a small number of additional labels during the PoC, active learning can help guide this process efficiently.[6]

- **Concept:** Instead of randomly selecting samples for labeling, active learning involves an iterative process where the model identifies the most *informative* unlabeled samples. These selected samples are then labeled by an expert and added to the training set, and the model is retrained. The goal is to achieve higher model performance with fewer labeled samples compared to random sampling.
- **Querying Strategies for Image Segmentation** [6]**:**
  - **Uncertainty Sampling:** The model queries samples for which its predictions are most uncertain (e.g., low softmax confidence, high entropy in predicted probabilities, or disagreement among an ensemble of models). For segmentation, this can be applied at the pixel or region level.[6]
  - **Diversity Sampling:** The model queries samples that are different from those already in the labeled set, aiming to cover the data distribution more broadly.
- **Relevance:** Useful if the team can iteratively label a few more images based on model feedback. Regional querying, where only uncertain regions within an image are labeled, can further reduce effort.[6]

A combined strategy often yields the best results in data-scarce scenarios. For instance, one might start with self-supervised pre-training on all available unlabeled simulation data to initialize the U-Net encoder. Then, this model could be fine-tuned using the small labeled set, augmented heavily. Subsequently, semi-supervised techniques could be applied to leverage the remaining unlabeled data during the supervised training phase. The specific combination and sequence of these techniques will require experimentation, as their effectiveness can be highly task- and data-dependent.[7] The simulation parameters themselves might offer novel ways to enhance these data scarcity solutions, for example, by conditioning GAN-based augmentation on these parameters to generate more diverse and physically relevant synthetic data, or by guiding augmentations to reflect realistic variations implied by parameter ranges.

Table 3: Techniques for Handling Limited Labeled Data

Technique	Brief Description	How it Uses Labeled/Unlabeled Data	Key Considerations for	Potential Impact on PoC

			Radiographs & Simulation Parameters	
Data Augmentation (Geometric) [3]	Creates modified versions of labeled images via flips, rotations, crops, scaling, translations.	Uses labeled data only (to expand it).	Transformations must be physically plausible for radiographs. Simulation parameters could guide ranges (e.g., realistic rotation limits).	Increases diversity of labeled set, improves model robustness, reduces overfitting. High impact, relatively easy to implement.
Data Augmentation (Intensity/Noise) [3]	Modifies brightness/contrast, adds noise, applies elastic deformations to labeled images.	Uses labeled data only.	Can simulate detector variations or material differences. Elastic deformations good for biological-like variability. Parameters might define realistic noise levels or intensity ranges.	Similar to geometric augmentation; helps model generalize to different image qualities.
Transfer Learning [4]	Initializes model with weights pre-trained on a large dataset, then fine-tunes on the small labeled target dataset.	Uses labeled data for fine-tuning. Leverages knowledge from large (often unrelated) external dataset.	Choose pre-trained backbone relevant to vision (e.g., ResNet). Medical/scientific pre-trained models are ideal but rarer. Effectiveness depends on domain	Can provide a strong starting point, faster convergence, and better performance with limited data, especially if pre-trained features are relevant.

			similarity. Simulation parameters are not directly used here but define the target domain.	
Semi-Supervised Learning (Pseudo-Labeling) [47]	Model trained on labeled data predicts labels for unlabeled data; high-confidence pseudo-labels are added to training set and model retrained.	Uses both labeled (initial training, retraining) and unlabeled data (for pseudo-labeling).	Confidence threshold for pseudo-labels is critical to avoid error propagation. Quality of initial model matters. Simulation parameters could potentially help filter or weight pseudo-labels if they correlate with confidence.	Can significantly boost performance by leveraging large amounts of unlabeled simulation data.
Semi-Supervised Learning (Consistency Reg.) [5]	Enforces model's predictions on unlabeled data to be consistent under perturbations (e.g., augmentations). Mean Teacher is an example.	Uses both labeled (for supervised loss) and unlabeled data (for consistency loss).	Requires careful design of perturbations and consistency loss. Simulation parameters could define realistic perturbation space.	Effective way to utilize unlabeled data by enforcing robust representations.
Self-Supervised Pre-training [50]	Model learns representations from unlabeled data by solving a pretext task (e.g., inpainting, contrastive learning). Then	Uses unlabeled data for pre-training, then labeled data for fine-tuning.	Choice of pretext task is important. Can learn domain-specific features from the radiographs. Simulation	Provides better weight initialization than random, potentially outperforming generic pre-training if

	fine-tuned on labeled data.		parameters could inform pretext task design (e.g., predict a masked parameter from an image).	unlabeled data is rich and relevant.

6. Model Evaluation and Interpretation

Once models are trained, their performance must be rigorously evaluated, and their decision-making processes understood. This is crucial for iterating on the PoC, identifying areas for improvement, and building confidence in the results.

6.1. Segmentation Metrics

For the pixel-level classification task (identifying neutron sources, gamma sources, noise, etc.), several standard metrics are used to evaluate performance [13]:

- **Intersection over Union (IoU) / Jaccard Index:** This metric calculates the ratio of the area of overlap between the predicted segmentation mask and the ground truth mask to the area of their union, for a given class. It is calculated as: IoU=TP/(TP+FP+FN) where:
  - TP (True Positives): Number of pixels correctly classified as the class.
  - FP (False Positives): Number of pixels incorrectly classified as the class (Type I error).
  - FN (False Negatives): Number of pixels belonging to the class but incorrectly classified as something else (Type II error). IoU ranges from 0 (no overlap) to 1 (perfect overlap). **Mean IoU (mIoU)** is often reported, which is the average IoU calculated across all classes. It is a standard metric for semantic segmentation benchmarks.[14]
- **Dice Coefficient (F1 Score):** This metric is conceptually similar to IoU and also measures the overlap between predicted and ground truth masks. It is particularly popular in medical image segmentation. It is calculated as: Dice=(2·TP)/(2·TP+FP+FN) Like IoU, Dice ranges from 0 to 1. It tends to give slightly more weight to TPs than IoU.
- **Pixel Accuracy:** This is the simplest metric, representing the percentage of pixels in the image that were correctly classified: PixelAccuracy=(TP+TN)/(TP+TN+FP+FN) where TN (True Negatives) is the number of pixels correctly classified as not belonging to the class. While easy to understand, pixel accuracy can be misleading, especially in cases of severe class

imbalance. For example, if 90% of an image is "background/noise" and the model correctly classifies all background pixels but fails on all source pixels, it would still achieve 90% pixel accuracy, which is not reflective of its utility for source identification.

When interpreting these metrics, it's important to consider the project's specific goals. For instance, if failing to detect a neutron source (a False Negative for the "neutron" class) is more critical than incorrectly identifying a noise pixel as a neutron source (a False Positive), then metrics like recall (sensitivity) for the neutron class (TP/(TP+FN)) become particularly important, alongside overall metrics like mIoU. These priorities should be discussed with the PM.

6.2. Regression Metrics (for Energy Prediction)

For the task of predicting energy (a continuous value), the following regression metrics are commonly used [20]:

- **Mean Absolute Error (MAE):** This measures the average absolute difference between the predicted energy values and the true energy values. $MAE=(1/N)\cdot\Sigma|TrueEnergyi-PredictedEnergyi|$ MAE is interpretable in the original units of energy (e.g., MeV) and is less sensitive to outliers than RMSE.
- **Root Mean Squared Error (RMSE):** This calculates the square root of the average of the squared differences between predicted and true energy values. $RMSE=(1/N)\cdot\Sigma(TrueEnergyi-PredictedEnergyi)2$ RMSE penalizes larger errors more heavily due to the squaring term. It is more sensitive to outliers. If large errors in energy prediction are particularly detrimental, RMSE might be a more appropriate metric to optimize or monitor.

The choice between MAE and RMSE depends on how errors in energy prediction should be weighted. If all errors are considered equally problematic regardless of magnitude, MAE is suitable. If large errors are disproportionately worse, RMSE is preferred.

6.3. Interpreting Model Decisions (Explainable AI - XAI)

Understanding *why* an ML model makes certain predictions is crucial, especially in scientific applications and for a team new to ML. XAI techniques can help build trust, debug models, and verify if the model is learning scientifically meaningful patterns.[59]

- **Feature Importance Analysis:** These techniques aim to identify which input features (e.g., specific pixel regions in the image, or particular simulation parameters if used as direct inputs) contribute most significantly to the model's

output for a given prediction.[61]
  - **LIME (Local Interpretable Model-agnostic Explanations):** LIME explains individual predictions by learning a simpler, interpretable model (e.g., a linear model) locally around the prediction.[59] For images, it can highlight superpixels that were important for a classification.
  - **SHAP (SHapley Additive exPlanations):** SHAP uses concepts from game theory to assign an importance value (SHAP value) to each feature, representing its contribution to the prediction compared to a baseline.[59] SHAP can provide both local (per-prediction) and global (overall feature importance) explanations.
- **Visualizing Activations/Saliency Maps:** For CNNs, various techniques can visualize which parts of an input image "activate" certain neurons or contribute most to the final decision.
  - **Grad-CAM (Gradient-weighted Class Activation Mapping):** Produces a coarse localization map highlighting the important regions in the image for predicting a specific class.[62]

Applying these XAI techniques can help validate whether the model is focusing on physically relevant signatures in the radiographs (e.g., areas with high attenuation for the predicted source type) or if it's latching onto artifacts or biases in the simulated data. This connection between ML interpretability and domain-specific (physics) understanding is a powerful tool for model validation and refinement.[60]

6.4. Uncertainty Estimation

Quantifying the model's confidence in its predictions can provide valuable insights.[63] Regions or samples where the model exhibits high uncertainty might indicate:

- Ambiguous or noisy input.
- Novel or out-of-distribution samples.
- Areas where the model's learned representations are weak, possibly requiring more training data or focused augmentation.
- **Introductory Methods for Uncertainty Quantification [64]:**
  - **Monte Carlo Dropout:** During inference (prediction time), dropout layers (typically only active during training) are kept active. By running the same input through the model multiple times with different dropout masks, a distribution of predictions is obtained. The variance or entropy of this distribution can serve as an uncertainty measure.
  - **Deep Ensembles:** Multiple models (e.g., with different random initializations or trained on slightly different data subsets) are trained independently. The disagreement (variance) in their predictions for a given input can indicate

uncertainty.

- **Relevance:** High-uncertainty predictions are less reliable. Identifying these can help in debugging the model or, in a more advanced scenario, guide an active learning strategy by suggesting which samples would be most beneficial to label next.[63] For this PoC, understanding where the model is uncertain can highlight weaknesses in the current training data or model capabilities, guiding iterative refinement. For example, if the model is consistently uncertain about gamma sources near material interfaces, this might prompt an investigation into how these interfaces are simulated or if more augmented examples of such scenarios are needed.

Table 4: Evaluation Metrics for PoC

Task	Metric	Brief Formula/Explanation	What it Measures	Key Considerations for Interpretation
Pixel-Level Segmentation	**mIoU (Mean Intersection over Union)** [13]	Average of (TP / (TP + FP + FN)) across all classes.	Average overlap between predicted and ground truth masks per class.	Robust general metric for segmentation.
	Dice Coefficient (per class) [13]	(2 * TP) / (2 * TP + FP + FN) for each class.	Overlap for a specific class; sensitive to TP.	Good for imbalanced classes; often used in medical imaging. Can struggle with evaluating complex shapes accurately.[13]
	Pixel Accuracy [14]	(Correctly classified pixels) / (Total pixels).	Overall percentage of correct pixel classifications.	Can be misleading if classes are imbalanced (e.g., large background).

Energy Regression	**MAE (Mean Absolute Error)** [20]	Average of	Predicted - True	.
	RMSE (Root Mean Squared Error) [20]	Sqrt(Average of (Predicted - True)^2).	Square root of average squared error; penalizes large errors more.	More sensitive to outliers. Use if large errors are particularly undesirable. Not directly comparable to MAE values.[58]

7. Advanced Considerations & Future Directions

While the primary focus of the PoC is to establish feasibility and build foundational understanding, it is beneficial to consider more advanced techniques and future directions that could be explored once the initial objectives are met. These considerations are primarily for the ML Lead and PM to inform the longer-term roadmap.

7.1. Anomaly Detection for "Noise" or Unexpected Signatures

The current plan includes "noise" as a segmentation class. However, if "noise" encompasses not just random sensor-like fluctuations but potentially unmodeled physical phenomena, simulation artifacts, or entirely unexpected signatures, then framing this as an anomaly detection problem could be more powerful.[66]

- **Concept:** Anomaly detection aims to identify data points or patterns that deviate significantly from what is considered "normal."
- **Methods:**
  - **Autoencoders:** An autoencoder neural network can be trained exclusively on examples of "normal" signatures (e.g., well-defined neutron and gamma events). When presented with an input, the autoencoder attempts to reconstruct it. If the input is anomalous (i.e., different from the normal data it was trained on), the reconstruction error will be high, flagging it as an anomaly.[66] This could be a way to identify one of the "one or two other classes" mentioned in the user query.
  - **Other Unsupervised Methods:** Techniques like Local Outlier Factor (LOF) or One-Class SVM could also be explored to identify pixels or regions that are dissimilar to the known source classes.[66]
- **Relevance:** This approach could help in discovering unexpected patterns in the

simulation data or identifying events that don't fit neatly into the predefined "neutron" or "gamma" categories. It offers a more principled way to handle "unknowns" than simply labeling them as generic noise.

7.2. Exploring Physics-Informed Neural Networks (PINNs) More Deeply

As the team gains experience, revisiting Physics-Informed Machine Learning (PIML) with greater depth could unlock further improvements, particularly if the initial PoC models struggle with physical consistency.

- **Recap:** PIML seeks to embed domain knowledge, often in the form of mathematical equations governing physical laws, directly into the ML model's learning process, typically via the loss function or network architecture.[1]
- **Potential:** If the models produce segmentations or energy predictions that violate known physical constraints (e.g., impossible energy values for a given source type, incorrect relationships between attenuation and material parameters if those are known), PINNs could help enforce these constraints. This can lead to more accurate, robust, and physically plausible models, often with improved data efficiency because the physical constraints act as a strong regularizer.[2]
- **Consideration:** Implementing full PINNs is a significant step up in complexity and typically requires a good understanding of the underlying differential equations. This is likely a post-PoC research direction but holds substantial promise for scientific ML applications.[1]

7.3. Leveraging Simulation Parameters for Enhanced Model Robustness and Meta-Learning

The simulation parameters offer opportunities beyond just direct input features.

- **Sensitivity Analysis:** Systematically varying simulation parameters and observing the impact on model predictions can provide insights into the model's robustness across different physical scenarios. This can also highlight which parameters the model is most sensitive to.
- **Meta-Learning / Domain Adaptation:** If simulations can be run with different configurations (e.g., varying detector types, material compositions, environmental conditions), the simulation parameters effectively define different but related "domains" or "tasks." Advanced techniques like meta-learning could train a model that can generalize across these different simulation setups or quickly adapt to a new, unseen set of parameters with minimal additional training.[1] This is a more advanced concept but represents a powerful way to leverage parameterized simulations if diverse simulation sets are available or can be generated.

7.4. Path from PoC to a More Robust System

The PoC is the first step. If successful, future work could involve:

- **Scaling Data:** Generating more simulated data, potentially guided by insights from the PoC (e.g., focusing on scenarios where the PoC model was uncertain). If real-world deployment is a goal, planning for the acquisition and labeling of real radiograph data would be necessary, along with strategies to bridge the "Sim2Real" gap.[7]
- **More Complex Models:** Exploring more advanced segmentation or MTL architectures if justified by performance plateaus with simpler models.
- **Rigorous MLOps Pipeline:** Implementing a more comprehensive MLOps pipeline for robust training, deployment, monitoring, and retraining as the project matures beyond the PoC stage.[36]
- **Validation:** If the application has parallels to medical imaging, eventual validation against real-world data and expert evaluations would be critical.[9]

The PoC itself can serve as an invaluable tool for informing future simulation efforts. If the model consistently struggles with certain types of signatures, energy ranges, or specific combinations of simulation parameters, it might indicate a need to generate more simulated data in those regimes to improve data coverage and model training. This creates a beneficial feedback loop between ML development and the simulation process.

8. Recommendations and Roadmap for the Team

This section provides a phased plan for the PoC, suggests relevant tools, and offers guidance on task prioritization to help the team navigate this initial ML project successfully. The roadmap is designed to be incremental, allowing the team to build skills and confidence progressively.

8.1. Step-by-Step Phased Plan for the PoC

This PoC can be structured into distinct phases, each with clear objectives:

- **Phase 1: Setup, Data Familiarization, and Initial Baseline (Estimated Duration: 2-3 weeks)**
  1. **Environment Setup:** Install necessary ML libraries (Python, PyTorch/TensorFlow), data processing tools, and establish a version control system (e.g., Git).
  2. **Data Loading and Exploration:** Develop scripts to load the 256x256 grayscale images and their associated simulation parameters. Perform initial

visual inspection and basic statistical analysis of the images and parameters. Understand the format of the limited labeled data.

3. **Basic Preprocessing:** Implement essential preprocessing steps:
   - Normalization of pixel values (e.g., scaling to).[29]
   - Verification of image dimensions.
4. **Simple U-Net Implementation:** Build a basic U-Net model for semantic segmentation.[12] Initially, focus on a simplified problem: segmenting perhaps only "neutron source" vs. "gamma source" vs. "background/noise."
5. **Training and Baseline Metrics:** Train this U-Net model on the *existing small labeled dataset*. Establish baseline performance using core segmentation metrics (e.g., mIoU, Dice score per class) on a held-out validation/test split of the labeled data.[13]

- **Goal of Phase 1:** Have a fully working data loading, preprocessing, training, and evaluation pipeline. Obtain initial performance figures that will serve as the benchmark for all subsequent improvements. Familiarize the team with the end-to-end workflow.

- **Phase 2: Incorporating Simulation Parameters & Initial Energy Prediction (Estimated Duration: 2-3 weeks)**
  1. **Integrate Simulation Parameters:** Experiment with simple ways to feed relevant simulation parameters into the baseline U-Net model. This could involve concatenating parameter vectors to the U-Net's bottleneck or to flattened image features before a classification layer. Evaluate if this provides any uplift over the image-only baseline.
  2. **Energy Prediction Head:** Add a regression head to the U-Net architecture (or develop a small separate model that takes segmented regions as input) to predict energy. Define the precise energy target based on simulation parameters.
  3. **Train and Evaluate Energy Prediction:** Train the energy prediction component and evaluate using MAE and RMSE.[57]
  4. **(Optional) Basic Multi-Task Learning:** If time permits and the team feels comfortable, attempt a simple MTL setup where the U-Net predicts both segmentation and energy simultaneously, using a combined loss function.[40]

  - **Goal of Phase 2:** Assess the initial impact of using simulation parameters directly and determine the feasibility of the energy prediction task. Gain experience with regression and potentially basic MTL.
- **Phase 3: Addressing Data Scarcity (Estimated Duration: 3-4 weeks)**
  1. **Data Augmentation:** Implement a robust data augmentation pipeline for the labeled training set, including:
     - Geometric transformations (flips, small rotations, crops, scaling,

translations).[3]
    - Intensity/noise-based augmentations (brightness/contrast adjustments, Gaussian noise).[3] Retrain the best model configuration from Phase 1 or 2 with augmented data and evaluate the performance improvement.
  2. **Transfer Learning:**
    - Identify a U-Net architecture with a standard pre-trained encoder backbone (e.g., ResNet50 pre-trained on ImageNet).
    - Implement transfer learning by initializing the encoder with these pre-trained weights and fine-tuning on the (augmented) labeled radiograph dataset.[4] Experiment with freezing encoder layers versus fine-tuning all layers. Retrain and evaluate.
  3. **(Optional, if significant unlabeled data exists) Basic Semi-Supervised Learning:** If the previous steps still yield suboptimal performance and there's a large pool of unlabeled simulated images, explore a straightforward SSL technique like pseudo-labeling.[5]
  - **Goal of Phase 3:** Achieve a significant performance improvement over the baseline by effectively leveraging techniques designed for limited labeled data scenarios.
- **Phase 4: Comprehensive Evaluation, Interpretation, and Reporting (Estimated Duration: 2-3 weeks)**
  1. **Final Model Evaluation:** Select the best performing model configuration from Phase 3. Conduct a thorough evaluation on the held-out test set using all defined segmentation and regression metrics.
  2. **Model Interpretation (XAI):** Apply interpretability techniques (e.g., LIME, SHAP, or Grad-CAM) to a few representative correct and incorrect predictions to understand what features the model is using.[59]
  3. **Uncertainty Estimation (Basic):** For a few sample predictions, estimate the model's uncertainty (e.g., using Monte Carlo Dropout) to identify areas where the model is less confident.[63]
  4. **Documentation and Reporting:** Compile all findings, results, key learnings, challenges encountered, and insights gained. Prepare a comprehensive report and presentation for the PM and stakeholders, clearly outlining what was achieved, the limitations, and potential future directions.
  - **Goal of Phase 4:** Develop a deep understanding of the final PoC model's capabilities and limitations. Communicate these effectively, along with actionable recommendations.

8.2. Tooling and Library Suggestions

- **Core ML Frameworks:**

    - **PyTorch** or **TensorFlow/Keras:** Both are powerful and widely used deep learning frameworks with extensive documentation and community support.[22] The choice may depend on team familiarity or specific library availability for certain pre-trained models.
- **Image Processing and Augmentation:**
    - **OpenCV:** A comprehensive library for general computer vision tasks and image manipulation.
    - **Scikit-image:** Another excellent Python library for image processing.
    - **Albumentations:** A popular library specifically designed for fast and flexible image augmentation, widely used in deep learning.[43]
    - **Augmentor:** A Python package for image augmentation and artificial image generation.[43]
- **Experiment Tracking (Basic for PoC):**
    - **MLflow:** An open-source platform to manage the ML lifecycle, including experiment tracking, model versioning, and deployment. Even its basic tracking features are useful for a PoC.[36]
    - **TensorBoard:** A visualization toolkit, often used with TensorFlow (and available for PyTorch), for visualizing training metrics, model graphs, and embeddings.
    - **Simple Spreadsheets/CSV Files:** For a very initial PoC, disciplined manual logging in spreadsheets can work, but automated tools are preferable as complexity grows.
- **Explainable AI (XAI) Libraries:**
    - **SHAP Library:** For calculating SHAP values for feature importance.
    - **LIME Library:** For implementing LIME explanations.
    - Many XAI techniques are also being integrated into core ML frameworks or specialized packages.

8.3. Prioritization of Tasks and Experiments

Given the team's novice status and the PoC nature of the project:

1. **Focus on Robust Segmentation First:** Achieving accurate pixel-level identification of neutron and gamma sources is the primary challenge. Energy prediction can be built upon a solid segmentation foundation.
2. **Data Augmentation is Key:** This is often a high-impact, relatively straightforward technique to implement for improving performance with limited labeled data. Prioritize this early in Phase 3.
3. **Transfer Learning before Complex SSL/Self-Supervised:** Transfer learning provides a good balance of potential performance gain and implementation

complexity for a novice team. More advanced SSL or self-supervised learning techniques can be explored if transfer learning and augmentation are insufficient.

4. **Iterate and Learn:** Encourage the team to experiment, document results meticulously, and not be afraid of "failed" experiments, as these often provide valuable learning opportunities.

8.4. Key Learning Resources for the Team

To support the team's learning journey:

- **Online Courses:** Platforms like Coursera, Udacity, and fast.ai offer excellent introductory and advanced courses on machine learning and deep learning.
- **Framework Documentation:** The official documentation for PyTorch or TensorFlow/Keras is an essential resource.
- **Key Research Papers:** Encourage reading foundational papers related to the techniques being used (e.g., the original U-Net paper, key papers on data augmentation or transfer learning cited in this report).
- **Internal Knowledge Sharing:** Regular team meetings to discuss progress, challenges, and learnings can accelerate skill development. Pair programming or collaborative debugging sessions can also be beneficial.

This phased roadmap is designed not just to produce a technical artifact, but also to serve as a structured learning experience for the team. By breaking down the complex problem into manageable stages, each building upon the last, the team can incrementally develop both the PoC model and their own ML expertise. The success of this PoC should be measured not only by the final model's performance metrics but also by the team's growth in understanding and the clarity of the insights provided to the PM regarding the possibilities and limitations of using ML for this radiograph analysis task.

9. Conclusions and Recommendations

This strategic report has outlined a comprehensive approach for developing a Machine Learning Proof of Concept (PoC) to perform pixel-level semantic segmentation and energy estimation on simulated radiograph images. The plan emphasizes an iterative development process, foundational learning for a team new to ML, and leveraging the unique aspects of the provided data, namely the 256x256 grayscale simulated images and their associated physics simulation parameters, in the context of limited labeled data.

Key Findings and Implications:

1. **Leveraging Simulation Data:** The availability of high-fidelity simulation data and parameters is a distinct advantage, offering a controlled environment for model development and the potential to incorporate physics-informed principles.[1] However, careful consideration of the simulation parameter space and potential "Sim2Real" gaps is necessary if future application to real-world data is envisioned.[7]
2. **Addressing Data Scarcity:** The limited labeled data is the primary challenge. A multi-faceted strategy combining robust **data augmentation** [3], **transfer learning** [4], and potentially **semi-supervised or self-supervised learning** [5] will be crucial for developing a generalizable model.
3. **Model Architecture Choice:** The **U-Net architecture** [12] is highly recommended as the starting point for semantic segmentation due to its proven efficacy in medical/scientific imaging, its ability to preserve fine details via skip connections, and its suitability for the given image dimensions. **Multi-Task Learning (MTL)** [40] presents a natural way to combine segmentation and energy prediction, though it introduces added complexity in balancing tasks.
4. **Importance of Fundamentals and Iteration:** For a team new to ML, a strong grasp of core concepts (supervised learning, neural network basics, loss functions, evaluation metrics) is essential.[11] An iterative PoC development cycle [26], starting with simple baselines and progressively adding complexity, will facilitate both learning and effective problem-solving.
5. **Evaluation and Interpretability:** Rigorous evaluation using appropriate metrics (mIoU, Dice for segmentation; MAE, RMSE for regression) is critical.[13] Furthermore, incorporating **Explainable AI (XAI)** techniques [59] and **uncertainty estimation** [63] will be vital for debugging, building trust, and validating that the model learns physically meaningful relationships.
6. **Role of Simulation Parameters:** These parameters can be used not only as direct input features but also to guide data augmentation, inform physics-constrained modeling, and potentially enable advanced techniques like meta-learning if diverse simulation sets are available.

Recommendations for the PoC Implementation:

1. **Prioritize a Phased Approach:** Follow the outlined four-phase roadmap:
   - **Phase 1:** Setup, data familiarization, and a simple U-Net baseline for segmentation.
   - **Phase 2:** Integration of simulation parameters and initial energy prediction.
   - **Phase 3:** Systematic application of data augmentation and transfer learning to address data scarcity.
   - **Phase 4:** Thorough evaluation, model interpretation using XAI, and

comprehensive reporting.

2. **Emphasize Team Learning and Enablement:** The PoC should be viewed as a learning opportunity. Allocate time for studying foundational concepts and the chosen tools/libraries. Foster a collaborative environment for knowledge sharing.
3. **Establish Strong Baselines Early:** Quantify the performance of simple models on the limited labeled data before implementing more complex techniques. This provides a clear benchmark for measuring progress.[35]
4. **Systematic Experimentation and Tracking:** Adopt basic MLOps principles from the start, including meticulous tracking of experiments, data versions, and model configurations to ensure reproducibility and facilitate learning from each iteration.[27]
5. **Focus Data Scarcity Efforts:** Data augmentation is likely to provide significant benefits with relatively moderate implementation effort and should be prioritized. Transfer learning is the next logical step. Semi-supervised and self-supervised methods can be explored if these initial approaches are insufficient.
6. **Iteratively Refine Energy Prediction:** Clarify the exact definition of "energy" to be predicted. Start with a simpler region-based regression approach before attempting pixel-wise regression or complex MTL, if necessary.
7. **Engage with XAI for Validation:** Use interpretability tools not just as a final step, but as a debugging and validation aid throughout development to ensure the model is learning relevant physical features rather than spurious correlations from the simulations.[60]
8. **Clearly Define PoC Success:** Success for this PoC includes not only achieving reasonable model performance but also enabling the team with ML skills and providing the PM with a clear understanding of what is possible, the current limitations, and a data-driven basis for future decisions.

By adopting this strategic and iterative approach, the team can effectively navigate the challenges of this project, develop a meaningful Proof of Concept, and build a strong foundation for future ML endeavors in analyzing radiograph data. The combination of simulated data with physics parameters presents a unique opportunity to explore the intersection of machine learning and physics-based modeling.

Works cited

1. Machine Learning with Physics Knowledge for Prediction: A Survey - arXiv, accessed June 5, 2025, https://arxiv.org/html/2408.09840v2
2. Physics-Informed Computer Vision: A Review and Perspectives - arXiv, accessed June 5, 2025, https://arxiv.org/html/2305.18035v3
3. Data Augmentation: Techniques, Examples & Benefits | CCSLA, accessed June 5,

2025, https://www.ccslearningacademy.com/what-is-data-augmentation/
4. (PDF) Segmentation Performance Analysis of Transfer Learning ..., accessed June 5, 2025, https://www.researchgate.net/publication/384334895_Segmentation_Performance_Analysis_of_Transfer_Learning_Models_on_X-Ray_Pneumonia_Images
5. Semi-Supervised Semantic Segmentation | Papers With Code, accessed June 5, 2025, https://paperswithcode.com/task/semi-supervised-semantic-segmentation
6. Dynamic-budget superpixel active learning for semantic ... - Frontiers, accessed June 5, 2025, https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2024.1498956/full
7. Transfer Learning in Medical Image Segmentation: New Insights from Analysis of the Dynamics of Model Parameters and Learned Representations, accessed June 5, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC8164174/
8. Redefining Radiology: A Review of Artificial Intelligence Integration in Medical Imaging - PMC - PubMed Central, accessed June 5, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC10487271/
9. Medical image analysis using deep learning algorithms - PMC, accessed June 5, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC10662291/
10. Machine Learning for Medical Imaging Analysis: A Comprehensive ..., accessed June 5, 2025, https://www.basic.ai/blog-post/medical-imaging-analysis-machine-learning-overview
11. 7 Key Terms Every Machine Learning Beginner Should Know ..., accessed June 5, 2025, https://machinelearningmastery.com/7-key-terms-every-machine-learning-beginner-should-know/
12. Segmentation: U-Net, Mask R-CNN, and Medical Applications ..., accessed June 5, 2025, https://glassboxmedicine.com/2020/01/21/segmentation-u-net-mask-r-cnn-and-medical-applications/
13. A Deep Dive Into Semantic Segmentation Evaluation Metrics ..., accessed June 5, 2025, https://hackernoon.com/a-deep-dive-into-semantic-segmentation-evaluation-metrics
14. Semantic Segmentation | Papers With Code, accessed June 5, 2025, https://paperswithcode.com/task/semantic-segmentation
15. Neutron Imaging Gallery | See Our Work - Phoenix Neutron Imaging, accessed June 5, 2025, https://www.phoenixneutronimaging.com/insights-and-updates/neutron-image-gallery
16. AP/INT-04 Principles and Applications of Neutron-based Inspection Techniques, accessed June 5, 2025, https://www-pub.iaea.org/MTCD/publications/PDF/P1433_CD/datasets/papers/ap_int-04.pdf

17. Neutron detection - Wikipedia, accessed June 5, 2025, https://en.wikipedia.org/wiki/Neutron_detection
18. A Multi-Task Convolutional Neural Network for Semantic Segmentation and Event Detection in Laparoscopic Surgery, accessed June 5, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC10054284/
19. U-Net Architecture Explained | GeeksforGeeks, accessed June 5, 2025, https://www.geeksforgeeks.org/u-net-architecture-explained/
20. What is Loss Function? | IBM, accessed June 5, 2025, https://www.ibm.com/think/topics/loss-function
21. Loss Functions in Deep Learning: A Comprehensive Review - arXiv, accessed June 5, 2025, https://arxiv.org/html/2504.04242v1
22. Adam Optimizer Simplified for Beginners in ML - ProjectPro, accessed June 5, 2025, https://www.projectpro.io/article/adam-optimizer/986
23. What is Adam Optimizer? | GeeksforGeeks, accessed June 5, 2025, https://www.geeksforgeeks.org/adam-optimizer/
24. What is Overfitting? - Overfitting in Machine Learning Explained - AWS, accessed June 5, 2025, https://aws.amazon.com/what-is/overfitting/
25. What Is Overfitting vs. Underfitting? - IBM, accessed June 5, 2025, https://www.ibm.com/think/topics/overfitting-vs-underfitting
26. All about the Iterative Design Process | Smartsheet, accessed June 5, 2025, https://www.smartsheet.com/iterative-process-guide
27. MLOps Principles - Ml-ops.org, accessed June 5, 2025, https://ml-ops.org/content/mlops-principles
28. Are all grayscale images of the size 256*256. Please clarify ..., accessed June 5, 2025, https://www.mathworks.com/matlabcentral/answers/257826-are-all-grayscale-images-of-the-size-256-256-please-clarify
29. Image Preprocessing Techniques - FasterCapital, accessed June 5, 2025, https://fastercapital.com/topics/image-preprocessing-techniques.html/1
30. The Ultimate Guide to Preprocessing Medical Images: Techniques, Tools, and Best Practices for Enhanced Diagnosis, accessed June 5, 2025, https://about.cmrad.com/articles/the-ultimate-guide-to-preprocessing-medical-images-techniques-tools-and-best-practices-for-enhanced-diagnosis
31. Sample image collection of the 200 (256x256) grayscale images used for the tests. - ResearchGate, accessed June 5, 2025, https://www.researchgate.net/figure/Sample-image-collection-of-the-200-256x256-grayscale-images-used-for-the-tests_fig1_221458028
32. Leveraging Machine Learning for Advanced Nanoscale X-ray ..., accessed June 5, 2025, https://pubs.acs.org/doi/abs/10.1021/acs.nanolett.4c02446
33. Using the Deep Learning Algorithm to Determine the Presence of Sacroiliitis from Pelvic Radiographs - MDPI, accessed June 5, 2025, https://www.mdpi.com/2075-1729/15/6/876
34. Machine Learning for Medical Imaging | RadioGraphics - RSNA Journals, accessed June 5, 2025, https://pubs.rsna.org/doi/abs/10.1148/rg.2017160130
35. Behind the Scenes: Setting a Baseline for Image Segmentation ..., accessed June

5, 2025, https://www.databricks.com/blog/behind-the-scenes
36. A Practical Guide to Implementing MLOps - Part 1 - Presidio, accessed June 5, 2025, https://www.presidio.com/a-practical-guide-to-implementing-mlops-part-1/
37. Choosing the Right MLOps Platform - Comet, accessed June 5, 2025, https://www.comet.com/site/lp/mlops-platform-guide/
38. Medical Image Segmentation Review: The Success of U-Net - IEEE Computer Society, accessed June 5, 2025, https://www.computer.org/csdl/journal/tp/2024/12/10643318/1ZAxIZmCbDi
39. U-Net in Medical Image Segmentation: A Review of Its Applications Across Modalities - arXiv, accessed June 5, 2025, https://arxiv.org/abs/2412.02242
40. Knowledge-Guided Multi-Task Network for Remote Sensing Imagery, accessed June 5, 2025, https://www.mdpi.com/2072-4292/17/3/496
41. openaccess.thecvf.com, accessed June 5, 2025, https://openaccess.thecvf.com/content/WACV2025/papers/Fontana_Optimizing_Dense_Visual_Predictions_Through_Multi-Task_Coherence_and_Prioritization_WACV_2025_paper.pdf
42. Multi-Scale Convolutional Architecture for Semantic Segmentation - Carnegie Mellon University's Robotics Institute, accessed June 5, 2025, https://www.ri.cmu.edu/pub_files/2015/10/CMU-RI-TR_AmanRaj_revision2.pdf
43. A Complete Guide to Data Augmentation | DataCamp, accessed June 5, 2025, https://www.datacamp.com/tutorial/complete-guide-data-augmentation
44. Data Augmentation Techniques Applied to Medical Images - ResearchGate, accessed June 5, 2025, https://www.researchgate.net/publication/382869099_Data_Augmentation_Techniques_Applied_to_Medical_Images
45. Medical image data augmentation: techniques, comparisons and interpretations - OUCI, accessed June 5, 2025, https://ouci.dntb.gov.ua/en/works/73wMew27/
46. High-Resolution Satellite Image Classification Using Deep Learning - Preprints.org, accessed June 5, 2025, https://www.preprints.org/frontend/manuscript/5f8fde0acb01039a66519b3550b15913/download_pub
47. Semi-supervised Medical Image Segmentation | Papers With Code, accessed June 5, 2025, https://paperswithcode.com/task/semi-supervised-medical-image-segmentation
48. SemiSAM+: Rethinking Semi-Supervised Medical Image Segmentation in the Era of Foundation Models - arXiv, accessed June 5, 2025, https://arxiv.org/html/2502.20749v1
49. Medical Image Segmentation: A Comprehensive Review of Deep Learning-Based Methods, accessed June 5, 2025, https://www.mdpi.com/2379-139X/11/5/52
50. arxiv.org, accessed June 5, 2025, https://arxiv.org/abs/2505.13584
51. Self-supervised Learning with Local Contrastive Loss for Detection and Semantic Segmentation - YouTube, accessed June 5, 2025, https://www.youtube.com/watch?v=5Rlgaz8GPco
52. Self-Supervised Learning for Image Segmentation: A Comprehensive Survey -

arXiv, accessed June 5, 2025, https://arxiv.org/html/2505.13584v1
53. Active Learning for Image Segmentation with Binary User Feedback - CVF Open Access, accessed June 5, 2025, https://openaccess.thecvf.com/content/WACV2025/papers/Goswami_Active_Learning_for_Image_Segmentation_with_Binary_User_Feedback_WACV_2025_paper.pdf
54. Realistic Evaluation of Deep Active Learning for Image Classification and Semantic Segmentation - ResearchGate, accessed June 5, 2025, https://www.researchgate.net/publication/389434764_Realistic_Evaluation_of_Deep_Active_Learning_for_Image_Classification_and_Semantic_Segmentation
55. A comprehensive survey on deep active learning in medical image analysis - Bohrium, accessed June 5, 2025, https://www.bohrium.com/paper-details/a-comprehensive-survey-on-deep-active-learning-in-medical-image-analysis/997690393707413507-3782
56. IoU, Dice coefficient and Pixel accuracy measures evaluated for ..., accessed June 5, 2025, https://www.researchgate.net/figure/IoU-Dice-coefficient-and-Pixel-accuracy-measures-evaluated-for-segmentation-results_tbl3_350157706
57. Root Mean Square Error (RMSE): The cornerstone for ... - Coralogix, accessed June 5, 2025, https://coralogix.com/ai-blog/root-mean-square-error-rmse-the-cornerstone-for-evaluating-regression-models/
58. least squares - Mean absolute error OR root mean squared error ..., accessed June 5, 2025, https://stats.stackexchange.com/questions/48267/mean-absolute-error-or-root-mean-squared-error
59. EXPLORING CONVOLUTIONAL NEURAL NETWORKS FOR RICE GRAIN CLASSIFICATION: AN EXPLAINABLE AI APPROACH - arXiv, accessed June 5, 2025, https://www.arxiv.org/pdf/2505.05513
60. Interpretable Machine Learning in Physics: A Review - arXiv, accessed June 5, 2025, https://arxiv.org/html/2503.23616v1
61. Understanding Feature Importance in Machine Learning | Built In, accessed June 5, 2025, https://builtin.com/data-science/feature-importance
62. A Comparative Study of Explainable AI Methods: Model-Agnostic vs. Model-Specific Approaches - arXiv, accessed June 5, 2025, https://arxiv.org/html/2504.04276v1
63. proceedings.neurips.cc, accessed June 5, 2025, https://proceedings.neurips.cc/paper_files/paper/2023/file/19ded4cfc36a7feb7fce975393d378fd-Paper-Conference.pdf
64. www.diva-portal.org, accessed June 5, 2025, https://www.diva-portal.org/smash/get/diva2:1752144/FULLTEXT01.pdf
65. NeurIPS Poster Topology-Aware Uncertainty for Image Segmentation, accessed June 5, 2025, https://neurips.cc/virtual/2023/poster/71258
66. Anomaly Detection Techniques: How to Uncover Risks, Identify ..., accessed June 5, 2025,

https://www.mindbridge.ai/blog/anomaly-detection-techniques-how-to-uncover-risks-identify-patterns-and-strengthen-data-integrity/
67. Anomaly Detection in Images and Videos: An In-Depth Exploration, accessed June 5, 2025, https://www.xenonstack.com/blog/anomaly-detection-in-images
68. Anomaly Detection in Medical Imaging - A Mini Review - arXiv, accessed June 5, 2025, https://arxiv.org/html/2108.11986v2