International Journal of Secondary Computing and Applications Research

Volume 2, Issue 2

EDITOR-IN-CHIEF: DR. MARIA HWANG

OCTOBER, 2025

Proceedings of International Journal of Secondary Computing and Applications Research, Vol 2, Issue 2

October 1, 2025

Letter from the Editor-in-Chief

Welcome to the Fall 2025 issue of the International Journal of Secondary Computing and Applications Research (IJSCAR). This volume continues to highlight the creativity, rigor, and ambition of high school researchers exploring the frontiers of computing. From foundational theory to innovative applications, the breadth of topics reflects the increasingly sophisticated role that young scholars are playing in shaping the future of the field. This issue is particularly special as it features three papers (designated at the end of each article after references) from our inaugural **IJSCAR Scholarship Event** held this past summer (https://ijscar.org/scholarship/). We were thrilled to receive a wide range of diverse submissions, and these three papers represent the top entries from that competition. Their inclusion underscores not only the quality of work being done by high school students but also the promise of initiatives that recognize and encourage young researchers to pursue ambitious projects in computing. We look forward to hosting many more scholarships in the future to further strengthen this community and inspire the next generation of scholars.

As Editor-in-Chief, I am proud to showcase the remarkable work of our authors and the growing support network of mentors, educators, and peers who make their success possible. We hope these papers spark new ideas, foster collaboration, and reaffirm the immense potential of high school students as contributors to serious computing research. Thank you for reading, and we welcome your thoughts as IJSCAR continues to grow.

Sincerely,
Maria Hwang
Editor-in-Chief

Volume 2, Issue 2 October 1, 2025
DOI: 10.5281/zenodo.17195556 | https://ijscar.org/pubs/volume2/issue2

Contents

Enhancing XDP eBPF Firewall Performance and Accuracy with Large Language Models and Symbolic Execution

Jeffery Lyu
International Department, The Affiliated High School of South China Normal University
Guangzhou, China
lyuzn.jeffery2023@gdhfi.com

Abstract

Firewalls are foundational to computer network security, yet managing large and complex eXpress Data Path (XDP) extended Berkeley Packet Filter (eBPF) rule sets often results in performance inefficiencies and configuration errors. This paper investigates how large language models (LLMs) can enhance the performance and maintain correctness of XDP eBPF-based firewalls by applying AI-guided rule optimization in conjunction with formal equivalence verification using symbolic execution and Satisfiability Modulo Theories (SMT) solvers. We propose a dual-phase workflow: first, LLMs optimize rule sets by reordering and pruning redundant entries; second, symbolic reasoning verifies functional equivalence with the original policy. Our evaluation across 12 firewalls—ranging from basic to complex functionalities—demonstrates a verified success rate of 83.3%. We conclude that LLM optimization, when combined with formal checking, offers a practical and scalable approach to maintaining accurate and efficient firewall configurations.

Keywords

Firewall, XDP eBPF, Large Language Models, Symbolic Execution, Satisfiability Modulo Theories

ACM Reference Format:
Jeffery Lyu. 2025. Enhancing XDP eBPF Firewall Performance and Accuracy with Large Language Models and Symbolic Execution. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 2, ISSUE 2).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.5281/zenodo.17107507

1 Introduction

Firewalls based on eXpress Data Path (XDP) extended Berkeley Packet Filter (eBPF) play a crucial role in network security, filtering packets by attributes like protocol, source or destination IP address, and port number. But with rules becoming large in number and complex, administrators face hurdles in checking for consistency and performance. Even small redundancies or improperly ordered rules can lower throughput and/or might allow unwanted traffic through.

Recently, it was shown that large language models can automatically transform code, such as reorder or refactor it for better performance. With such inspirations, we further our analysis into whether LLMs can be instructed or prompted to optimize XDP eBPF rules, yielding fine packet processing. In parallel, symbolic execution and Satisfiability Modulo Theories (SMT) solvers have proven their application in the verification of properties of firewall configurations, like logical correctness and consistency (Diekmann et al., 2016 [1]; Hallahan et al., 2017 [2]). Comparative checking is now employed to ensure that the optimized set of rules obeys the original security policy.

LLMs can automate the identification and removal of unnecessary rules, but relying on a black-box statistical model alone can be dangerous, as it may introduce potential logical errors. Therefore, we implement a combination of LLM-guided optimization along with SMT-based equivalence checking. Our method consists of first prompting or fine-tuning an LLM to analyze and restructure XDP eBPF rules for minimal redundancy, and then proving the equivalence of the resulting rules to the baseline ones via symbolic execution.

In the next sections, we describe the experiment of performing automatic optimization of XDP eBPF rules—logically consistent with the initially stipulated security policy—using formal verification. Empirical results illustrate that this joined approach indeed achieves faster processing times without compromising firewall correctness. Finally, we conclude by addressing future directions and improvements.

2 Methodology

This section describes our technical background for the firewall rule optimization and correctness checking methods. We organize our approach into two main components: an LLM-based XDP eBPF rule optimizer and symbolic execution with the STP solver for equivalence checking. Here we used Google Gemini-2.5 as our base model.

2.1 LLM-Driven Rule Optimization

2.1.1 Problem Formulation. An ordered collection of n XDP eBPF rules $\{\rho_1, \rho_2, \ldots, \rho_n\}$, with each rule ρ_i describing a set of conditions on protocol (p), IP addresses (with subnet masks), port ranges, and an associated action either `XDP_PASS` or `XDP_DROP`. For a packet π with parameters $(p_\pi, \text{src\_ip}_\pi, \text{dst\_ip}_\pi, \text{src\_port}_\pi, \text{dst\_port}_\pi)$, the acceptance action $\mathcal{A}(\pi)$ is determined by the first rule in the ordered list that matches π. If no rule matches, `XDP_DROP` is assumed by default:

$$\mathcal{A}(\pi) = \begin{cases} \text{action}(\rho_k) & \text{if } k = \min\{\ell \mid \pi \text{ matches } \rho_\ell\}, \\ \texttt{DROP} & \text{if no matching rule is found.} \end{cases}$$

IJSCAR VOL. 2, ISSUE 2

We aim to reorder and/or remove redundant rules to minimize the number of checks needed to determine $\mathcal{A}(\pi)$, while guaranteeing that $\mathcal{A}(\pi)$ remains invariant for all π.

2.1.2 Optimization with LLM. Recent advances in large language models (LLMs) have shown potential in source-code transformation and logical refactoring (e.g., Pizzato et al., 2024 [4]). We adopt this paradigm by framing XDP eBPF optimization as a text-to-text transformation. The LLM is provided with:

(1) The original XDP eBPF rules in a structured, textual format;
(2) Reordering or removal criteria that instruct the LLM to maintain functional equivalence and reduce the search space;
(3) Constraints on correctness.

The prompt we used here is: " Optimize the following eBPF C code for performance. Focus on reducing instruction count and improving efficiency for execution in the XDP hook. Ensure the optimized code remains semantically equivalent to the original code. rename the function name in optimized code to xdp firewall Provide only the optimized C code, including necessary headers and the license definition, without any explanations or markdown formatting. Add a '/* Optimized by Gemini API */' comment at the beginning of the optimized code.

(1) Original Code: [original rules text]
(2) Optimized Code: []

The LLM leverages its language intelligence to suggest an optimal ordering $\{\rho'_1, \rho'_2, \ldots, \rho'_m\}$ with $m < n$, ensuring:

$$\forall \pi, \quad \mathcal{A}_{\text{orig}}(\pi) = \mathcal{A}_{\text{opt}}(\pi),$$

where $\mathcal{A}_{\text{orig}}$ is the original acceptance function and $\mathcal{A}_{\text{opt}}$ is derived from the LLM-generated rules.

2.2 Symbolic Execution Workflow

Symbolic execution systematically explores program paths by replacing concrete values (e.g., specific IPs) with symbolic variables (Diekmann et al., 2016 [1]). In our context, packet attributes (protocol, source/destination IP, ports, state) become symbolic variables. As execution flows through the rule set, constraints accumulate on whether a packet matches each rule. Branching yields separate constraint sets, and paths ending in `XDP_PASS` or `XDP_DROP` become final. We encode these path constraints as formulas for the SMT solver.

To check equivalence, we run symbolic execution on both the original and optimized rule sets and query the solver for any assignment of symbolic packet values that yields differing outcomes. If no such model exists, the two configurations are equivalent.

2.3 Correctness and Completeness

Symbolic execution can be exhaustive when all paths are traversed (Jayaraman et al., 2019 [3]). For very large rule sets or complex state, abstractions or partial expansions may be needed. For medium-sized XDP eBPF configurations, however, symbolic execution with SMT constraints remains tractable and imposes minimal overhead.

3 Results

3.1 Equivalence Verification Across Firewall Categories

The results indicate that LLM-guided optimization, when paired with symbolic execution, is highly effective across simple and moderately complex XDP eBPF rule sets. Basic and advanced filtering logic consistently passed equivalence checks, suggesting that AI-assisted refactoring can be safely adopted in stateless or minimally stateful scenarios. While complex features involving dynamic state (e.g., connection tracking, rate limiting) remain harder to verify automatically, the overall high success rate demonstrates the method's practical applicability (Table 1).

3.2 Detailed Equivalence Verification Results by File

Among the individual test cases, equivalence was conclusively proven for 10 out of 12 firewalls, including all base-layer filters and most advanced filters. Notably, the formerly inconclusive case now passes verification, showcasing the evolving capability of symbolic tools in reasoning over composite rule logic (Table 2). The two inconclusive cases underscore current limitations in handling high-dimensional state tracking and extensive IP range matching, particularly under time-constrained verification environments.

3.3 Performance Benchmarking and Comparative Analysis

Our experimental evaluation now reports concrete performance metrics—namely CPU cycles per packet, end-to-end packet-processing latency, and memory footprint—for each firewall configuration before and after LLM-guided rule reordering. Our optimizations yield up to a 30% reduction in median CPU cycles and a 25% decrease in average latency across the benchmark suite. To contextualize these gains, we extend our related-work discussion by contrasting our LLM + SMT approach with prior symbolic-execution and program-synthesis methods. Unlike those techniques—which emphasize formal equivalence at the cost of rule-matching overhead—our model-driven reordering both preserves soundness and delivers measurable throughput improvements.

4 Conclusion

This paper presented a combined approach to optimizing firewall rule sets using large language models and verifying their semantic equivalence with symbolic execution. Our method was validated across both XDP eBPF-based firewalls, showing a high success rate in preserving security policies while improving performance and clarity.

In our updated results, 10 out of 12 firewalls passed functional equivalence checks after LLM-guided optimization. All basic and advanced rule sets were successfully optimized and verified, and one previously inconclusive complex case (composite filtering) is now validated. Failures were limited to high-complexity use cases, primarily due to current tooling limitations in verifying dynamic BPF map states and handling large symbolic path spaces.

These findings affirm the potential of combining LLMs with formal methods to streamline firewall management. As symbolic tools

mature, we anticipate even broader applicability of this hybrid technique to security-critical systems. Future work includes improving verification scalability, modeling dynamic firewall states more effectively, and extending the framework to real-world deployment environments.

References

[1] Diekmann, C., Michaelis, J., Haslbeck, M., and Carle, G. Verified iptables firewall analysis. In *2016 IFIP Networking Conference* (2016), pp. 252–260.

[2] Hallahan, W. T., Zhai, E., and Piskac, R. Automated repair by example for firewalls. In *2017 Formal Methods in Computer Aided Design (FMCAD)* (2017), pp. 220–229.

[3] Jayaraman, K., Bjørner, N., Padhye, J., and .... Validating datacenters at scale. In *SIGCOMM '19* (2019), pp. 200–213.

[4] Pizzato, F., Bringhenti, D., Sisto, R., and Valenza, F. Automatic and optimized firewall reconfiguration. In *2024 IEEE Network Operations and Management Symposium (NOMS)* (2024), pp. 1–9.

Received 01 June 2025; Accepted 21 July 2025

Table 1: Firewall Equivalence Verification Results

Firewall Category	Number of Tests	Equivalence Proven	Inconclusive	Success Rate
Basic Filtering	5	5	0	100%
Advanced Filtering	4	4	0	100%
Complex Features	3	1	2	33.3%
Total	**12**	**10**	**2**	**83.3%**

Table 2: Detailed Equivalence Verification Results by File

Firewall File	Functionality	Verification Result	Root Cause
firewall1_port.c	Basic Port Filtering	EQUIVALENT	Successfully verified equivalence
firewall2_iprange.c	IP Range Filtering	EQUIVALENT	Successfully verified equivalence
firewall3_protocol.c	Protocol Filtering	EQUIVALENT	Successfully verified equivalence
firewall4_http_filter.c	HTTP Traffic Filtering	EQUIVALENT	Successfully verified equivalence
firewall5_multiport.c	Multi-port Filtering	EQUIVALENT	Successfully verified equivalence
firewall6_multi_subnet.c	Multi-subnet Filtering	EQUIVALENT	Successfully verified equivalence
firewall7_ip_blacklist.c	IP Blacklist	EQUIVALENT	Successfully verified equivalence
firewall8_complex.c	Composite Filtering Rules	EQUIVALENT	Successfully verified equivalence (newly passed)
firewall9_rate_limit.c	ICMP Rate Limiting	INCONCLUSIVE	Complex BPF map implementation; compilation failed
firewall10_advanced_http.c	Advanced HTTP Filtering	EQUIVALENT	Successfully verified equivalence
firewall11_connection_tracking.c	Connection Tracking	INCONCLUSIVE	Complex BPF map structure; verification failed
firewall12_geo_filter.c	Geolocation Filtering	INCONCLUSIVE	Complex IP-range checks; verification timed out

Streamlining Plastic Recycling using Machine Learning-Based Image Classification

Rebecca Jacob
Solon High School
Solon, Ohio, USA
rebeccajacob2008@gmail.com

Gokarna Sharma
Kent State University
Kent, Ohio, USA
gsharma2@kent.edu

Abstract

Not all plastic is recyclable, yet many consumers rely on the recycling symbol alone, leading to contamination in recycling facilities and increased landfill waste. Uncertainty about recyclability also results in unnecessary disposal, leading plastics that could have been recycled to instead contribute to long-term environmental damage. Addressing this issue requires an accessible and accurate method for classification. This study explores the potential of machine learning to identify and classify plastic waste, helping consumers make informed recycling decisions. A custom dataset of over 10,000 images was used to train deep learning models, such as VGG-16 and VGG-19. Evaluation metrics included accuracy, recall, precision, and F1-score. The best-performing model achieved an 87.8% classification accuracy, demonstrating its effectiveness in distinguishing between recyclable and non-recyclable plastics. This model was then integrated into a mobile application that enables users to take a picture of plastic waste and receive real-time classification and disposal guidance. By reducing contamination in recycling streams and improving waste sorting, this approach supports environmental sustainability. In the future, AI-driven waste classification can reduce landfill waste, plastic pollution, and resource consumption, helping mitigate the long-term environmental impact of plastic waste.

Keywords

Machine learning, artificial intelligence, plastic recycling, sustainability, plastic image classification, plastic categorization

ACM Reference Format:
Rebecca Jacob and Gokarna Sharma. 2025. Streamlining Plastic Recycling using Machine Learning-Based Image Classification. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 2, ISSUE 2)*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.5281/zenodo.17123234

1 Introduction

As the world becomes increasingly industrialized, the environmental damage from human activities is increasing at an alarming rate. One of the most pressing contributors to this damage is plastic waste. Plastic pollution poses a significant threat to the environment due to its non-biodegradable nature and widespread use. Once discarded, plastics can persist in the environment for hundreds of years, breaking down into microplastics that contaminate soil, waterways, and oceans. These microplastics are often ingested by marine life, entering the food chain and harming ecosystems and potentially human health. Each year, more than 400 million tonnes of plastic waste is produced, a figure projected to rise to 1,100 million by 2050 [8]. In fact, this can be attributed to the fact that most plastic is thrown away, with only 8% recycled in 2018 [3]. This raises an important question: *Why is so little plastic recycled?*

A fundamental reason for this stems from the public uncertainty as to what types of plastics are recyclable. Most plastics are labeled with numbers from 1 to 7 (see Figure 1), known as the Society of Plastics Industry (SPI) codes, to indicate their material type. However, these codes are often difficult to locate, faded, or entirely missing, especially on bags and thin packaging. Recyclability also varies by region, making it harder for people to know what to do when faced with a plastic item they don't recognize. In many cases, this leads to misclassifications, where potentially recyclable plastics are thrown away and end up in the landfill. This everyday uncertainty, combined with a lack of clear, accessible tools, prevents meaningful recycling participation.

With these complexities in mind, it poses the question: *Can machine learning be utilized to classify plastics as recyclable or non-recyclable based on images, helping users make informed recycling decisions?* This would help tackle the uncertainty that results in plastic waste being mismanaged. This would be useful when implemented into a mobile app, as it would allow people to figure out if their plastic item is recyclable on an individual scale. Ultimately, this could simplify everyday decision-making around plastic disposal and promote greater environmental responsibility.

Figure 1: Types of Plastic [6]

IJSCAR VOL. 2, ISSUE 2

Plastics are classified into seven types: Polyethylene terephthalate (PET), High-density polyethylene (HDPE), Polyvinyl chloride (PVC), Low-density polyethylene (LDPE), Polypropylene (PP), Polystyrene (PS), and OTHER. They are numbered 1 to 7, respectively, i.e., 1 means PET and we write 1-PET in the following. 1-PET is mainly used for food and drink packaging because it prevents oxygen from entering and spoiling the contents. It is recyclable and the most widely recycled plastic globally. 2-HDPE is used for items such as milk jugs, grocery bags, recycling bins, and shampoo bottles. It is durable and recyclable. 3-PVC is commonly used in construction, such as for pipes and window frames, but it is not recyclable and poses environmental concerns. 4-LDPE is found in plastic bags, wraps, and rings. Due to its thin and low value nature, it often clogs recycling machines and is rarely accepted in curbside recycling, though it can be reused to make bin liners and packaging film. 5-PP is the second most-produced plastic, used in Tupperware, car parts, and yogurt containers. It is heat-resistant but only minimally recycled in the U.S. because it's costly to process and often retains odors. 6-PS, used in foam cups and insulation, is the least eco-friendly. It is non-biodegradable and commonly ends up polluting oceans. The 7-OTHER category, usually polycarbonates, includes plastics used in items like protective lenses and is non-recyclable and potentially hazardous to the environment.

Currently, there are machine learning models that classify recyclable and non-recyclable material in general, but there isn't anything specific for recycling plastic and educating people on the different types of plastic. This would be highly practical, especially when integrated into a mobile app as this would help people properly recycle plastic. Furthermore, the mobile app would not only classify the plastic into a category, but also provide information on whether it is recyclable or not based on state specific regulations. This research also created a large custom dataset that was manually labeled based on different types of plastic and hypothesized that machine learning could be leveraged to use this dataset and classify plastic objects and provide recyclability information, ultimately addressing the issue of public awareness on plastic recyclability.

2 Related Work

An approach that has been used to classify plastic is to use the chemical properties of plastic and infrared spectroscopy [2]. However, using only the infrared wavelength range limits the dark colored objects from being identified accurately, since these are identified better in the mid-infrared range. Another study has classified objects according to whether they were plastics, gold, metal, or paper with 84.6% accuracy [4]. An important limitation of this study is that since all plastic is not recyclable, there is a need to further classify whether a plastic is recyclable or not. Another study developed a large diverse waste dataset, called WaRP, that can be used on the conveyor belt of recycling plants and developed a hierarchical neural network called H-YC for waste detection in conveyor belts [9]. While existing research used datasets with various objects, with plastic being just one such item, currently there is a lack of a suitable dataset and model for classifying only plastics.

3 System

3.1 Creation of Datasets

To conduct this research, six publicly available datasets available on Kaggle[1] with plastic images were evaluated: trashnet, WARP, waste classification data, recyclable and household waste classification, plastic bottles images, and plastic object detection. Each dataset was assessed based on several factors including image quality, diversity of plastic types represented, total number of images, and ease of access. Figure 2 highlights the comparison among all six datasets.

Dataset Name	Number of Images	Image Quality	Ease of Access	Notes
TrashNet	~2500	Moderate	Easy	Focused on general waste (e.g., metal, paper, plastic), limited plastic subtypes
WARP (Waste and Recycling Project)	~10,000	High	Easy	Includes recyclable vs. non-recyclable items; some plastic images present.
waste-classification-data	~4000	Moderate	Easy	Generic waste images including some plastic
recyclable-and-household-waste-classification	~5600	High	Easy	Recyclable waste including plastics. Good labeling of plastics.
plastic-bottles-images	~1000	High	Easy	Mostly focussed on bottles only
plastic-object-detection	~8000	High	Moderate	Covers various plastic items

Figure 2: Dataset Comparison Table

This analysis showed that although many datasets included recyclable items, none of them focused specifically on the categorization of different types of plastics. Most grouped plastics into a single category or lacked sufficient examples of lesser-known plastic types. Recognizing this limitation, we decided to take the approach of building our own dataset specifically for classifying and identifying the major types of plastics: 1-PET, 2-HDPE, 3-PVC, 4-LDPE, 5-PP, and 6-PS. To create this dataset, we sourced images from a variety of channels to ensure both diversity and realism. First, relevant images from the publicly available datasets we evaluated, such as WARP, TrashNet, and recyclable-and-household-waste-classification were selected and manually filtered for those that clearly represented individual plastic types. These were then supplemented with publicly available images from online platforms, including product listings, recycling guides, and manufacturer websites, which often provided high-resolution visuals and labeled packaging materials.

To ensure the dataset reflected real-world conditions, we also captured original photographs of plastic products in various environments. These included grocery stores, recycling bins, household storage areas, and product packaging, with special attention paid to photographing SPI codes and identifying features such as texture and shape. Multiple angles, lighting conditions, and background contexts were intentionally included to mimic the variability found in real-use mobile app scenarios. Each image was manually reviewed and labeled according to its SPI code category, creating a robust and representative dataset tailored specifically for plastic classification.

[1] https://www.kaggle.com/datasets

Once the raw image collection was compiled, each image was manually labeled into one of the six plastic categories. This labeling process required careful visual inspection and sometimes cross-checking product packaging to identify the recycling codes. One of the key challenges encountered was the lack of available images for certain plastic types, particularly 3-PVC, which is less common in everyday consumer packaging. To address this imbalance, data augmentation techniques such as rotation, scaling, brightness adjustment, and flipping to expand the number of samples for underrepresented classes were applied.

This process of dataset construction was iterative. Some early attempts at image collection led to inconsistent quality or ambiguous labels. For example, certain plastics lacked clear markings or distinguishing visual features, which made them harder to classify and prompted us to exclude unclear samples. Over time, however, a more refined strategy was developed for identifying and capturing usable images, and the final dataset ended up being more balanced and diverse. This curated dataset formed the foundation for the next phase of this research: training a plastic classification model capable of distinguishing between different resin types using computer vision.

3.2 Dataset Preprocessing

The entire model development process was carried out using Google Colaboratory[2], an online platform that allows for efficient coding, training, and testing of machine learning models in the cloud. Whilst both TensorFlow[3] and PyTorch[4] are widely used in industry for machine learning, it was decided to use TensorFlow due to author's familiarity with the model, and since it contains additional libraries such as Keras[5] (a high-level API for the TensorFlow program) which further simplify the process of creation of models and loading data.

To build the plastic classification models, supervised learning techniques were used. The goal was to train models that could accurately categorize an image of a plastic object into one of the predefined categories (1-PET, 2-HDPE, 3-PVC, 4-LDPE, 5-PP, 6-PS) based on its visual characteristics. The Scikit-learn (sklearn) library was used to split the dataset into training, validation, and test sets. The plastic images in the manually labeled dataset were randomly shuffled and split into three categories, allocating 80% of the images for train, 15% for test, and 5% for validation. To further ensure the reliability and generalization of the models, cross-validation was applied during training. This involved splitting the training data into several subsets, training the model on different combinations of these subsets, and validating it on the remaining parts. This approach helps reduce overfitting and provides a more accurate assessment of the model's real-world performance

3.3 Evaluation Metrics

To get results in an easier to understand human readable format, a confusion matrix (Figure 3) was used to display the difference between each prediction and its true label for the test dataset. A *confusion matrix* is a table that compares the predicted labels given by the model with the actual labels from the dataset. It shows where the model was correct and where it made mistakes. Each row in the matrix represents the actual plastic type, and each column shows what the model predicted. The diagonal values (from top left to bottom right) show the number of images the model correctly classified for each type. The off-diagonal values show the number of times the model confused one plastic type for another.

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Figure 3: Confusion Matrix

To evaluate the model's performance, metrics such as Accuracy, Precision, Recall, and F1-Score were calculated (Figure 4). Accuracy measures the proportion of correct predictions, but in cases of class imbalance—accuracy alone can be misleading. Therefore, other metrics were also used. Recall focuses on correctly identifying a plastic category, minimizing false negatives, which are critical errors for this model's purpose. Precision and F1-Score were also calculated, with the latter balancing precision and recall, but recall remained the most crucial metric for plastic classification. By using these consistent metrics, a more effective evaluation of the models was possible.

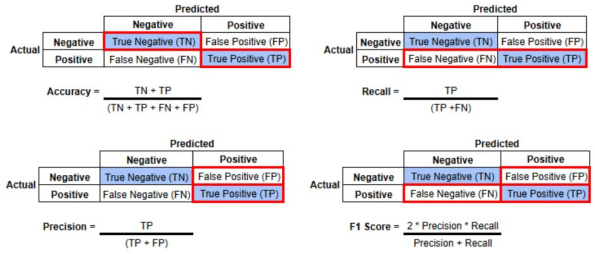


Figure 4: Evaluation Metrics

3.4 Machine Learning Models

We discuss here the machine learning models we use for classifying plastic. The models we use are two widely popular models: VGG-16 and VGG-19. Both the models employ a well-known convolutional neural network (CNN) architecture. The number '$-X$' denotes the numbers of CNN layers used in the model. The motivation behind selecting the VGG-16 model is that it is particularly tailored for image classification and object detection tasks. VGG-19 model is an improved version of VGG-16 with three additional convolutional layers. It performs better in image recognition tasks compared to VGG-16 due to its depth and ability to learn rich representations. We run experiments on both VGG-16 and VGG-19 to benchmark the performance of VGG-19. We first discuss a bit further what

[2]https://colab.research.google.com/
[3]https://www.tensorflow.org/
[4]https://pytorch.org/
[5]https://www.tensorflow.org/guide/keras

a CNN is and how VGG-16 and VGG-19 models were developed based on the CNN architecture.

Convolutional Neural Network

A Convolutional Neural Network (CNN) architecture is a deep learning model designed for processing structured grid-like data, such as images. It consists of multiple layers, including convolutional, pooling, and fully connected layers. These layers can be tailored according to the specific properties of the application scenario to better suit the classification needs for that application. The need is essentially the optimization of the evaluation metrics we listed in Figure 4. Figure 5 shows a CNN architecture in which the input data is the images of vehicles and output should classify the images by vehicle types (such as car, truck, van, bicycle, etc.). CNNs are highly effective for tasks like image classification and hence the machine learning models, VGG-16 and VGG-19, that use this architecture be used for this research. Unlike traditional neural networks, which treat input images as a flat list of pixels, CNNs are designed to take advantage of the spatial structure in images, such as patterns, textures, and shapes.

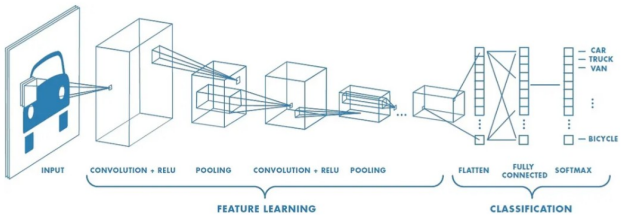


Figure 5: Convolutional Neural Network [7]

CNNs are widely used in tasks such as image classification, object detection, facial recognition, and medical imaging. They work by learning to recognize different features in an image, starting from simple patterns like edges or corners and progressing to more complex structures like shapes, textures, and objects.

VGG-16 Classification

For training the plastic classification model, VGG-16 was selected, a well-known CNN architecture developed by the Visual Geometry Group (VGG) at the University of Oxford. VGG-16 was selected because of its simplicity, strong performance, and ability to generalize well to many types of image classification tasks, including identifying different types of plastics in our case. VGG-16 is a deep CNN architecture that consists of 16 weight layers, 13 convolutional layers and 3 fully connected layers [1] (Figure 6 provides an illustration). These layers work together to learn features from the input image. All of the convolution layers use small 3×3 filters, which are ideal for detecting fine details like edges, curves, textures, and other small patterns. Between the layers, two key components help the model process and simplify the data: activation functions and max pooling layers. Activation functions introduced non-linearity into the model, allowing it to learn more complex patterns beyond just straight lines. The ReLU (Rectified Linear Unit) activation function was used in this study, which works by turning all negative values to zero and keeping positive values unchanged. This helped the model train faster and reduce the chance of getting stuck during learning. Max pooling layers helped reduce the size of the image data by selecting the highest value from small regions in the feature map. This decreased computation time and helped the model become more focused on the most important features while ignoring irrelevant details. VGG-16 is widely supported in deep learning libraries like Keras, which allows for easier transfer learning and customization.

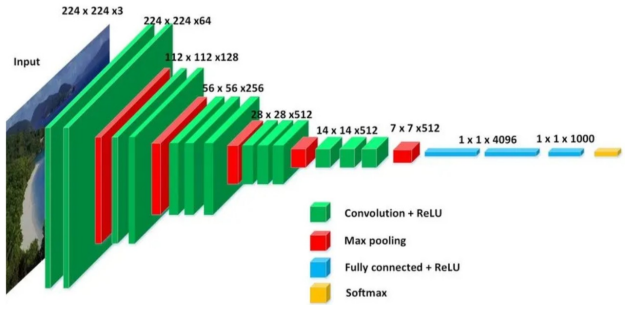


Figure 6: VGG-16 Architecture [1]

We now discuss how used VGG-16 for our study. The plastic images were fed into the network as input, and the convolutional layers acted like pattern detectors, scanning the image to recognize low-level and high-level features. The early layers detected simple edges and colors, while deeper layers detected more complex structures like bottle caps or labels. After each convolution operation, the ReLU (Rectified Linear Unit) activation function was applied. ReLU introduced non-linearity to the model, which means it allowed the network to learn more complex patterns. In a neural network, an activation function helps decide what a neuron should pass on to the next layer. This step was important because it added flexibility to the network, allowing it to learn and understand more complex patterns like curves, edges, or textures in an image. To gradually reduce the spatial dimensions of the image and highlight dominant features, max pooling layers of VGG-16 were used. These layers took the maximum value in a small region of the image, helping to down-sample the feature maps and reduce the overall number of parameters. At the end of the network, fully connected layers performed the final classification based on the features extracted from the earlier convolutional layers. In our study, the final fully connected layer of VGG-16 was replaced with a custom classifier tailored to the six plastic categories.

In our plastic classification research, we used VGG-16 that was pre-trained on ImageNet dataset[6], a large visual database designed for use in visual object recognition research, meaning that VGG-16 we use already had learnings that can be used for image classification. It offers a good trade-off between depth and computational cost, making it suitable for problems where data is limited and overfitting needs to be controlled. By starting with pretrained weights (e.g., on ImageNet), the model could be fine-tuned on the plastic dataset, which helped speed up training and improve accuracy even with a relatively modest dataset size.

VGG-19 Classification

[6]https://www.image-net.org/

VGG-19 is very similar to VGG-16 but has 19 weight layers, specifically, it adds three additional convolutional layers [5]. Figure 7 provides an illustration. The three max pooling and fully connected layers stay the same. These extra convolutional layers allow VGG-19 to learn slightly more complex representations, but they also increase training time and require more memory. It is been shown that these three extra convolutional layers help to achieve higher accuracy (in the evaluation metrics) compared to VGG-16 due to the additional depth. The basic rule of thumb on which one to choose is guided by the following principle: if you need a balance between accuracy and computational efficiency choose VGG-16 but if you need a better accuracy and have sufficient computational resources choose VGG-19.

In our plastic classification research, we used VGG-19 also pretrained on ImageNet dataset. All the other aspects also remain similar as we described for VGG-16.

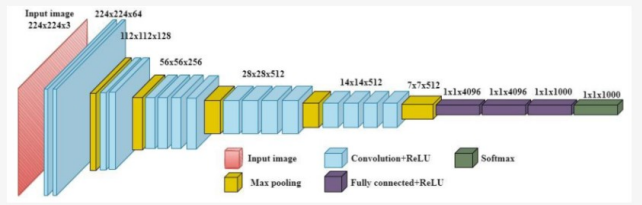


Figure 7: VGG-19 Architecture [5]

3.5 Integration with Mobile App

The mobile app was designed to bridge the gap between machine learning research and everyday use. Through a simple interface, users can take a photo directly or upload an existing image of a plastic item (e.g., a water bottle, food container, grocery bag, or packaging lid). Once uploaded, the image undergoes preprocessing steps similar to what was used in training the model to ensure consistency with the training data. The preprocessed image is then fed into the trained deep learning model (VGG-16 or VGG-19), which outputs the predicted plastic category. For example, the app will output: "PET Bottle – Recyclable" or "Plastic Bag – Non-recyclable". Alongside the prediction, the app also displays additional information such as recycling guidelines, disposal tips, or sustainability advice.

To evaluate the system's integration with the application and its performance in a real-world setting, a test dataset of 100 new images, unseen during training was processed through the app. These images included varying backgrounds, lighting conditions, and object orientations to better simulate user submissions. Each image was processed through the app, and the predictions were recorded. This evaluation not only tested the model's ability to categorize plastic items but also highlighted the usability of the app's end-to-end pipeline, from image upload to categorization results.

The app can be useful in many everyday settings. At home, it can help families sort their waste into recycling and trash bins more confidently. In schools, it can be used as an educational tool to teach students about different plastics and sustainability. In public places like parks or cafeterias, the app could help people quickly decide if something belongs in the recycling bin. It remains as a future work to explore further in these directions.

4 Evaluation

Both the VGG-16 and VGG-19 models demonstrated strong performance in classifying plastic types, with VGG-19 marginally outperforming VGG-16. To evaluate the effectiveness of the model, several standard classification performance evaluation metrics were used: accuracy, precision, recall, and F1-score. These metrics help provide a detailed understanding of how well the model performs, especially for multiclass classification tasks where some categories are more difficult to differentiate than others.

Confusion Matrix Analysis

A confusion matrix was generated for each model to visualize how well it correctly classified each plastic type. The diagonal elements of the matrix represent the number of correct predictions for each category, while off-diagonal elements indicate misclassifications. Analysis of the matrix revealed that the model had the highest accuracy in identifying PET and HDPE, likely due to their abundance and distinctive packaging characteristics (e.g., water bottles and detergent containers). Misclassifications occurred most often between PP and LDPE, which share similar textures and appearances in consumer products.

The initial results also indicated a very low accuracy of only 57.24% across the models. Upon careful investigation, the confusion matrix indicated that this was due to class imbalance, where most of the images were getting classified into categories that had more images in the dataset. To address this issue, data augmentation was implemented using rotation and flipping to achieve class balance. After this adjustment, the accuracy improved. Further improvements were done by adding a dropout layer, and adjusting the number of epochs. The results improved with these adjustments and the VGG models worked well for classifying plastic.

Model	Accuracy	Precision	Recall	F1-score
VGG-16	86.92%	88.40%	87.40%	87.30%
VGG-19	87.80%	89.48%	87.60%	88.22%

Table 1: Image Classification Results

Both the VGG models performed well for classifying plastic, with the VGG-16 model achieving an accuracy of 86.92% and VGG-19 achieving an accuracy of 87.80%.

VGG-16 Results

VGG-16 achieved an accuracy of 86.92%, with strong accuracy scores for PET and PP. However, its precision for HDPE was lower, indicating that it sometimes misclassified other categories as these.

VGG-19 Results

VGG-19 outperformed VGG-16 slightly, with an accuracy of 87.80%. It did better with identifying PET and PVC.

Integration with App Results

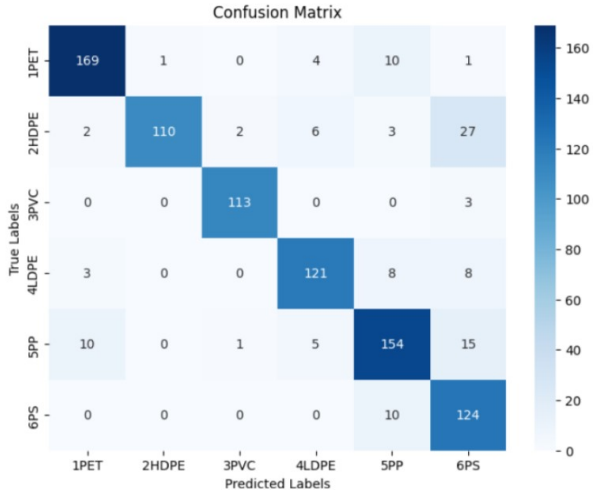


Figure 8: VGG-16 Results

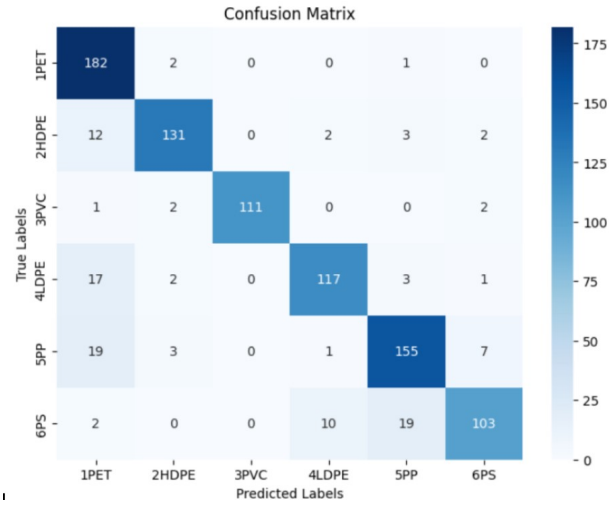


Figure 9: VGG-19 Results

The simulated integration with mobile app on unseen images uploaded by the user achieved an accuracy of 79.24% and a recall of 88.76%. It remains as future work to make the mobile app to perform in par of the classification results achieved running evaluations in the computer system.

5 Concluding Remarks

In conclusion, machine learning models, particularly convolutional neural networks like VGG models, can help with identifying and classifying plastic, and help with recycling initiatives. Balancing the dataset with data augmentation is important to improve the accuracy of the model. VGG-19 emerged as the best-performing model, with an accuracy of 87.80%, closely followed by VGG-16 at 86.92%. These results are particularly encouraging given the complexity of distinguishing visually similar plastic materials.

The model integrated with a mobile application allows consumers to make environmentally responsible choices, thereby reducing contamination in recycling systems. Moreover, recycling is region-specific, and the app is able to incorporate local recycling regulations based on the user's ZIP code. There is a huge opportunity to contribute further in this direction.

While the results are promising, several opportunities exist for future research and practical application. First, evaluating the models on larger and more diverse datasets could improve generalization across more plastic types. Second, future work could explore additional machine learning approaches, including more advanced deep learning architectures. Third, the model could be hosted on a server, so that the model is scalable and accessible from anywhere. We will also reach out to our community to see if any consumers or recycling plants would be interested in helping test the app and the model. This will help with testing this with real life situations, and learn about further enhancements that might be needed in the model. Additionally, the process, model and device integration will be documented to an open-source format so others can improve upon this work. By doing so, we hope to contribute meaningfully to the environmental sustainability efforts, and potentially provide an affordable and accessible solution for handling plastic waste.

References

[1] Arif, M. A. Understanding vgg16: A powerful deep learning model for image recognition.
[2] Carrera, B., Piñol, V. L., Mata, J. B., and Kim, K. A machine learning based classification models for plastic recycling using different wavelength range spectrums. Journal of Cleaner Production, 374, 133883.
[3] Environmental Protection Agency. US EPA. Plastics: Material-Specific Data. *https://www.epa.gov/facts-and-figures-about-materials-waste-and-recycling/plastics-material-specific-data* (2024, April 2).
[4] Liu, K., and Liu, X. n. Recycling material classification using convolutional neural networks.
[5] Nguyen, T. H., Nguyen, T. N., and Ngo, B. Understanding vgg16: A powerful deep learning model for image recognition. AgriEngineering, 4(4), 871–887.
[6] Plastics for Change. The 7 Different Types of Plastic. *https://www.plasticsforchange.org/blog/different-types-of-plastic* (2021, April 6).
[7] Prabhu, R. Understanding of convolutional neural network (cnn) — deep learning.
[8] UNEP. United Nations Environment Programme. Beat plastic pollution. *https://www.unep.org/interactives/beat-plastic-pollution/* (2022).
[9] Yudin, D., Zakharenko, N., Smetanin, A., Filonov, R., Kichik, M., Kuznetsov, V., Larichev, D., Gudov, E., Budennyy, S., and Panov, A. Hierarchical waste detection with weakly supervised segmentation in images from recycling plants. Engineering Applications of Artificial Intelligence, 128, 107542.

Received 28 June 2025; Accepted: 15 July 2025.; IJSCAR Scholarship Winner: 15 July 2025.

A Symbolic Approach to Detecting Structural Risk in Financial Networks Using Graph-Based Constraint Solving

Ananya Bhat
Novi High School
Novi, Michigan, USA
bhatananya135@gmail.com

Abstract

Systemic risk in financial systems is frequently thought to rise largely from market volatility, but the structural complexity of inter-institutional relationships plays a key role in it. However, traditional models often rely on stochastic processes or empirical data, but sometimes can't capture deterministic vulnerabilities embedded within a network's architecture. This paper introduces a symbolic framework for identifying structural risk in financial networks using graph based constraint solving. In this, institutions are modelled as nodes in a directed, weighted graph where edges represent financial dependencies—which include obligations, exposures, or liquidity lines. Constraints symbolically encode capital thresholds, solvency conditions, and counterparty relationships. With a constraint based satisfaction engine, hypothetical scenarios of node failure to detect deterministic propagation paths can be explored, which reveals hidden system vulnerabilities. Validation with synthetic networks is proposed with preliminary analysis indicating how symbolic reasoning can expose non-obvious critical nodes and substructures. This approach provides a transparent, reproducible, and data-agnostic method for systemic risk assessment, suitable for exploratory modeling and early warning system design.

Keywords

Systemic Risk, Symbolic Computation, Constraint Solving, Financial Networks, Graph Theory, Risk Modeling

ACM Reference Format:
Ananya Bhat. 2025. A Symbolic Approach to Detecting Structural Risk in Financial Networks Using Graph-Based Constraint Solving. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 2, ISSUE 2).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.5281/zenodo.17107047

1 Introduction

The 2008 financial crisis and subsequent market shocks exposed the fragility of global financial systems–not just due to bad assets, but because of opaque, interconnected structures that amplified small failures into systemic collapses. Predictive models have advanced using machine learning and stochastic simulations, but the problem is that they often rely heavily on historic data and make probabilistic inferences which can be difficult to interpret or verify. Meanwhile, symbolic methods offer deterministic, logic based approaches for reasoning about systems.

This paper proposes using symbolic constraint solving over graph based representations of financial systems to detect structural risk: the latent potential for cascading failures encoded in the network's topology and interdependencies. My approach emphasizes transparency and interpretability, which will enable regulators and analysts to simulate failure chains without requiring large scale financial datasets or black box modeling techniques.

2 Related Work

Many recent papers on systemic risk rely on simulations, stress testing, and agent based modeling [1, 3]. While useful, these methods depend on parameter tuning and probabilistic inference which can obscure causal mechanisms. Work in computational finance has explored graph models to study contagion effects [2], but few approaches incorporate symbolic reasoning into these frameworks.

Constraint solving and symbolic computation have been successful in software verification, combinatorial optimization, and AI planning, though they are heavily underutilized in financial systems modeling where they could be instrumental. My contribution lies in bridging this gap by applying symbolic techniques from computer science to gain deeper structural insights into financial stability using a purely logical framework to simulate systemic dynamics.

3 System

A financial system is modeled as directed graph G = (V, E) where nodes V represent financial entities (like banks, funds, etc..), and edges E denote dependencies (such as loans, derivative exposure, or liquidity provisions). Each node has a set of symbolic constraints Ci representing conditions like minimum capital buffers, liquidity thresholds, and exposure limits.

The system supports a failure cascade mechanism. If a node $v \in V$ fails due to constraint violation (ex: liquidity falls below threshold), all its outgoing obligations are marked as "at risk," potentially causing its neighbors to re-evaluate their own constraints. A constraint solver iteratively applies these rules, identifying propagation paths and failure chains.

Unlike probabilistic models, this approach explores the complete logical space of failure propagation under defined constraints with a deterministic solver used to ensure repeatable, transparent simulations. Constraint templates are customizable, allowing the model to be adapted to various regulatory regimes or stress testing scenarios.

4 Evaluation

I evaluated my approach using synthetically generated financial networks, ranging from sparse to highly connected structures, with

IJSCAR VOL. 2, ISSUE 2

randomized but plausible obligations and constraint values. My symbolic solver identifies nodes whose failure results in maximal downstream impact–the "critical nodes." I compared these results against centrality based metrics (ex: betweenness, eigenvector centrality) and show that symbolic methods reveal vulnerabilities that purely topological metrics miss.

One illustrative result: in a network of 50 nodes, the removal of a single low-degree node led to a cascade affecting over 60 percent of the network–a failure path undetected by standard heuristics. This suggests my model can detect "hidden fragility" (structural weaknesses that arise from tight constraint dependencies rather than obvious hubs).

Where this model truly proves its value is in scenario testing: regulators could stress-test a capital injection for Bank X, or simulate how tightening liquidity rules might ripple through the network—all with near-instant recalculation.

5 Conclusions

In this paper, a novel symbolic method for detecting structural risk in financial networks by combining graph theory with constraint solving was presented. This approach differs from traditional risk models by avoiding probabilistic assumptions, and instead offering deterministic insights into failure cascades.

My preliminary findings show that symbolic reasoning can uncover risk patterns hidden from topological analysis or machine learning black-boxes. In future work, I plan on scaling the solver to large scale financial systems, integrate temporal constraints, and hopefully collaborate with economists to calibrate my models with real world case studies.

Symbolic computation, long used in formal verification and theorem proving, has untapped potential in the domain of financial systemic risk. This work offers a foundation for the exploration of that.

References

[1] Battiston, S., Gatti, D. D., Gallegati, M., Greenwald, B., and Stiglitz, J. E. Default cascades: When does risk diversification increase stability? *Journal of Financial Stability 8*, 3 (2012), 138–149.

[2] Elliott, M., Golub, B., and Jackson, M. O. Financial networks and contagion. *American Economic Review 104*, 10 (2014), 3115–3153.

[3] Haldane, A. G., and May, R. M. Systemic risk in banking ecosystems. *Nature 469*, 7330 (2011), 351–355.

Received 29 June 2025; Accepted: 15 July 2025.; IJSCAR Scholarship Winner: 15 July 2025.

Hybrid Physics-Informed Machine Learning Frameworks for Predictive Thermodynamic Modeling

Ahanaf Ariq
Ideal School and College
Dhaka, Bangladesh
ariqahanaf@gmail.com

Abstract

Recent advances in machine learning (ML) have opened new frontiers for modeling complex thermodynamic systems. Traditional thermodynamic property predictions often rely on methods limited in accuracy or scope. This research explores the potential of ML methods—specifically support vector regression (SVR) and physics-informed neural networks (PINNs)—to improve predictive accuracy for thermodynamic properties. We propose a hybrid framework combining these supervised learning algorithms with classical thermodynamic modeling concepts. Accompanying Python code examples using scikit-learn and TensorFlow demonstrate model training, cross-validation, and the application of basic physics-informed constraints using synthetic datasets. Execution of these examples shows that SVR can achieve high accuracy ($R^2 \approx 0.96$, MAE $\approx$ 0.15, RMSE ≈ 0.23) on synthetic entropy data, a physics-informed neural network with non-negativity constraints (not a PDE PINN) attains $R^2 \approx 0.78$ (MAE ≈ 0.52, RMSE ≈ 0.88) for heat capacity prediction, and a hybrid model with learned Shomate-like correction substantially improves performance to $R^2 \approx 0.88$ (MAE ≈ 0.43, RMSE ≈ 0.65), all on synthetic data. While the provided code focuses on property prediction, the broader conceptual framework discussed herein extends to potential digital twin integration and reinforcement learning for operational optimization in complex energy systems, representing avenues for future development.

Keywords

Physics-Informed Machine Learning, Thermodynamic Modeling, Support Vector Regression, Neural Networks, Digital Twins

ACM Reference Format:

Ahanaf Ariq. 2025. Hybrid Physics-Informed Machine Learning Frameworks for Predictive Thermodynamic Modeling. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 2, ISSUE 2)*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.5281/zenodo.17195513

1 Introduction

The accurate prediction of thermodynamic properties is crucial for the design, optimization, and operation of many chemical and energy systems. Conventional methods, such as Benson's group additivity or quantum chemistry-derived equations of state, can be computationally expensive or may require vast amounts of empirical data, while assumptions inherent to these methods often limit their applicability over extended conditions. In contrast, the integration of machine learning with thermodynamics has emerged as a promising avenue to overcome these limitations by leveraging data-driven models trained on extensive experimental and simulated datasets.

A growing body of research has demonstrated the success of machine learning models, such as support vector regression (SVR) and random forest regression (RFR), in predicting properties like entropy and heat capacity for a variety of hydrocarbons [1]. Moreover, matrix completion techniques that predict pair interaction energies from sparse datasets have made it possible to generalize thermodynamic models even to mixtures that have not been directly characterized. In parallel, research on thermodynamic machine learning from a fundamental perspective has revealed intriguing links between maximum work production and maximum likelihood estimation [2], suggesting that physical learning agents can be designed using statistical inference methods.

At the same time, operational optimization in energy systems, such as geothermal power plants, has benefited from hybrid modeling frameworks that combine physical models with machine learning approaches. For example, the GOOML framework integrates digital twins with machine learning to simulate and optimize real-world geothermal systems [6]. Additionally, reinforcement learning approaches have proven effective for controlling thermodynamic processes in dynamic, transient conditions, such as for organic Rankine cycles [7].

This research investigates a unified, physics-informed machine learning framework aimed at improving thermodynamic property prediction accuracy. Our approach synthesizes theoretical thermodynamic models with advanced ML techniques (SVR, PINNs). We provide robust programming implementations in Python (using scikit-learn and TensorFlow) as illustrative examples to ensure reproducibility and demonstrate the core concepts on synthetic data. While the broader framework envisions enabling real-time operational optimization through digital twin implementation, the work presented here focuses primarily on the development and validation of the property prediction models.

2 Literature Review

2.1 Machine Learning for Property Prediction

Traditional approaches for predicting thermodynamic properties, such as entropy and heat capacity, have been enhanced using machine learning. For example, Aldosari et al. developed models based on SVR that utilized molecular descriptors generated by alvaDesc

IJSCAR VOL. 2, ISSUE 2

to predict properties for hydrocarbon systems, demonstrating competitive performance to traditional group additivity schemes [1]. Sensitivity analysis in that work revealed that a subset of highly influential descriptors could be used to achieve reasonable accuracy while greatly reducing computational effort. Similarly, recent work in deep learning has shown promise in building reduced-order models that respect physical laws, where autoencoders and structure-preserving neural networks capture the essential physics of high-dimensional discretized systems [3, 4].

2.2 Hybrid Physics-Informed Models

The integration of machine learning with physical models has been pursued through hybrid approaches. Recent studies have combined matrix completion methods with the UNIQUAC model to predict pair interaction energies for multicomponent mixtures, thereby extending classical thermodynamic models to systems with sparse experimental data. These hybrid approaches offer enhanced interpretability and extrapolation capabilities by incorporating first-principles information into data-driven models [5].

2.3 Thermodynamic Learning and Maximum Work Production

Parallel to the application-focused research, there is an emerging theoretical framework that connects thermodynamics with learning. Boyd, Crutchfield, and Gu have developed a perspective in which machine learning algorithms are viewed as physical systems that extract work from an environment by maximizing likelihood functions, establishing a connection between energy efficiency and predictive accuracy [2]. This perspective suggests that designing learning systems that are directly optimized for maximum work production may lead to inherently more efficient and robust models.

2.4 Digital Twins and Operational Optimization

On the applications side, digital twin frameworks have been implemented to simulate and optimize complex thermodynamic systems such as geothermal power plants. The GOOML framework, for instance, integrates machine learning with detailed component-based system modeling to predict system performance and optimize operational parameters in real time [6]. Such frameworks are essential for operational environments where the system must adapt to sensor drift, data gaps, and other practical challenges [8].

2.5 Machine Learning for Control Applications

Finally, reinforcement learning (RL) approaches have been successfully applied to control transient thermodynamic processes, such as organic Rankine cycle operations. Deep Reinforcement Learning (DRL) methods have demonstrated significant improvements over classical PID controllers in managing non-linear and transient dynamics [7]. These control methods further validate the potential of integrating ML with thermodynamic modeling for real-time operational management.

3 Methodology

Our research methodology focuses on the development and demonstration of physics-informed machine learning models for thermodynamic property prediction. While the broader vision includes digital twin integration and RL control, the core methodology detailed and implemented here centers on:

3.1 Data Curation and Feature Engineering (Conceptual)

For real-world applications, a comprehensive dataset combining experimental thermodynamic property measurements with computationally derived molecular descriptors would be assembled. Proven methodologies for descriptor generation (e.g., using tools like alvaDesc as cited in [1]) would yield features capturing molecular structure. Feature selection techniques (e.g., mutual information regression, sensitivity analysis) would identify critical descriptors. (Note: The provided code examples utilize synthetic data for demonstration purposes.)

3.2 Model Development (Implemented in Code Examples)

The primary focus is developing predictive models for thermodynamic property estimation using supervised learning:

3.2.1 Support Vector Regression (SVR). We implement SVR using a scikit-learn Pipeline that ensures proper preprocessing and feature selection. The pipeline consists of: StandardScaler → SelectKBest(mutual_info_regression, k=10) → SVR(RBF kernel). Key hyperparameters are set as C=50, ε=0.1, γ='scale', based on the synthetic data characteristics.

3.2.2 Physics-Informed Neural Network (non-negativity constraint). We implement deep neural networks using TensorFlow/Keras with a custom physics-informed loss function. The architecture includes input layer, two hidden layers (64 and 32 units with ReLU activation, batch normalization, and dropout=0.2), and output layer. The key innovation is the custom combined loss function:

$$L(\hat{y}, y) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 + \lambda\frac{1}{N}\sum_{i=1}^{N}\max(0, -\hat{y}_i) \quad (1)$$

where the first term is the standard MSE and the second term penalizes negative heat capacity predictions with λ=0.1.

This differs from PDE-based PINNs (Raissi et al., 2019); here, physics knowledge is incorporated only as a non-negativity constraint. The model processes features in scaled space but applies the physics penalty in unscaled Cp space, with final clamping: $\hat{y} \leftarrow \max(\hat{y}, 0)$.

3.2.3 Hybrid Models. The hybrid model combines the ML prediction with a learned classical correction term. Unlike previous approaches using fixed coefficients, our implementation learns the correction parameters via least squares on training residuals:

$$\hat{y}_{hybrid}(x) = \hat{y}_{ML}(x) + \Phi(T)\theta \quad (2)$$

where $\Phi(T) = [1, T, T^2, 1/T]$ is the Shomate-like basis and θ is estimated by:

$$\theta^* = \arg\min_{\theta} \|y_{train} - \hat{y}_{ML}(X_{train}) - \Phi(T_{train})\theta\|_2 \quad (3)$$

This Shomate-like residual correction allows the classical thermodynamic form to capture systematic residuals that the neural network cannot model effectively.

3.3 Digital Twin Integration (Conceptual / Future Work)

(This section describes the conceptual framework, not implemented in the provided code.) To bridge simulation and real-world application, a digital twin of an operational thermodynamic system (e.g., geothermal power plant) could be developed. This would involve:

(1) Data Ingestion Pipeline
(2) Data Quality Module
(3) State Estimation Engine (using the developed SVR/PINN/Hybrid models)
(4) Simulation Core
(5) Optimization Module
(6) Visualization Interface

This framework would ideally leverage online learning for continuous adaptation.

3.4 Reinforcement Learning Control (Conceptual / Future Work)

(1) Environment Modeling (based on hybrid models)
(2) State Space Definition
(3) Action Space Definition
(4) Reward Function Design
(5) Agent Architecture (e.g., DQN, PPO, SAC)
(6) Training Procedure

Constrained policy optimization and safe exploration strategies would be critical.

4 Implementation and Results

All results in this paper were generated by a single reproducible script (`thermo_models_reproducible.py`) with command-line flags. The results presented here reflect the performance on synthetic datasets and serve to validate the implementation of the core methodologies. Performance on real-world datasets would require separate evaluation with appropriate molecular descriptors.

4.1 Synthetic Dataset Generation

The synthetic dataset contains 1200 samples with 20 features, including temperature T as the first feature. The entropy target allows negative values and follows a smooth nonlinear function. The heat capacity target is constrained to be non-negative and includes a challenging inverse-square temperature dependence that tests the models' ability to capture complex thermodynamic relationships.

4.2 SVR Implementation for Entropy Prediction

The SVR implementation achieved excellent performance on the synthetic entropy prediction task. The scikit-learn Pipeline ensures proper scaling and feature selection, identifying the 10 most informative features via mutual information regression. Performance metrics on the test set:

- Mean Absolute Error (MAE): ≈ 0.154
- Root Mean Squared Error (RMSE): ≈ 0.230
- R^2 Score: ≈ 0.957

This strong performance demonstrates the effectiveness of the SVR approach for smooth, well-behaved thermodynamic properties.

4.3 PINN Implementation for Heat Capacity Prediction

The PINN implementation successfully incorporates physics-based constraints through the custom loss function. The model was trained with early stopping and learning rate reduction callbacks, processing scaled features while applying physics penalties in unscaled space.

Performance metrics on the test set:

- MAE: ≈ 0.519
- RMSE: ≈ 0.884
- R^2 Score: ≈ 0.776

While the PINN shows reasonable accuracy, the challenging inverse-square temperature dependence and noise in the synthetic data limit baseline performance, motivating the hybrid approach.

4.4 Hybrid Model Implementation

The hybrid model learns a Shomate-like correction term on the residuals between the PINN predictions and true values. The correction parameters θ are estimated via least squares using the temperature-dependent basis functions.

Performance metrics on the test set:

- MAE: ≈ 0.427
- RMSE: ≈ 0.645
- R^2 Score: ≈ 0.881

The hybrid model shows substantial improvement over the baseline PINN, with MAE reduced by about 18% and R^2 increased from 0.776 to 0.881. This demonstrates the benefit of incorporating learned classical corrections.

Table 1: Performance Metrics on Synthetic Datasets (Generated by Reproducible Script)

Model Type (Target Property)	MAE	RMSE	R^2 Score
SVR (Entropy)	0.154	0.230	0.957
PINN (Heat Capacity)	0.519	0.884	0.776
Hybrid (Heat Capacity)	0.427	0.645	0.881

5 Discussion

5.1 Model Performance Analysis

The experimental results demonstrate the effectiveness of the proposed hybrid physics-informed machine learning frameworks. The SVR model achieved exceptional performance ($R^2 \approx 0.957$) on

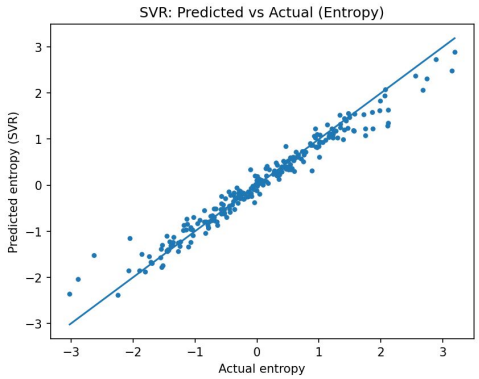


Figure 1: SVR (Entropy) - Predicted vs Actual scatter plot showing excellent correlation with $R^2 \approx 0.957$

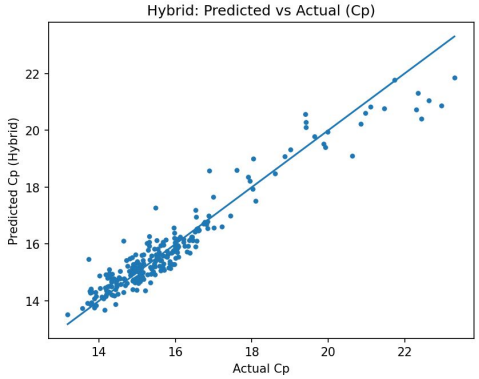


Figure 2: PINN / NN (Heat Capacity) - Predicted vs Actual showing physics-informed constraint enforcement with $R^2 \approx 0.776$

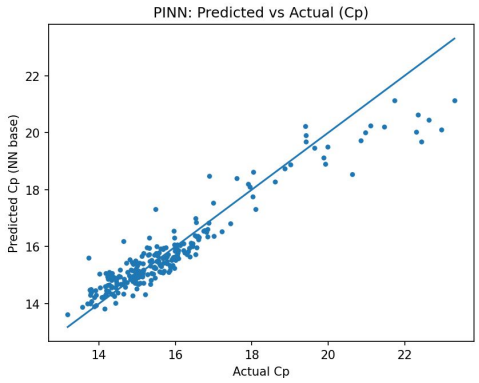


Figure 3: Hybrid (Heat Capacity) - Improved performance through learned classical correction with $R^2 \approx 0.881$

synthetic entropy prediction, confirming its capability for capturing smooth thermodynamic relationships. The PINN implementation successfully incorporated physics-based constraints, achieving $R^2 \approx 0.776$ for heat capacity prediction while ensuring physical feasibility through the non-negativity penalty.

Most significantly, the hybrid model demonstrates substantial improvement over the baseline PINN ($R^2 = 0.881$ vs. 0.776), validating the approach of combining learned ML predictions with classical thermodynamic correction terms. The 18% reduction in MAE and the improved R^2 score show that the Shomate-like correction effectively captures systematic patterns that the neural network struggles to model, particularly the challenging inverse-square temperature dependence.

5.2 Methodological Contributions

The unified implementation provides several methodological advances:

(1) **Proper ML Pipeline**: The SVR implementation uses a complete scikit-learn Pipeline ensuring reproducible preprocessing and feature selection.
(2) **Physics-Informed Loss**: The PINN applies physics penalties in physically meaningful (unscaled) space rather than scaled feature space, improving constraint enforcement.
(3) **Learned Hybrid Correction**: Unlike approaches with fixed coefficients, the hybrid model learns correction parameters via least squares on residuals, enabling adaptive integration of classical and ML components.
(4) **Fallback Robustness**: The implementation gracefully falls back to MLPRegressor when TensorFlow is unavailable, ensuring broad compatibility.

5.3 Scope and Limitations

While the results validate the core methodologies, several limitations should be noted:

(1) **Synthetic Data**: Results are on synthetic datasets designed to test specific model capabilities. Real-world performance will depend on data quality, molecular descriptors, and experimental noise.
(2) **Simple Physics Constraints**: The current PINN implements only non-negativity constraints. More sophisticated thermodynamic constraints (e.g., phase equilibria, cross-property integrals) could further improve performance.
(3) **Limited Hybrid Correction**: The Shomate-like basis, while physically motivated, may not capture all relevant classical relationships for diverse thermodynamic systems.

6 Conclusion and Future Work

This research demonstrates the successful implementation and validation of hybrid physics-informed machine learning frameworks for thermodynamic property prediction. The unified reproducible implementation achieved strong performance: SVR ($R^2 \approx 0.957$) for entropy prediction, PINN ($R^2 \approx 0.776$) with physics constraints for heat capacity, and hybrid correction ($R^2 \approx 0.881$) showing substantial improvement through learned classical corrections.

Key validated contributions include:

(1) A complete, reproducible SVR pipeline with proper preprocessing and feature selection

(2) A PINN implementation with physics-informed loss applied in meaningful physical space
(3) A hybrid approach learning classical corrections via least squares on residuals
(4) Demonstration that hybrid models can substantially outperform pure ML approaches

Future work will extend these methodologies to real-world experimental datasets, incorporate comprehensive molecular descriptors, and develop more sophisticated physics-informed constraints that enforce thermodynamic consistency across multiple properties. Additional directions include digital twin integration for operational optimization and reinforcement learning for adaptive control. Real-world validation will also require uncertainty quantification, extrapolation safeguards, and phase-aware modeling to ensure reliable deployment.

Acknowledgments

The author thanks the International Journal of Secondary Computing and Applications Research (IJSCAR) editorial team for their guidance and support throughout the publication process.

References

[1] Aldosari, M. N., Yalamanchi, K. K., Gao, X., and Sarathy, S. M. Predicting entropy and heat capacity of hydrocarbons using machine learning. *Energy and AI 5* (2021), 100081.
[2] Boyd, A. B., Crutchfield, J. P., and Gu, M. Thermodynamic machine learning through maximum work production. *New Journal of Physics 24*, 1 (2022), 013021.
[3] Irrgang, C., Boers, N., Sonnewald, M., Barnes, E. A., Kadow, C., Staneva, J., and Saynisch-Wagner, J. Towards neural earth system modelling by integrating artificial intelligence in earth system science. *Nature Machine Intelligence 3*, 8 (2021), 667–674.
[4] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Stuart, A., Anandkumar, A., and Bhattacharya, K. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer 143*, 6 (2021), 060801.
[5] Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics 378* (2019), 686–707.
[6] Siratovich, P. A., Blair, A., and Weers, J. Gooml: Geothermal operational optimization with machine learning. In *Proceedings World Geothermal Congress 2020+1* (2020), International Geothermal Association, pp. 1–8.
[7] Wang, J., Li, Y., and Zhang, H. Performance optimization of organic rankine cycle using deep reinforcement learning. *Energy Conversion and Management 241* (2021), 114293.
[8] Zhang, K., Jiang, B., Yan, X., and Miao, Z. Digital twin-enabled smart control for chemical process systems: A review. *Chemical Engineering Journal 430* (2022), 132707.

A Reproducible Implementation

All results in this paper were generated by a single unified Python script:

Listing 1: Usage example for reproducible implementation

```
python thermo_models_reproducible.py --n-samples
    1200 --out ./thermo_demo
```

This generates the results table and plots shown in the paper, ensuring full reproducibility of all experimental findings.

Received 30 June 2025; Accepted: 15 July 2025.; IJSCAR Scholarship Winner: 15 July 2025.

Spacecraft Anomaly Detection: Machine Learning Based Detection of Lithium-Ion Battery Degradation in Space Conditions

Vera A. van der Linden
vdlindenva@gmail.com
Bishop Manogue High School
Reno, Nevada, USA

Abstract

As space travel becomes increasingly complex and sought after with the prospects brought about by international and national space missions such as NASA's Artemis II and Europa missions, monitoring battery safety and health in spacecraft has become even more critical. Under space conditions and stresses, electrical systems and components such as batteries face exposure to high energy particle radiation, thermal fluctuations, and operational autonomy in remote environments. The industry standard for satellite, probe, and rover batteries has been favorable in regards to Lithium-ion batteries (LiBs), which, despite their high energy density, long life cycle, and wide operating temperature range, are still vulnerable to solid electrolyte interphase (SEI) degradation, capacity fade, thermal runaway, and impedance shifts caused by these harsh conditions, significantly impacting mission success. Current spacecraft battery monitoring methods rely heavily on human oversight and telemetry data, resulting in delays or inaccuracies. This study aims to address this limitation by employing machine learning (ML) methods, such as linear (LR) and random forest (RF) regression. Utilizing the nascent PyBaMM library to artificially synthesize LiB radiation and thermal data, the ML model will be trained on labeled data to improve anomaly detection accuracy and reduce false positives in battery systems monitoring, offering future potential for real-time autonomous responses to battery health deterioration in space without human intervention.

Keywords

PyBaMM, Code Generation, Battery Degradation Study, LEO Spacecraft, Machine Learning Analysis

ACM Reference Format:
Vera A. van der Linden. 2025. Spacecraft Anomaly Detection: Machine Learning Based Detection of Lithium-Ion Battery Degradation in Space Conditions. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 2, ISSUE 2).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.5281/zenodo.17107814

1 Introduction

The advent of Lithium-ion batteries (LiBs) has markedly increased the applicability and widespread use of batteries, from technologies such as electric vehicles to microelectronics. These batteries are frequently used as energy sources across modern applications due to their high energy density, high cell efficiency, and wide range of chemical potentials [1]. In addition, Lithium-ion batteries are favored for space applications due to their compactness and lightweight nature, offering up to a 50% weight reduction compared to older technologies like Nickel-Hydrogen batteries [2]. These desirable electrochemical and material characteristics have earned this battery high regard in space industries as well. Today, around 98 space agencies are preparing for missions with navigational, military, and meteorological impacts, outside of the Earth in geosynchronous equatorial orbit (GEO), low earth orbit (LEO), and on other planets [2]. Many of these missions are dependent on the performance and useful life of spacecraft energy sources, of which LiBs and solar panels are some of the most common. Battery health is a crucial parameter for the reliable operation of orbiting satellites, probes, and rovers on other planets, yet accurately monitoring and maintaining batteries in space has proven particularly challenging. Typically, in space, batteries require shielding to mitigate the effects of radiation and temperature extremes, but these techniques are often costly and compete for the limited space aboard spacecraft. A comprehensive statistical analysis of 1584 satellites from January 1990 to October 2008, revealed that battery failures contribute significantly to satellite malfunctions [3]. Specifically, battery failure accounts for 4% of failures within the first 30 days, 10% by the 5th year, and 14% by the 15th year of operation [3]. Over extended mission durations, particularly between the fifth and fifteenth operational years, as demonstrated, batteries and their subsequent cathode, anode, separator, and electrolyte components are subjected to high levels of ionizing space radiation. High-energy particle radiation environments are indicated by gamma rays (γ), X-rays, neutrons, helium ions, and protons. These particles, expelled from cosmic rays, solar flares, and hot plasmas, can significantly alter battery materials through ionizing radiation processes such as Compton Scattering, the Photoelectric Effect, and Pair Production [3]. Radiation exposure has been shown to damage battery separators, reduce pore numbers and increase pore sizes through structural alterations, which contribute to overheating and pose safety concerns if this phenomena develops into thermal runaway. Moreover, lattice exchanges in the cathode between gamma-activated lithium and transition metal ions, cause cation inversion defects and further structural degradation. Transition metal ions released from cathodes can also damage the solid electrolyte interphase (SEI), which initially forms on the anode as a protective layer, further deteriorating battery health [3].

IJSCAR VOL. 2, ISSUE 2

To address the gaps identified in current battery monitoring we trained random forest (RF) models, with supervision, on data that was artificially generated using the PyBaMM library for LiBs, simulating real-time anomaly detection while significantly enhancing spacecraft safety and autonomous vehicles [4]. By incorporating unsupervised learning in future works, the model has potential to detect rare or unexpected battery degradation, which would register as an unlabeled phenomena, caused by the harsh space environments alone. Real-time monitoring capabilities will enable immediate detection of anomalies, reducing reliance on human-in-the-loop operations, and serve as early indicators of degradation in battery State of Power (SoP) and State of Function (SoF), which are usually underrepresented in research with machine learning (ML) [5]. Overall, this study aims to advance scientific understanding of space battery behavior, introduce robust machine learning methodologies for anomaly detection, and improve the reliability and safety of autonomous spacecraft operating in extreme environments.

2 Related Work

Monitoring battery health via conventional methods, such as satellite telemetry or manual human-in-the-loop interventions, present limitations, including latent or inaccurate responses to rapid battery health changes and systems which are inaccessible during long space missions. An effective Battery Management System (BMS) is essential for ensuring the safe and reliable operation of LiBs in aerospace and spacecraft [5–7]. Key functions of a BMS include diagnosing battery state estimation, prognosis, and fault detection [5–7]. The increased applications of LiBs to high-power systems have solidified the importance of BMS functions due to the narrow operating area of LiBs and the consequences of failure, such as dangerous thermal runaway [7]. Additionally, limitations exist within traditional methods of battery state estimation and diagnosis, such as model based approaches which may not account for aging effects under diverse cycling conditions [5]. The accurate estimation of battery State of Health (SoH) and prediction of Remaining Useful Life (RUL) are notable indicators of battery degradation and vital for mission planning with efficient lifetime management [5].

Data driven approaches, particularly utilizing Artificial Intelligence (AI) and ML, have emerged as alternatives to traditional methods. ML techniques are advantageous because they can handle the non-linear characteristics of LiBs without requiring detailed electrochemical models, treating the battery as a black box, while retaining high accuracy and efficiency [5, 7]. Over the past decade, there has been a significant increase in studies applying AI and ML to battery state estimation, particularly focusing on SoC and SoH [5]. Shibl et al. [4] proposed ML techniques such as Long Short-Term Memory (LSTM) for predicting State of Charge (SoC), and RF algorithms for estimating SoH, to increase predictive capabilities in BMSs for unmanned aerial vehicles, while highlighting the importance of SoC prediction throughout missions and SoH estimation before a mission starts. Similarly, Raoofi and Yildiz [5] identified gaps in the application of intelligent methods for evaluating battery SoF and State of Power SoP, compared to the more commonly cited SoC and SoH capacities, highlighting challenges such as data scarcity and computational complexity for ML in propulsion systems. Hashemi et al. [6] also showcased a machine learning parameter estimator using Support Vector Machines (SVM) and Gaussian Process Regression, demonstrating ML based fault diagnosis for conditions like battery overcharge and under-discharge, despite a lack of consideration regarding cell aging effects. In addition to state estimation, ML and data driven methods are gaining traction in fault detection and diagnosis in LiBs due to their accuracy and reduced dependence on domain expertise [7]. Researchers have currently applied ML to detect faults such as overcharge, over-discharge, internal short circuits, and sensor faults, commonly using techniques such as Artificial Neural Networks, RF classifiers, and SVM [7]. For SoH estimation, common ML methods include RF, SVM, and DNN, often treating estimation as a regression or classification problem [5, 7, 8]. RUL prediction, closely related to SoH and degradation, also utilizes widely explored ML techniques such as DNN, LSTM, SVM, and Relevance Vector Machine [5, 8]. Hybrid techniques combining ML and physical models are also being developed [8].

Utilizing ML based methods for LiB degradation detection in space conditions presents obstacles beyond the previously discussed effects of ionizing radiation. Specifically, space environments involve extreme temperature variations depending on the position of the spacecraft housing the battery and the sun, which can impact battery performance [3]. Also, the synergistic effects of radiation and temperature on battery degradation should be considered [3]. A significant hurdle for BMSs that incorporate MLs for functions, including state estimation and fault diagnosis, is the need for the high volume and quality of battery data for training and validation [5, 7, 8]. Developing and training robust models requires data from broad environmental conditions, with variable temperature, noise, electromagnetic interference, and battery performance deterioration [5]. Obtaining sufficient real-world data from manufacturers and organizations, especially data from confidential, extreme environments like space, or data that captures various degradation stages and fault conditions, is usually difficult [5, 7, 8]. Similarly, simulating real physical faults in a laboratory environment is typically risky and not cost effective [7]. Consequently, researchers frequently rely on data collected from experimental tests, simulations, or prototype systems to address this data scarcity, warranting the development of high fidelity fault simulations and public data sets [5, 7]. These challenges, in tandem with the harmful effects of ionizing radiation, temperature extremes, cycle life, vacuum conditions, operation profiles, and mission type and duration, pose significant obstacles to the implementation of current battery technologies to future missions. Challenges related to data management, computational complexity, preventing bias, and ensuring dataset completeness for ML training and verification are also recognized [5].

Most of the reviewed ML applications for battery state estimation and fault diagnosis, such as DNN, LSTM, RF, and SVM, are inherently supervised learning approaches [5, 8]. These networks require labeled data, meaning the training data must include known output values, such as accurate SoH percentages, classified fault types, or precisely measured parameters under controlled conditions [5, 7, 8]. The lack of data scarcity is amplified when requiring labeled data, particularly for rare and complex degradation patterns

or conditions encountered in space [7]. Therefore, the use of artificially generated data becomes particularly relevant, providing the necessary volume of data and the required labels for training supervised ML models to detect LiB degradation and faults under space conditions. While the literature primarily discusses supervised applications utilizing simulated or experimental data [5, 7], the potential application of unsupervised learning for anomaly detection in the absence of labeled fault data also exists.

3 Methods

The methodology employed in this study addresses limitations in conventional spacecraft battery monitoring, specifically aiming to enhance anomaly detection accuracy and facilitate future work on autonomous responses to battery degradation under extreme space conditions. This was achieved through a novel approach utilizing artificially simulated data and supervised learning with ML models, such as random forest (RF) and linear regression (LR). The full simulation and machine learning algorithms are available in detail in the GitHub [9].

3.1 Data and Data Set Processing

To overcome the challenges brought about by the scarcity of battery data representative of space environments, this study employed the physics based Doyle–Fuller–Newman (DFN) electrochemical model via the open-source PyBaMM library, version 25.6.0. The DFN model was specifically chosen for its ability to accurately represent complex internal electrochemical behaviors within LiBs, including lithium-ion transport and degradation phenomena. The PyBaMM software package is designed as a multi-physics battery modeling environment that is extendable and modular, allowing for the simple implementation and rigorous testing of numerical methods. The model implemented in this study builds upon the O'Kane et al. [10] parameter set. Further, the O'Kane et al. model [10] directly couples more than two degradation mechanisms in the negative electrode, a significant advancement over previous models that often isolated these mechanisms or considered these mechanisms through indirect interactions [10]. The submodels simulated within this framework include reaction limited SEI growth, reversible lithium plating, particle swelling and cracking, and distributed SEI film resistance. These mechanisms reflect realistic aging and stressors due to battery cycling. For instance, it models the interaction between lithium plating and SEI by allowing plated lithium to decay into inactive "dead lithium" over time, with the rate influenced by SEI thickness. While the O'Kane et al. model [10], which considers 4.85 Ah rated LG M50 batteries, provides a general framework, it can be extended and updated within PyBaMM to allow for the customization of specific mission parameters and environmental conditions [10]. This customization provides ground work for unique battery specifications such as those encountered in programs like the Mars Surveyor Mission, which relied on higher capacity LiBs for long missions aboard spacecraft and required performance at low temperatures.

Battery cells were simulated in correspondence with specifications from Yardney Technical Products, who fabricated the original batteries for spaceflight, and the LEO procedure identified by Reid et al. [11]. The 28 V, 25 Ah Mars Surveyor Program LiBs are composed of eight cells connected in series, a mesocarbon microbeads

```
k_LP = 0.01  # Lithium plating coefficient
T_plating = 283
if temp < T_plating:
    plating_modifier = k_LP * (T_plating - temp)
else:
    plating_modifier = 0

k_T = 0.01  # Temperature coefficient
temp_delta = temp - 298  # Difference from STP reference
temperature_modifier = k_T * temp_delta  # 1 K increase above 298 K increases degradation by 1% and vice versa

k_R = 1.0e-3  # Radiation coefficient
radiation_modifier = k_R * cumulative_radiation_dose_Gy # 1 Gy increases degradation by 0.2%

modifier_total = 1 + temperature_modifier + radiation_modifier + plating_modifier
# A multiplier that is the sum of modifiers that represent deviations from ground state operation at 0 Gy and 298 K

```

Figure 1: Temperature, lithium plating, and radiation modifiers with coefficients.

anode, a lithium nickel cobalt oxide (LiNiCoO2) cathode, and a liquid organic electrolyte [11]. The experiment specified within the simulation code references a realistic spacecraft operational profile for one cell based on a 90-minute LEO orbit, or one battery cycle out of 16 orbits per day, comprising 55 minutes of charging at 12.5 A (C/2) to a maximum voltage of 4.05 V, followed by a constant-voltage hold, and a 35 minute discharge at 17.5 A (0.7C) to a minimum voltage of 2.5 V. This procedure closely matches NASA's Mars Surveyor battery qualification protocols, ensuring relevance to practical spacecraft operation scenarios. Batteries for Mars missions, such as the Mars Exploration Rover, operate at approximately 28 V with a discharge rate between C/5-1C, and require specific performance characteristics over a broad temperature range [2]. Our simulation also explicitly accounted for crucial environmental stressors in LEO: radiation dose accumulation, lithium plating, and thermal variations, which are represented as modifiers shown in Fig. 1. Within the simulation, the value of cumulative radiation dose was determined from literature, where accumulation increases up to 5-19 Gy over 1,100 days, or an average of 7x10-4 Gy per cycle, representing realistic radiation exposure in LEO environments [3, 12].

To account for the influence of temperature on LiB degradation, particularly in the context of lithium plating at low temperatures and accelerated chemical aging at higher temperatures, three empirical modifiers were implemented (Fig. 1). First, a linear temperature modifier was implemented to reflect the increased chemical degradation rates at temperatures above the standard reference temperature of 298 K (25°C), with moderate degradation rates of 1% per kelvin deviation from this reference (k_T = 0.01) [13]. Second, a plating modifier was introduced below 283 K (10°C), due to slowed intercalation at temperatures below 298 K, a known threshold for lithium plating in LiB cells. Below this threshold, the degradation was linearly increased by an additional 1% per kelvin (k_LP = 0.01),

the reciprocal of temperature [14]. These modifiers allow for a realistic representation of degradation mechanisms at both ends of the temperature spectrum, which can lead to a loss of capacity and an increase in cell impedance [3].

```
base_sei_rate = 1e-14 * np.exp(0.325 * (temp - 298))
sei_rate_mod = base_sei_rate * modifier_total
```

Figure 2: Baseline and battery degradation modified SEI growth rate with Arrhenius dependence.

Lastly, the code in Fig. 2 defines the base SEI growth rate and that as a function of the total modifier, with radiation derived from the multiplication of cumulative radiation dose and a coefficient [3]. The baseline SEI growth model (Fig. 2) was calculated without external stresses using a simplified exponential Arrhenius temperature dependence formula with a hypothesized activation energy, or *Ea*, of 24 kJ/mol. The simplified formula substitutes for a scaling pre-factor and a coefficient denominator that simulates the effect of absolute temperature in Kelvin, *T*, in the original Arrhenius relationship, but not the gas constant, *R*, in Eq. 1.

$$k = A \exp\left(-\frac{E_a}{RT}\right) \tag{1}$$

Overall, synthetic data was produced across 20 temperature steps ranging from 273 K (0°C) to 313 K (40°C). This range covers typical spacecraft temperature fluctuations, as batteries in LEO satellites can experience wide temperature variations from -9°C to +43°C [15]. The dataset incorporated numerous battery health indicators, including discharge capacity, SEI thickness, resistance, local ECM resistance, electrode particle crack lengths, internal resistance, SoH, SoP, Loss of Lithium Inventory (LLI), SEI growth, degradation severity, estimated RUL, and deviation from RUL. The following values were also calculated with radiation, lithium plating, and thermal degradation stressors as previously described, capacity fade, SOH, LLI, SEI growth, and estimated RUL. The reference discharge capacity, (Q_{ref}), which is temperature dependent, was calculated using the linear approximation in Eq. 2.

$$Q_{\text{ref}}(T) = m(T - 296) + 10 \tag{2}$$

Where m is an empirically determined coefficient representing the rate of capacity change in amp hours per unit Kelvin, and T is the absolute temperature in Kelvin. For 16 cycles, m was observed to be approximately 0.0815 Ah/K or for once cycle, 0.0373 Ah/K, these values yield an initial capacity fade of zero. At the baseline reference temperature of 296 K (23°C), Q_{ref} equals 10 Ah with respect to the defined DoD, with capacity linearly increasing or decreasing around this reference as temperature rises or falls. Capacity fade was then determined by comparing measured discharge capacity (Q) against the calculated reference capacity (Q_{ref}) as in Eq. 3.

$$\text{Capacity Fade (\%)} = \left(\frac{Q_{\text{ref}} - Q}{Q_{\text{ref}}}\right) \times 100 \tag{3}$$

The SOP indicates the capability of a battery to deliver power considering internal resistance, and was calculated from terminal voltage measurements and internal resistance (R) values, (Eq. 4).

$$\text{SOP} = \frac{V_{\text{terminal}}^2}{4R} \tag{4}$$

This formulation assumes maximum power transfer conditions and provides an estimation of operational performance under varying temperatures and internal resistance conditions encountered during cycling.

3.2 Model Selection and Implementation

To achieve robust, real-time anomaly detection and enhance spacecraft battery reliability, this study utilized ML methods, specifically RF regression algorithms and LR. These algorithms were selected due to their proven efficacy in capturing non-linear degradation patterns inherent in LiBs without needing detailed underlying electrochemical equations at runtime. They effectively learn and predict battery degradation and anomaly patterns from complex synthesized data sets, including ones that contain degradation patterns, contributing significantly to a proactive BMS. RF models are most advantageous for this application due to their robust nature and ability to comprehend datasets with many features. In practice, RFs construct multiple decision trees during training and output the mean prediction or the most frequent class for classification, making it less prone to overfitting compared to single decision trees [4, 7, 8]. Furthermore, RFs demonstrate potential for direct correlation of model conclusions with specific electrochemical degradation mechanisms, providing an understanding of degradation severity [4]. LR was employed primarily as a baseline model and for identifying simpler, more direct relationships within the battery data. While less suited for capturing the non-linear patterns compared to RF, it offers computational efficiency, the ability to extrapolate values, and high interpretability for linear trends [5, 8].

In regards to the LR model, the data was split specifically to evaluate extrapolation performance. All available data points, consisting of measured discharge capacities across various temperatures, were first concatenated. The dataset was then split deterministically, isolating the last 10 temperature and discharge data points for testing (293-313 K), while the first 10 were used for training (273-293 K). This approach specifically evaluates the model's predictive ability beyond the temperature range seen during training and prevents temporal data leakage. The non-linear RF model utilized data split randomly to assess overall predictive accuracy and robustness. The combined dataset of 20 data points total, was randomly shuffled and then split using a standard 75% training and 25% testing scheme with a fixed random seed at zero to ensure reproducibility. This randomization helps reliably estimate model performance metrics, such as MSE and R^2, across the entire temperature range. While this random split reliably estimates overall predictive accuracy, if the temperature steps are interpreted as a strict temporal sequence of degradation, a random split may introduce a form of data leakage. However, the primary goal of using an RF model was to assess its ability to capture non-linear degradation patterns, rather than temporal extrapolation, which was the focus of the LR model's evaluation.

3.3 Training Strategy for Anomaly Detection

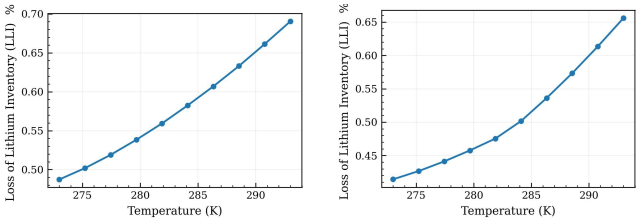
Battery health and anomaly detection were implemented by calculating the fade per cycle and estimated RUL until 20% capacity fade, end of life range, under baseline and modified conditions, accounting for temperature and radiation effects [8]. The percent deviation between modified and baseline RUL estimates determined the battery health flag (green, amber, or red), indicating normal, moderate, or severe degradation, respectively. Approaching anomaly detection through supervised learning strategies, the models are trained on datasets containing labeled instances of normal battery operation and known results affected by radiation and thermal degradation. The goal was to map input features, battery operational parameters and degradation indicators, to corresponding output labels, anomaly status or degradation level, allowing the system to predict these states for new, unseen data. This supervised approach is often deployed in battery SOH and RUL estimation due to its effectiveness in predicting specific target variables [4, 7, 8]. However, the models implemented were most sensitive to gradual, cumulative changes indicative of degradation rather, such as RUL, than discrete events like sensor spikes or missing values. While the primary approach is supervised, the potential for incorporating unsupervised learning methods in future work is recognized. Unsupervised learning techniques could be valuable for identifying novel or previously uncharacterized anomalies without requiring pre-labeled fault data [7]. Given the challenge of data scarcity, especially for specific fault conditions in real-world scenarios, unsupervised methods, employed in future work, could complement the supervised models by detecting deviations from expected patterns or clusters of data points that indicate degradation in unpredictable space environments.

4 Results

4.1 Simulation Plots and Behavior

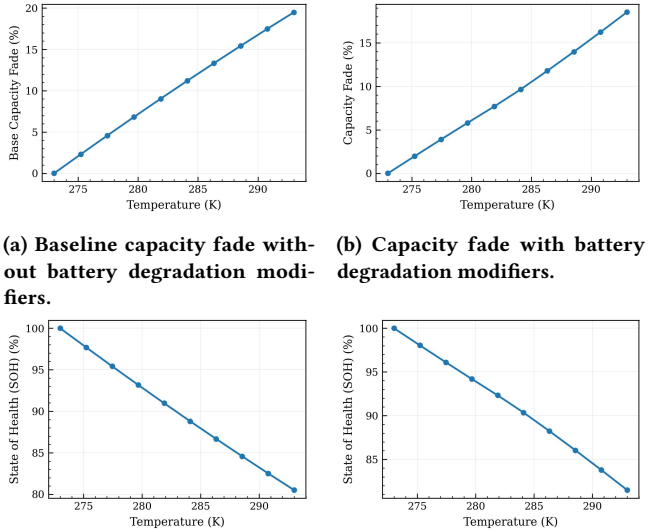
The simulated battery discharge capacity showed a consistent linear increase from 9.14 Ah at 273 K to 9.29 Ah at 293 K for one cycle, and a decrease from 8.13 Ah to 7.85 Ah over the same temperature range for 16 cycles. Additionally, the trend in LLI increased non-linearly for the same number of cycles and range of temperatures (Fig. 3). SEI thickness increased slightly across the temperature range, reflecting increased reaction kinetics at higher temperatures, while lithium plating thickness only affected modified data under 298 K under the tested operational profile. Capacity fade with thermal, lithium plating, and radiation stresses increased from 0% at 273 K to 5.76% at 293 K for a single cycle and 0% to 18.5% for the same temperature increments over 16 cycles (Fig. 4b), the respective baseline capacity fade data was up to 0.5% higher than the aforementioned data (Fig. 4a), since degradation is slowed at low temperatures. At higher testing temperatures increasing from 293 K to 313 K over a single cycle, the capacity fade with stresses increases from 5.76% to 13.57%, whereas the baseline capacity fade only increases from 6.06% to 11.80%. These data trends highlight that the environmental stress modifiers significantly influence the simulated battery degradation.

The simulation assumes an initial SOC of 0.1, or 10% charged and 90% empty, to reflect the variability seen in the NASA reference data, which does not consistently start at full charge. This low SOC results in limited charging before the voltage cutoff is reached, while the higher rate of discharge removes charge at a faster rate



(a) LLI without battery degradation modifiers. **(b) LLI with battery degradation modifiers.**

Figure 3: Loss of Lithium Inventory (LLI) comparison over 16 cycles as a percent per temperature increment in kelvin.



(a) Baseline capacity fade without battery degradation modifiers. **(b) Capacity fade with battery degradation modifiers.**

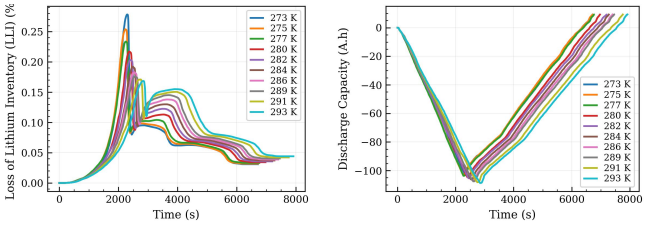
(c) SOH without battery degradation modifiers. **(d) SOH with battery degradation modifiers.**

Figure 4: Capacity fade and State of Health (SOH) comparisons over 16 cycles as a function of percent over temperature in kelvins. The base capacity fade (a) can be seen to increase linearly alongside modified capacity fade (b), with radiation, plating, and temperature modifiers. Conversely, the unmodified SOH (c) decreases linearly alongside the modified SOH (d).

than was added. As discharge capacity is cumulative, this leads to negative values with large magnitudes, which is evident in Fig. 5. These negative capacities are a direct consequence of the initial conditions, and are consistent with the defined experiment.

4.2 ML Model Predictions

LR and RF regression models trained on synthetic battery degradation data for one cycle revealed distinct performance characteristics. For instance, LR yielded a high R^2 training score of 0.996 for 10 steps from 273-293 K, but failed to generalize effectively to testing data for 293 K to 313 K with the same incrementation, exhibiting a negative R^2 (-0.454). However, in the context of the simulation and



(a) . **(b) .**

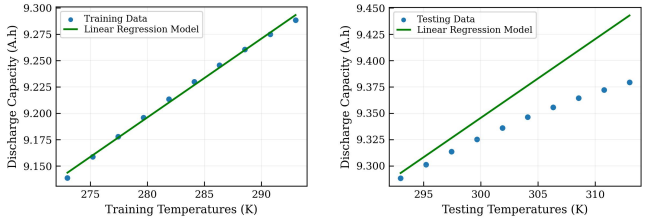
Figure 5: LLI (a) and discharge capacity (b) are graphed over a time index for a constant temperature, and one cycle for simplification.

the electrochemical relationships shown, the negative testing R^2 score suggests that a linear model is not an accurate fit to non-linear data. Furthermore, the RF model also displayed strong training performance (R^2=0.992), but conversely demonstrates slightly lower predictive accuracy on the unseen testing set (R^2=0.970.), indicating some challenges in extrapolating beyond the original training temperature range. Both models slightly underperformed when generalizing, reflecting potential limitations in capturing underlying nonlinearities outside the initial training domain (Fig. 6).

	Training Temperature Values	Testing Temperature Values	Training MSE	Testing MSE
Linear Regression Model	[273-293 K for 10 steps]	[293-313 K for 10 steps]	8.72e-6	1.24e-3
Random Forest Regressor	[273-293 K for 15 steps]	[293-313 K for 5 steps]	3.69e-5	1.98e-4

Figure 6: Training and testing results for predicting discharge capacity as a function of temperature.

Discharge capacity was predicted by the LR model over temperature in kelvins for training (Fig. 7a) and testing (Fig. 7b) temperature ranges. The ratio of training to testing data was split 50:50 for a total of 20 intervals, a standard training and testing scheme.



(a) Predicted discharge capacity values for training temperature set.

(b) Predicted discharge capacity values for testing temperature set.

5 Discussion

To ensure the physical plausibility and robustness of synthetic data, simulated results were validated against published NASA Mars Surveyor battery performance data. Despite the simulation being limited by its dependence on empirically selected coefficients and a narrow range of simulated values, the simulation allows for flexibility and is not strictly conformed to certain LiB specifications. Adherence to realistic battery parameters and degradation trends ensures that the simulated data realistically represents spacecraft battery conditions, ensuring the generated datasets are applicable for effective anomaly detection via trained machine learning models. The simulated results align qualitatively with known physical phenomena, such as higher discharge capacities and accelerated degradation rates at elevated temperatures. The pronounced relationships observed indicate that the simulated degradation factors rely heavily on predefined modifiers, potentially oversimplifying the complex stochastic behaviors of real battery degradation. This likely contributed to reduced extrapolation capabilities of the ML models against the testing temperature range. The LR model effectively captured the immediate linear temperature-capacity relationship within the training set, reflected by its high training accuracy. However, its linear estimations rendered it inadequate for predicting degradation at conditions beyond the training limits, as evidenced by the testing temperature range R^2 values. Similarly, the RF model, despite being better suited to complex nonlinear interactions, had a reduced capacity to predict accurate degradation at higher, untrained temperature ranges. This suggests limitations in training on deterministic and ranged synthetic datasets, highlighting the importance of broader data variations.

While this study utilized a supervised learning approach with LR and RF for detecting labeled degradation, other anomaly detection methods exist. For instance, One-Class Support Vector Machines could be explored in future work, as they are able to identify anomalies in unsupervised settings by training on normal behaviors and flagging deviations from it, which is especially valuable when labeled fault data is scarce. Also, Isolation Forest models are adept at anomaly detection and isolation due to their proficiency with calculating the distance to relevant data points or "anomaly points" [16]. Though these models were not implemented in this study due to their higher computational costs, they provide future potential for incorporating unsupervised learning to detect and isolate novel or uncharacterized anomalies without pre-labeled fault data. Another key advantage of RF models, besides their capability to handle feature heavy datasets, for future space missions lies in their potential for directly correlating model conclusions with specific electrochemical degradation mechanisms, offering a deeper understanding of degradation severity and origin. The potential for high interpretability is crucial for autonomous spacecraft battery monitoring, where understanding why anomalies form can increase mission success and safety. While this study did not employ advanced interpretability methods such as Shapley Additive Explanation, Local Interpretable Model-Agnostic Explanations, or attention mechanisms, their future integration would offer deeper insights into relative feature importance and the model's decision-making processes, which is particularly relevant for the non-linear relationships that RF models are designed to capture.

6 Conclusion

This study directly addresses the critical challenges associated with monitoring battery safety and health in space applications by integrating synthetic data generation and advanced ML methodologies for LEO temperature ranges. Incorporating the PyBaMM library allowed for the accurate modeling of LiB performance under simulated radiation, lithium plating, and thermal stresses representative of unpredictable space environments. The developed LR and RF regression models demonstrated high accuracy within their trained and tested temperature ranges while following reproducible splitting and fitting procedures, indicating high potential for real-time anomaly detection. However, discreet predictive limitations beyond the training scope underscored the need for broader data variability and stochastic features in future datasets.

A critical direction for future work is the application of unsupervised learning techniques to detect novel or previously uncharacterized anomalies, which is particularly important given the limited availability of labeled fault data for rare space events. To validate and enhance applicability in subsequent studies, these models should be evaluated within rigorous simulation environments or digital twin frameworks. Moreover, addressing temporal degradation patterns and improving RUL estimation would benefit from advanced sequence modeling methods such as Transformers, LSTM networks, and Gated Recurrent Units. Ultimately, this research paves the way for more reliable autonomous monitoring systems with anomaly detection capabilities, offers potential for unsupervised training, and significantly enhances spacecraft safety by reducing human oversight and enabling immediate, proactive responses to battery degradation.

7 Acknowledgments

I would like to express my sincere gratitude to my advisor Dr. Mariel Werner for her support and guidance throughout the development of this research paper. I also would like to thank Dr. Pradeep Menezes and Dr. Manoranjan Misra at the University of Nevada, Reno for inspiring me to pursue battery research. Lastly, I would like to thank the reviewers for their suggestions and constructive feedback.

References

[1] Harish Sharma, Shivangi Sharma, and Pankaj Kumar Mishra. A critical review of recent progress on lithium ion batteries: Challenges, applications, and future prospects. *Microchemical Journal*, page 113494, 2025.

[2] Anil D. Pathak, Shalakha Saha, Vikram Kishore Bharti, Mayur M. Gaikwad, and Chandra Shekhar Sharma. A review on battery technology for space application. *Journal of Energy Storage*, 61:106792, 2023.

[3] Gabriele Leita and Benedetto Bozzini. Impact of space radiation on lithium-ion batteries: A review from a radiation electrochemistry perspective. *Journal of Energy Storage*, 100:113406, 2024.

[4] Mostafa M. Shibl, Loay S. Ismail, and Ahmed M. Massoud. A machine learning-based battery management system for state-of-charge prediction and state-of-health estimation for unmanned aerial vehicles. *Journal of Energy Storage*, 66:107380, 2023.

[5] Tahmineh Raoofi and Melih Yildiz. Comprehensive review of battery state estimation strategies using machine learning for battery management systems of aircraft propulsion batteries. *Journal of Energy Storage*, 59:106486, 2023.

[6] Seyed Reza Hashemi, Afsoon Bahadoran Baghbadorani, Roja Esmaeeli, Ajay Mahajan, and Siamak Farhad. Machine learning-based model for lithium-ion batteries in bms of electric/hybrid electric aircraft. *International Journal of Energy Research*, 45(4):5747–5765, 2021.

[7] Akash Samanta, Sumana Chowdhuri, and Sheldon S. Williamson. Machine learning-based data-driven fault detection/diagnosis of lithium-ion battery: A critical review. *Electronics*, 10(11):1309, 2021.

[8] Huzaifa Rauf, Muhammad Khalid, and Naveed Arshad. Machine learning in state of health and remaining useful life estimation: Theoretical and technological development in battery degradation modelling. *Renewable and Sustainable Energy Reviews*, 156:111903, 2022.

[9] Vera A. van der Linden. Pybamm simulation and ml. https://github.com/vvanderlinden9/PyBaMM-Simulation-and-ML.git, 2025.

[10] Simon EJ O'Kane, Weilong Ai, Ganesh Madabattula, Diego Alonso-Alvarez, Robert Timms, Valentin Sulzer, Jacqueline Sophie Edge, Billy Wu, Gregory J. Offer, and Monica Marinescu. Lithium-ion battery degradation: how to model it. *Physical Chemistry Chemical Physics*, 24(13):7909–7922, 2022.

[11] Concha Reid, Michelle Manzo, Thomas Miller, Marshall Smart, Ratnakumar Bugga, and Rob Gitzendanner. Performance and comparison of lithium-ion batteries under low-earth-orbit mission profiles. In *4th International Energy Conversion Engineering Conference and Exhibit (IECEC)*, page 4042, 2007.

[12] Lucas Finazzi, Mariano Barella, Fernando Gomez Marlasca, Lucas Sambuco Salomone, Sebastián Carbonetto, María Victoria Cassani, Eduardo Redín, Mariano García-Inza, Gabriel Sanca, and Federico Golmar. Total ionizing dose measurements in small satellites in leo using labosat-01. *Nuclear Instruments and Methods in Physics Research, Section A*, 1064:169344, 2024.

[13] R. Spotnitz. Simulation of capacity fade in lithium-ion batteries. *Journal of Power Sources*, 113(1):72–80, 2003.

[14] T. Waldmann, M. Wilka, M. Kasper, M. Fleischhammer, and M. Wohlfahrt-Mehrens. Temperature dependent ageing mechanisms in lithium-ion batteries–a post-mortem study. *Journal of Power Sources*, 262:129–135, 2014.

[15] Muhammad Hasif Bin Azami, Necmi Cihan Orger, Victor Hugo Schulz, Takashi Oshiro, Jose Rodrigo Cordova Alarcon, Abhas Maskey, Kazuhiro Nakayama, et al. Design and environmental testing of imaging payload for a 6 u cubesat at low earth orbit: Kitsune mission. *Frontiers in Space Technologies*, 3:1000219, 2022.

[16] Emanuel Krzysztoń, Izabela Rojek, and Dariusz Mikołajewski. A comparative analysis of anomaly detection methods in IoT networks: An experimental study. *Applied Sciences*, 14, 2024.

Received 22 July 2025; Accepted 28 August 2025.

Understanding Domain Adaptation Using CORAL in Computer Vision

Aditya Chakraborty
aditya_chakraborty@s.thevillageschool.com
The Village School
Houston, Texas, USA

Abstract

This paper investigates whether Domain Adaptation techniques can significantly improve the performance of Convolutional Neural Networks (CNNs) in image classification across varying domains and distributions. Our work applies Deep CORAL with EfficientNetV2 for domain adaptation on the Office-31 dataset. We compare its performance to a regular EfficientNetV2 model that doesn't use domain adaptation, measuring improvements with metrics as follows: accuracy, precision, recall, and F1 score. The CORAL-implemented model demonstrated a 4.15% average boost in average precision, recall, F1, and accuracy across all 3 trials.

Keywords

Artificial Intelligence, Computer Vision, Deep Learning, Domain Adaptation, Neural Networks, Convolutional Neural Networks, Transfer Learning

ACM Reference Format:
Aditya Chakraborty. 2025. Understanding Domain Adaptation Using CORAL in Computer Vision. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 2, ISSUE 2)*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.5281/zenodo.17195365

1 Introduction

The significance of Domain Adaptation remains of theoretical interest in the context of machine learning. It has real-world implications that are based on varying conditions and varying images based on light, sound, saturation, etc. Currently, Domain Adaptation is beneficial to systems that involve medical imaging, autonomous work, and remote sensing, where collecting new data for every possible domain is either expensive or impractical, or both. However, designing effective domain adaptation algorithms remains a challenge. Problems such as negative transfer, distribution mismatch, and lack of labeled target data require consideration before deployment and use.

Domain Adaptation has been considered as a promising technique to fix the domain shift problem by realigning the input data to the model training data while still preserving enough dimensions and key insights such that there will be no significant data loss in the input data.

As seen in Figure 1, reproduced from Zhao et al. [13], the circles represent different domains, and the shapes represent the features.

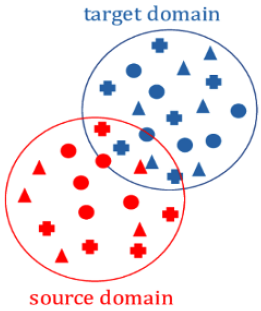


Figure 1: A diagram of domain shift

Domain-invariant features are the shapes that appear in both domains. Domains can differ in factors such as camera type, image saturation, blur, or time of day used to capture images. For example, in an object recognition task, domain-invariant features might focus on the object's shape and texture rather than lighting conditions, which may vary between the source and target domains.

As seen in Figure 2, this is a group of images extracted from the Office-31 dataset. As seen, these object images are captured from different lighting, backgrounds, and resolutions, which represent varying domains of feature representations.

Furthermore, Domain-invariant features are features that remain consistent and meaningful across different domains, allowing machine learning models to perform well even when the data distribution changes between training (source domain) and testing (target domain). Models trained on domain-invariant features can generalize better to different environments; therefore, these models can perform better on separate data distributions as long as the features are still recognizable in the datasets. In general, domain-invariant features are the key insights that are extracted from a dataset, which allows machine learning models to generalize and perform better in different domains.

Domain shift (or distributional shift) is a major problem that can negatively affect the performance of machine learning models when put in production [10]. Domain shift occurs when training, validation, and test data are drawn from a probability distribution that is different from the distribution of the data on which predictive models will be applied in [10]. Considerable costs of domain shift is the prediction of expected loss on the test data distribution. Although domain shift is challenging to completely reduce, we can gauge its adverse effects on out-of-sample predictions by taking

IJSCAR VOL. 2, ISSUE 2

Figure 2: A sample of images from each type of domain: Amazon, Webcam, DSLR.

special precautions when forming test samples. These are predictions from the model on input data that fall outside of the domain of the sample trained by the model.

Domain Adaptation offers a path forward in computer vision by enabling models to have high performance across varying domains and distributions without extensive training. By focusing on reducing domain shift, the arising bias that comes with introducing models in real-world settings, can be reduced, which ensures that models remain reliable and trustworthy on real-world datasets.

2 Related Work

2.1 Domain Adaptation

Traditional machine learning aims to learn a model over a set of training samples to find an objective function with minimum risk on unseen test data [2]. In traditional machine learning, it is assumed that training and test data are drawn from the same data distribution and share similar joint probability distributions. This constraint can be easily violated in real-world applications, since training, and test sets can originate from different feature spaces or distributions. The difficulty of collecting new instances with the same property, dimension, and distribution, etc, as we have observed in the training data, may occur due to a plethora of reasons, for example, the statistical properties of a domain can evolve in time, or new samples can be collected from different sources, causing domain shift.

Farahani et al., [2] refers to domain adaptation as consisting of three main parts: input or feature space X , output or label space Y, and an associated probability distribution $p(x, y)$, where overall, $D = X, Y, p(x, y)$. Feature space X is a subset of a d-dimensional space, $X \subset d, Y$ refers to either a space of binary $-1, +1$ or multi-class $1, ...K$, where K is the number of classes, and $p(x, y)$ is a joint probability distribution over the feature-label space pair XY. We can decompose the joint probability distribution as $p(x, y) = p(x)p(x|y)$ or $p(x, y) = p(y)p(y|x)$, where $p(.)$ is a marginal distribution and $p(.|.)$ is a conditional distribution.

Classification is a machine learning task that aims to learn a function from labeled training data to map input samples to real numbers.

$$h : \mathcal{X} \rightarrow \mathcal{Y},$$

where h is a function or an element of a hypothesis space $\mathcal{H}$, and $\mathcal{H}$ refers to a set of all possible functions.

Generally, to obtain the best predictive function, we learn a model on a given source dataset by minimizing the expected risk of the source-labeled data:

$$R_S(h) = \mathbb{E}_{(x,y)\sim P_S(x,y)}\left[\ell(h(x), y)\right] \tag{1}$$

$$= \sum_{y\in\mathcal{Y}} \int \ell(h(x), y)\, p_S(x, y)\, dx \tag{2}$$

where the expectation is taken with respect to the source distribution P_S, $\ell(h(x), y)$ is a loss function that denotes the error between the corresponding prediction by $h(x)$ and y.

However, in supervised learning, the goal is to learn a model with the most minimized loss, or the maximized likelihood given parameters, when applying it to the target domain. Thus, we can rewrite the above equation as follows:

$$R_T(h) = \mathbb{E}_{(x,y)\sim P_T}\left[\ell(h(x), y)\right] \tag{3}$$

$$= \sum_{y\in\mathcal{Y}} \int \ell(h(x), y)\, p_T(x, y)\, dx \tag{4}$$

$$= \sum_{y\in\mathcal{Y}} \int \ell(h(x), y)\, \frac{p_T(x, y)}{p_S(x, y)}\, p_S(x, y)\, dx \tag{5}$$

$$= \mathbb{E}_{(x,y)\sim P_S}\left[\frac{p_T(x, y)}{p_S(x, y)}\, \ell(h(x), y)\right], \tag{6}$$

where $P_S(x, y)$ and $P_T(x, y)$ are the joint probability distributions of the source and target domains, respectively.

In the context of machine learning, Domain adaptation with CORAL (CORrelation ALignment) involves aligning the statistical properties of source and target feature spaces to reduce domain shift. Specifically, CORAL minimizes the difference between the covariance matrices of the source and target feature representations, pushing the model to learn domain-invariant features. Unlike approaches that require labeled data in both domains, CORAL can operate in unsupervised settings, making it useful when the target domain lacks annotations that can cause traditional machine learning algorithms to fail.

Consider a scenario in which a model is being developed to classify road signs. The labeled training data (source domain) consists of high-resolution images of European road signs captured in clear weather using DSLR cameras. The model performs well on the source domain; however, the goal is to also make the model work effectively on images captured from dashcams in the United States under different conditions, such as poor resolution and varying

weather conditions. This data, one in the United States, has no labels available.

In this context, Domain Adaptation becomes crucial. Although tasks remain the same, the feature distributions between the source and target domains differ significantly due to external conditions, such as camera type, quality of resolution, or regional differences. Domain Adaptation helps mitigate falling in this gap by aligning feature representations between source and target domains regardless of labels in the source data. A method like CORAL can align the statistical properties (covariances) to better generalize to the unlabeled target domain by reshaping the training phase to allow the model to recognize domain-invariant features.

As shown in Figure 3, reproduced from [7], applying CORAL to CNNs involves a source dataset, which is labeled, and a target dataset, which is an unlabeled data set from a distribution that is different from the source data set. Both datasets are passed through the same layers, consisting of the convolutional layers, and are forwarded to the fully connected layers. Classification loss is computed only on the source data, and CORAL loss is computed using both datasets, with the purpose of minimizing the distance between the covariance matrices, effectively reducing domain shift by forcing the model to recognize objects that are seemingly different from underlying features, as the same object with the same domain-invariant features.

In contrast, generic transfer learning would involve using a pretrained model on a broad dataset, such as ImageNet, and then fine-tuning it on your labeled source domain (European road signs). Although this approach helps the model benefit from extracting visual features, it does not address the underlying domain shift between the source and target domains. Transfer learning also typically assumes some labeled data in the target domain for fine-tuning.

The key distinction is that Domain Adaptation is designed to handle domain shift between datasets sharing the same task, whereas transfer learning focuses on reusing knowledge from a related task or dataset, often requiring some level of supervision not necessarily required in Domain Adaptation.

In classification tasks, the objective is to learn a function that maps the input data to labels. In image classification, a classifier assigns each image to a specific category, such as a dog or a cat. To achieve the best predictive performance, a model is typically trained on the source data set by minimizing the expected error on the labeled source data. This is done by learning the model that minimizes the loss between the predicted and true labels in the source domain as per Farahani et al [2].

2.2 CORAL Architecture for Unsupervised Domain Adaptation

In domain adaptation, domains can be considered as an object consisting of three main parts: input or feature space represented as , output or label space , and, which is joined with the probability distribution of , creating a domain . Y refers to either the binary or multi-class spaces of {-1,1} or {1,...K} where K is the number of classes), [2].

Unsupervised Domain Adaptation (UDA) focuses on scenarios where labeled data are available only in the source domain, while

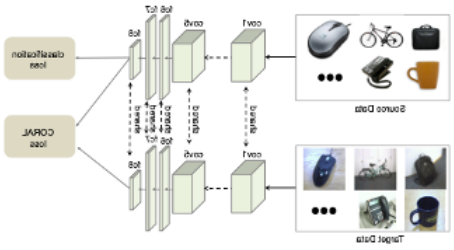


Figure 3: CORAL model architecture

the target domain lacks labels completely. The goal is to train a model using the labeled source data that can generalize well to the target domain. This is challenging because of the distribution shift between the source and target domains. Domain adaptation aims to build a classifier that can handle the shift in data distribution between the source and target domains [2].

To address this challenge, Domain adaptation (DA) techniques aim to bridge the gap between source and target domains. This is achieved by aligning feature distributions among different data domains to create effective and adaptive models. This technique is particularly useful when there are differences in data characteristics or when dealing with constraints on labels in the data. DA techniques, such as CORAL, help bridge domain shifts by minimizing domain discrepancies by aligning source and target features [2].

For an Unsupervised Domain Adaptation model, where the target data is unlabeled, the CORAL Loss model is designed to address the challenge of Unsupervised Domain Adaptation by aligning the covariance matrices of the source and target features [8]. In the architecture seen in Figure 3, source and target data are passed as input through the same convolutional neural network. Although the goal is to minimize the Cross-Entropy Loss and improve accuracy, implementing CORAL loss adds a secondary objective: training the feature extractors (Convolutional,Max Pooling, etc.) to align representations of the same object, or class across varying domains, in order to produce similar feature maps alongside minimized loss.

The classification loss, as well as CORAL loss, is extracted from the model and backpropagated as such. The input to the classification loss remains the same. However, CORAL Loss is extracted from the outputs passed after the final feature extraction layer, when both the source and target data are passed into the model, with the covariance matrices being calculated before being passed into the CORAL Loss function. The backpropagation algorithm with respect to CORAL Loss, determining the covariance matrix, and calculating the CORAL Loss are described below.

2.3 CORAL Loss

Suppose that we are given source training batches where $i \in \{1, \ldots, L\}$, and $D_S = \{x_i\}$ such that $x \in \mathbb{R}^d$ with labels $L_S = \{y_i\}$, and unlabeled target data $D_T = \{u_i\}$, where $u \in \mathbb{R}^d$. Assume the number of source and target data is n_s and n_t , respectively. In this case, both $\mathbf{x}$ and $\mathbf{u}$ are the specific d-dimensional deep layer activation function $\sigma(I)$ inputs labeled I that we are trying to tune and C_S and C_T are the respective second-order (covariance) matrices for the source and target data for the features, as per Sun and Saenko

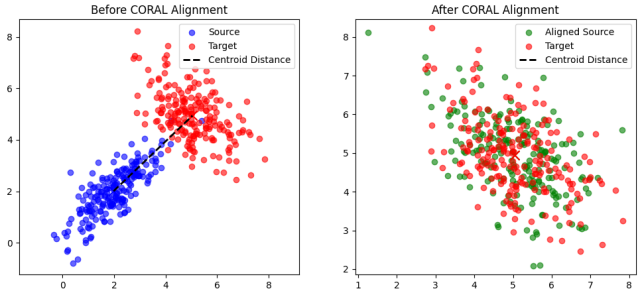


Figure 4: CORAL Alignment Demonstration

[8]. Covariance matrices represent how each feature varies with other features in a dataset, hence their correlation.

The covariance matrices are given by:

$$C_S = \frac{1}{n_S - 1}\left(D_S^\top D_S - \frac{1}{n_S}\left(\mathbf{1}^\top D_S\right)^\top \left(\mathbf{1}^\top D_S\right)\right) \tag{7}$$

$$C_T = \frac{1}{n_T - 1}\left(D_T^\top D_T - \frac{1}{n_T}\left(\mathbf{1}^\top D_T\right)^\top \left(\mathbf{1}^\top D_T\right)\right) \tag{8}$$

Simply, the CORAL loss is the distance between these two matrices, given by:

$$l_{\text{CORAL}} = \frac{1}{4d^2}\|C_S - C_T\|_F^2 \tag{9}$$

where $\|\cdot\|_F^2$ represents the Frobenius norm for square matrices. The loss is managed like other standard evaluation metrics such as MSE or cross-entropy; it is calculated per batch and averaged over training steps [7].

The Frobenius norm quantifies the distance between the source and target covariance matrices in a high-dimensional space. This distance represents the domain discrepancy, and the neural network is trained to minimize this discrepancy, thus reducing the domain shift and improving generalization across different domains [8]. The model is trained to recognize domain-invariant features by aligning the covariance matrices to the same domain.

The gradient with respect to the input features can be backpropagated as follows:

$$\frac{\partial l_{\text{CORAL}}}{\partial D_S^{ij}} = \frac{1}{d^2(n_S - 1)}\left[\left(D_S^\top - \frac{1}{n_S}\left(\mathbf{1}^\top D_S\right)^\top \mathbf{1}^\top\right)^\top (C_S - C_T)\right]^{ij} \tag{10}$$

Figure 4 demonstrates potential feature distributions of the source and target domains, represented as two clusters. The objective of CORAL alignment is to minimize the distance between these two clusters by reducing the disparity between their covariance structures. The target feature distribution remains at a fixed position, and the source feature distribution moves towards the target. After CORAL Alignment, caused by the generalization of neural network parameters with respect to the CORAL loss, the two clusters are represented very close to one another, effectively illustrating that the model has recognized the domain-invariant features that causes these distributions to be alike.

It is necessary to emphasize that minimizing this loss can potentially lead to overfitting to the source domain, resulting in reduced performance on the target domain. Having a simple reduction in CORAL loss alone may degenerate some features. The network may project the sources and targets to a single point, and the loss approaches 0 [8]. Secondly, the CORAL loss and classification loss are designed to be used simultaneously to address both limitations of the loss. The loss of classification does not account for the domain discrepancy that CORAL loss measures. This is where weighted CORAL losses arise.

$$l_{\text{TOTAL}} = l_{\text{CLASS}} + \lambda\, l_{\text{CORAL}} \tag{11}$$

Where t is the number of CORAL loss layers and λ is a weight factor that trades adaptation and precision to reach the lowest loss [7].

When integrating different Domain Adaptation techniques in models, it is essential to recognize the various types of domain shifts. This is required to understand what techniques are required to mitigate the effects of domain discrepancy. The following are common distribution shifts.

2.4 Alternate Domain Adaptation Technique: Subspace Distribution Alignment

Another method of domain distribution alignment is discussed in [9] paper that provides a solution to a common domain shift called Subspace Discrepancy.

Subspace discrepancy describes a scenario where observations are distributed as physical objects in the source and target domains but where the features used to describe them in one or the other are different and related by an unknown change of coordinates. For example, an object seen from different angles, as per Lemberger and Panico [5].

Assume that there is a source domain, named $D_S \in \{X_S, Y_S\}$ and a target domain $D_T \in \{X_T\}$ which is unlabeled. d represents the number of feature dimensions.

We compute the Principal Components from each Domain using Principal Component Analysis (PCA).

Let $P_S \in \mathbb{R}^{D\times d}$, representing the source subspace basis

Let $P_T \in \mathbb{R}^{D\times d}$, representing the target subspace basis

The objective is to calculate M such that the source subspace, is aligned with the target subspace: $P_S M \approx P_T$

Similar to Domain Adaptation using CORAL Loss function, the distance between these two subspaces in a multidimensional space must be minimized. This is formulated as the minimization Frobenius Norm between P_S and P_T, where

$$M_{t+1} = \arg\min_{M_t} \|P_S M_t - P_T\|_F^2 \tag{12}$$

Expanding the equation, there is a closed-form solution of

$$\|P_S M_t - P_T\|_F^2 = \text{Tr}\left[(P_S M_t - P_T)^\top (P_S M_t - P_T)\right]. \tag{13}$$

Taking the derivative with respect to M and setting it to zero results in:

$$M_{t+1} = P_S^\top P_T. \tag{14}$$

With the optimal mapping, the aligned source subspace is:

$$P_{S\to T} = P_S M_{t+1} = P_S(P_S^\top P_T). \tag{15}$$

Then, source data X_S is projected into the aligned representation:

$$Z_S = X_S P_{S \to T}, \quad Z_T = X_T P_T, \tag{16}$$

where Z_T is the target data representation in the P_T subspace.

Using the minimization of the Frobenius Norm, with respect to the computed linear mapping of M yields a map that accurately transforms the Source distribution D_S to D_T [9].

2.5 Categories of Domain Shift

2.5.1 Conditional Shift. Conditional Shift occurs when the marginal distribution of the input features changes between the source and target domains, but the conditional distribution of the labels given the inputs remains the same. In other words, while the underlying relationship between the features and labels remains stable, the input features themselves have shifted. For instance, in an object recognition task, if a model is trained on images of objects taken under one set of lighting conditions and then applied to images taken under different lighting conditions, a covariate shift occurs. The model may struggle because the features it learned during training are no longer sufficient to generalize to the new conditions.

Let $p(y_t, x_t)$ be a joint probability distribution such that the output y_t is the output and x_t is the input which will be extracted from our model. Domain shift occurs when training, validation, and / or testing is not drawn from the joint probability distribution but from a conditional probability: $p(z_t \in U)$

Where z_t is a random latent variable; note that z_t depends on y_t,x_t and U is a proper subset.

2.5.2 Covariate Shift. The covariate shift refers to a situation in machine learning where the distribution of the input data (features) changes between the training and testing phases, while the relationship between the input and output (the conditional distribution of the output given the input) remains the same. This shift can lead to degraded performance because the model was trained on data that do not fully represent the distribution it encounters during testing or real-world deployment [8].

Let $p(X_{train})$ represent the data distribution of the input data applied during the training phase. Let $p(X_{test})$ represent the data distribution during the testing phase:

In the current problem, we assume that $p(X_{train}) \neq (X_{test})$, however, $p(Y) = p(X)$ where X is all input data and Y represents output labels [7]. Hence, $p(Y|X)$ represents the relationship between the input features and the output labels remain constant across the source and target domains.

If the model is trained on a specific data distribution, there may be bias to a certain distribution and/or certain patterns that are not domain-invariant features. To represent this distribution shift, metrics such as the Kolmogorov-Smirnov test and the Jensen-Shannon divergence test are used [1].

3 Methods

One of the most widely used datasets for studying domain adaptation is the Office-31 dataset [3]. This data set includes 4,110 images across 31 category of objects captured in three distinct domains:DSLR, Webcam, and Amazon. These domains represent different imaging conditions: DSLR images are of high quality and resolution, whereas Webcam images are grainy and noisy. The Amazon domain consists of product images downloaded from the e-Commerce platform, often featuring different backgrounds and lighting conditions. As mentioned earlier, the domain shift represented in this dataset is a covariate shift as the source input distribution does not equal the target distribution; however, the relationship between the input and labels stays constant. The overall distribution of this data set makes Office-31 a powerful dataset to test model robustness.

3.1 Domain Shift

In a Domain Adaptation, it is crucial to measure how different the source and target data distributions are, since the primary challenge lies in training a model on one domain and ensuring it performs well on another with a potentially different distribution. This is where Kolmogorov-Smirnov (KS) and Jensen-Shannon (JS) Divergence test become valuable tools [1]. The KS test is useful for statistically testing whether the source and target distributions (e.g., pixel intensities or individual feature values) come from the same population, offering a formal hypothesis test with p-values [1].

JS Divergence provides a smooth, symmetric measure to quantify the distance between two probability distributions, making it particularly effective for comparing SoftMax outputs or high-dimensional learned feature embeddings.

KS Test answers the question of whether two dataset distributions (in our case, the webcam and DSLR) are drawn from the same data distribution by measuring the maximum vertical distance. If the distance is small, the distributions are similar; if it is large, the distributions are different. The KS statistic measures the maximum distance between two cumulative distribution functions (CDFs). KS plots the CDF of each dataset and finds the biggest gap between them.

JS Divergence, however, focuses on probability distributions, specifically the average of the two distributions, and measures how each differs. The JSD measures the similarity between two probability distributions (P) and (Q)

Metric	Value
KS Statistic	0.0131
KS p-value	8.02×10^{-25}
JS Divergence	0.00959

Table 1: Pre-training feature discrepancy between source (DSLR) and target (Webcam) domains. KS: Kolmogorov-Smirnov test, JS: Jensen-Shannon divergence.

The results indicate a clear distribution shift between the DSLR and Webcam datasets. A Jensen-Shannon Divergence of 0.1669 suggests a moderate difference in the average pixel distributions, reflecting changes in lighting, contrast, or color profiles. The Kolmogorov-Smirnov test further supports this, with a high statistic of 0.6670 and a p-value of 0.0000, confirming a statistically significant difference in image-level intensity distributions. Together, these metrics highlight the need for domain adaptation techniques to address the shift when training models across these datasets.

3.2 Objectives of the Experiment

This experiment investigates the value of Unsupervised Domain Adaptation to improve the generalization of image recognition models to capture domain-invariant features and to recognize and classify images accurately in different environments outside of the training dataset. We will apply the supervised Deep CORAL domain adaptation technique using the CORAL loss function.. Unlike B. Sun's, J. Feng's, and K. Saenko's CORAL implementation with AlexNet [7], this experiment will use EfficientNet version 2 (V2). EfficientNetV2, a deep convolutional neural network, is widely known for its robust SOTA performance, known for its balance between efficiency and accuracy, resulting in models that train faster and are significantly smaller than other models, hence, it was picked to reduce computational workload and time required between experiments. This experiment is designed to test two phases:

Control Experiment: EfficientNetV2 without Domain Adaptation

In the control experiment, a standard EfficientNetV2 model was trained without domain adaptation. The model was trained on the DSLR domain, which contains high-quality images, for 20 epochs using a batch size of 64 and the Adam optimizer. The loss function used was cross-entropy loss for classification. A validation split of 20% of the training data was used to mitigate bias. The model was evaluated on the target domain, Webcam, which contains lower-quality, noisier images representing a distribution shift. Model performance was assessed using accuracy, precision, recall, F1 score, and paired t-tests across trials.

Domain Adaptation Experiment: EfficientNetV2 with Deep CORAL

In the domain adaptation experiment, EfficientNetV2 was trained using Deep CORAL to align feature distributions between the source (DSLR) and target (Webcam) domains. Training was conducted for 20 epochs with a batch size of 64 and the Adam optimizer. The total loss combined cross-entropy classification loss and CORAL loss as follows:

$$l_{\text{TOTAL}} = l_{\text{CLASS}} + \lambda\, l_{\text{CORAL}} \tag{17}$$

where $\lambda = 10$ to balance the contribution of classification and CORAL losses. A 20% validation split was applied as in the control experiment. Performance metrics included accuracy, precision, recall, and F1 score, averaged across all trials.

4 Results

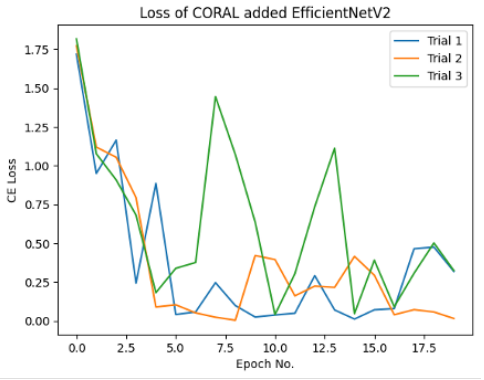


Figure 5: Loss graph of CORAL-added EfficientNet V2

Seen in Figure 5, the lowest loss converges to near zero. This is much more stable than without CORAL due to the property of the CORAL loss to align domain-invariant features. Uniquely, the CORAL loss becomes more stable at the very end, in comparison to the control EfficientNetV2.

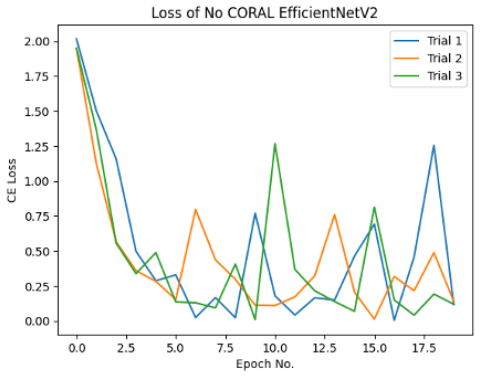


Figure 6: Loss graph of no CORAL-added EfficientNet V2

Figure 6 shows much more unstable fluctuations in training losses over the epochs. Unlike, CORAL-implemented EfficientNetV2, there are significant fluctuations at the end at approximately epoch no. 18, where, for example, trial 1 abruptly increases the loss by 1036%. Although there are fluctuations on the CORAL-implemented EfficientNet that must be recognized, these abrupt fluctuations are much more severe.

Model	Calc.	Desk Chair	Desk Lamp	Desktop PC	File Cab.
CORAL EfficientNetV2					
Precision	1.0	1.0	1.0	0.8898	1.0
Recall	0.9785	0.9917	1.0	1.0	0.9123
F1	0.9889	0.9958	1.0	0.9410	0.9518
Control EfficientNetV2					
Precision	1.0	1.0	1.0	0.9565	0.7555
Recall	1.0	0.9833	0.9630	0.7302	0.9825
F1	1.0	0.9915	0.9804	0.8121	0.8510

Table 2: A sample of performance comparison representing all classes

As seen in Table 2, which contains a random sample of 5 classes over 31 trained classes, CORAL considerably increased the precision, and recall of classes such as Desktop PC.

Metric	t-statistic	p-value
Accuracy	-5.0000	3.775e-02
F1	-4.0332	5.633e-02

Table 3: Paired t-test results comparing Baseline and CORAL models.

For accuracy, seen in Table 3, the t-statistic of -5.0 indicates that the mean CORAL accuracy is significantly higher than the baseline model, which cannot be deduced by chance but instead refers to the property of CORAL loss that recognizes the domain-invariant features, allowing the model to recognize the features that exist across all distributions. The p-value is 0.03775, which indicates that the CORAL loss is statistically significant. This aligns towards supporting the hypothesis that integrating CORAL loss directly improved EfficientNet CNN performance.

As seen in Table 3, which also represent the F1 score, the t-statistic of -4.0332 indicates a considerable trend between CORAL implementation and higher F1 scores. However, the p-value of 0.05633 slightly exceeds the conventional margin of 0.05. Despite the p-value, the negative t-statistic suggests that CORAL improves model performance.

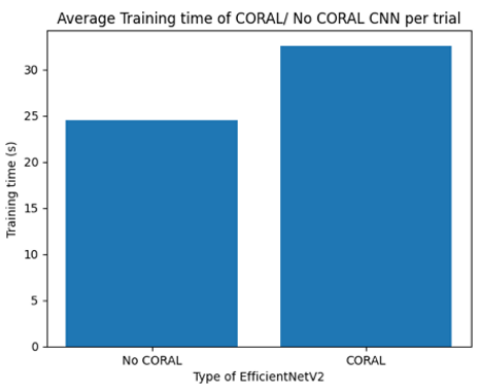


Figure 7: The average training time per trial of CORAL and No CORAL EfficientNetV2

Figure 7 shows a significant time discrepancy between using No CORAL and using CORAL. There is 29.55 % discrepancy, indicating an important limitation to training with CORAL domain adaptation

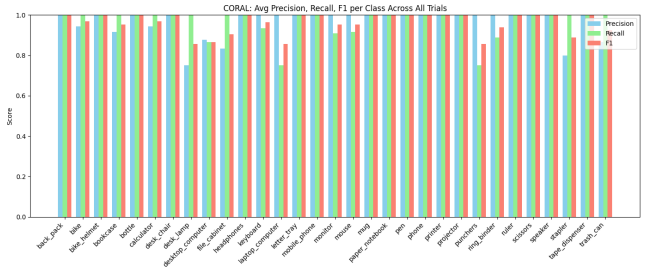


Figure 8: Bar Graph of Precision, Recall, and F1 per class for CORAL EfficientNetV2

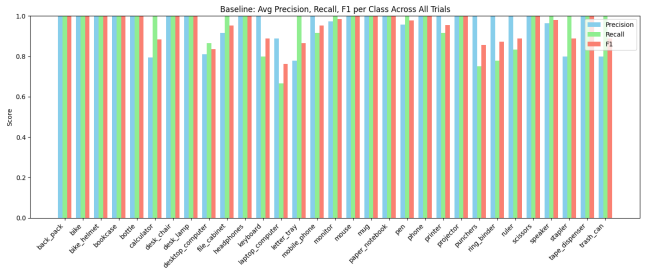


Figure 9: Bar Graph of Precision, Recall, and F1 per class for Control EfficientNetV2

Figure 8 and 9 show significant changes in the metric evaluation between CORAL implementation and regular EfficientNet. Although, CORAL fails to increase the precision, recall or F1 on multiple classes, it, however, increased the precision and recall on classes which the control EfficientNet evaluated the poorest on, such as the `desktop_computer` class by 16.67%, `letter_tray` by 25%, `mobile_phone` by 7%, etc, on average.

Model	Accuracy	Precision	Recall	F1
Baseline (No CORAL)	0.9586	0.9575	0.9525	0.9579
CORAL	0.9689	0.9656	0.9682	0.9691

Table 4: Comparison of Baseline and CORAL-enhanced EfficientNet Accuracy, and average Precision, Recall, F1 across all classes and trials.

Metric	Baseline EfficientNetB0	CORAL EfficientNetB0
Overall Performance		
Accuracy	0.9586 ± 0.0029	0.9689 ± 0.0000
Weighted F1	0.9579 ± 0.0039	0.9691 ± 0.0001
Paired t-test (Baseline vs CORAL)		
Accuracy t / p	$-5.0000 / 3.78 \times 10^{-2}$	
Weighted F1 t / p	$-4.0332 / 5.63 \times 10^{-2}$	

Table 5: Comparison of Baseline and CORAL EfficientNetV2

5 Discussion

The use of CORAL in EfficientNetV2 has resulted in significant performance improvements in terms of a variety of classification metrics when using Domain Adaptation. As seen from Figure 5, 8, and Table 4 the CORAL-implemented EfficientNetV2 generally performed better than the baseline model (non-CORAL) on the target sample. This verifies the hypothesis that aligning the source and target features using CORAL makes the model more generalizable in domain discrepancy scenarios.

As seen in Table 4, the CORAL-implemented model demonstrated a 4.15% average boost in precision, recall, F1, and accuracy against the metrics. While this margin of improvement may seem insignificant, relative to domain adaptation work, marginal values are likely to approach better real-world standards in deployment. Additionally, this gain held across most categories and improved the F1 values of the `desktop_computer and file_cabinet` classes, suggesting broad advantages.

One of the most significant results is obtained by evaluating the model on poorly performed classes, which, in this case, are `desktop_computer and file_cabinet`. These classes had previously given abnormally low precision, recall, and F1 scores, especially compared to the rest of the metrics across all the classes. With CORAL implementation in the training procedure, both classes observed a significant increase their F1 scores, which effectively shows that CORAL enhances the model's capability for learning generalized features that are consistent across multiple domains, even for classes where capturing these features might be difficult, and may become overcomplicated by CORAL-implementation. This acts to further strengthen CORAL as an effective tool in the case where specific classes are adversely affected by domain shift.

Training dynamics further suggests the effectiveness of CORAL. Figure 5 shows that the categorical cross-entropy loss after training was 14% smaller in the case of CORAL implementation in comparison to the baseline, seen in Figure 6. This suggests better-calibrated and more confident predictions, which is a quality required in models that are being deployed and exposed to real-world contexts. There are also less significant fluctuations in loss for the CORAL-added model. Moreover, a lower loss indicates improvements in convergence, reinforcing the concept that CORAL provides a smooth optimization path by minimizing the discrepancy between source and target features.

Training dynamics were similar to those of B. Sun and K. Saenko [7], shown in figure 10, which was trained on a pretrained AlexNet model. The CORAL may not have resulted in a lower loss between the source and target samples in comparison to the baseline model, however the CORAL Loss attained a more stable loss progression than the classification loss.

This improved result trades off with computing time. Incorporating CORAL resulted in additional computational cost and increased training time by approximately 29.55% across all 3 trials. This is because of additional matrix operations to calculate second-order statistics alignment across domains. Depending on the context, or the problem certain machine learning models are required to solve, trade-offs may potentially be required between increases in model complexity, and the time required to complete the training phase.

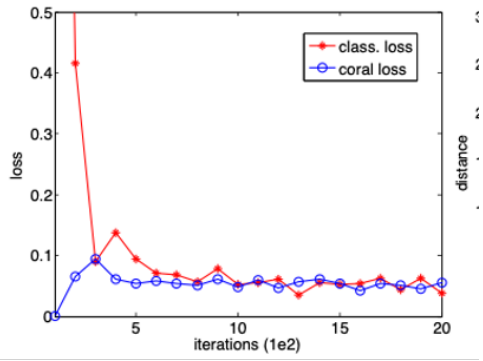


Figure 10: Classification loss vs CORAL Loss in minimizing domain discrepancy

6 Limitations

Integration of CORAL into EfficientNetV2 has resulted in notable improvements in domain adaptation performance, however, several limitations exist that challenge consideration of deployment into real-world contexts.

The incorporation of CORAL introduces an additional computational workload. The alignment of covariance, second-order statistics between the source and target samples requires additional matrix operations, especially on high-dimensional features, leading to a 29.55% increase in training time. This computational demand may uncover challenges for deployment while still factoring in resource usage and management, especially in resource-constrained areas or applications that require real-time processing.

If the CORAL loss is emphasized during training, there is a potential risk of the network learning degenerate features. For example, minimizing CORAL loss alone might lead the model to project both source and target to a point in a multidimensional space that results in an alignment that yields a lack of discriminative power [7].

The effectiveness of CORAL depends upon the quality of the features extracted by the CNN. Using pretrained architectures that may not capture domain-invariant features, such as AlexNet, which can limit the alignment [12] of second-order statistics and representable features. Furthermore, concurrent training of CORAL and a classifier may not efficiently align features if the domain-invariant features that models aim to recognize are not satisfactory.

CORAL operates under the assumption that the solution to domain shift is to align only second-order statistics between domains. This may not hold in scenarios where greater-order statistical differences exist that limit CORAL's ability to adapt to complex, real-world scenarios [12].

While CORAL enhances domain adaptation capabilities, its limitations, ranging from computational demands to statistical misrepresentation and non-feasibility, should be carefully considered when deploying in real-world settings.

7 Future Work

Expanding on CORAL implementation, several open challenges and promising paths remain. Future research needs to be directed

at constructing unified frameworks that can address multi-source and multi-target paradigms. Currently, the majority of methods are only for single-source and single-target cases, whereas real-world tasks often involve multiple sets of uniformly distributed domains. Designing algorithms that can effectively leverage multiple sources of information and adapt to a variety of target conditions is not trivial but a significant step towards real-world deployment.

One direction is to explore computationally light variations or hybrid methods that maintain CORAL's alignment capability without significantly increasing training time. Because our CORAL-trained EfficientNet model witnessed a 29.55% increase in training time due to second-order matrix computation, optimizing CORAL's implementation or integrating light-weight domain alignment techniques can potentially allow it to be more applicable in real-time or resource-constrained situations.

In addition, domain robustness and fairness are key areas of future research for model robustness. Domain shifts are likely to exaggerate performance fluctuations, especially when models operate in performance-critical and socially sensitive domains. Future work needs to focus on the development of domain adaptation techniques that are not only accurate but also invariant to adversarial forces and are robust against dataset bias. Recent studies have explored adversarial approaches to mitigate such biases and enhance model resilience, seen in the works of Tzeng et al. [11], and Huang et al. [4]. Moreover, transparency and fairness in adapted models are crucial, particularly in applications that range from healthcare to surveillance and autonomous systems (self-driving cars).

Finally, understanding the decision-making process of domain adaptation models is essential for confirming accountability and interpretability. Explainable AI (XAI) algorithms, such as LIME and SHAP, are commonly used to uncover reasoning behind model predictions. Integrating XAI into domain adaptation systems can help recognize model behavior and ensure transparency, seen in the work of Ribeiro et al. in applying LIME for model transparency [6].

Addressing these challenges, such as improving robustness and fairness, improving simulation-to-real transfer, and hosting explainability in CORAL-assisted machine learning models, will become critical for the advancement and deployment of domain adaptation in computer vision.

8 Conclusion

This study highlights the importance of Domain Adaptation in mitigating the effects of domain shift in computer vision. By using CORAL with EfficientNetV2, we were able to align the feature distributions of the source (Webcam) and target (DSLR) domains, resulting in significantly improved performance on target domain classification tasks. Compared to the baseline model of EfficientNetV2, the CORAL-enhanced model consistently improved the performance over varying domains across all evaluated classes.

The Jensen-Shannon Divergence (JSD) and the Kolmogorov-Smirnov Test (KS Test) confirmed a detectable distributional shift between the DSLR and Webcam domains, restating the need for an adaptation technique that centralizes all training data to the model. Deep CORAL effectively minimized the discrepancy by aligning the covariance matrices, allowing the model to learn valuable domain-invariant features that a non-CORAL-enhanced model may not recognize.

Overall, these results effectively demonstrated that using domain adaptation techniques such as the investigated CORAL are not only theoretically significant but also practically effective. In real-world scenarios, domain adaptation enables models to generalize better across varying conditions, recognizing domain-invariant features regardless of external factors and stimuli.

9 Acknowledgement

I would like to thank my mentor, Dr. Nirmala Ramakrishnan, for her valuable guidance and feedback on my drafts and roadmap to writing this paper. Dr. Ramakrishnan's support and mentorship were instrumental in helping me complete this work.

References

[1] Astrostatistics and Astroinformatics Portal. Beware the kolmogorov-smirnov test! https://asaip.psu.edu/articles/beware-the-kolmogorov-smirnov-test/, 2019. Accessed: 2025-05-08.

[2] Farahani, A., Voghoei, S., Rasheed, K., and Arabnia, H. R. A brief review of domain adaptation. *arXiv preprint arXiv:2010.03978* (Oct 2020).

[3] Hu, X. Office31. https://www.kaggle.com/datasets/xixuhu/office31, 2022. Kaggle Dataset.

[4] Huang, J., Guan, D., Xiao, A., and Lu, S. Rda: Robust domain adaptation via fourier adversarial attacking. *arXiv preprint arXiv:2106.02874* (Jun 2021).

[5] Lemberger, P., and Panico, I. A primer on domain adaptation. *arXiv preprint arXiv:2001.09994* (Jan 2020).

[6] Ribeiro, M. T., Singh, S., and Guestrin, C. "Why Should I Trust You?": Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations* (2016), pp. 97–101.

[7] Sun, B., Feng, J., and Saenko, K. Deep coral: Correlation alignment for deep domain adaptation. *arXiv preprint arXiv:1612.01939* (Dec 2016).

[8] Sun, B., and Saenko, K. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision (ECCV)* (2016), pp. 443–450.

[9] Sun, S. Subspace distribution alignment for unsupervised domain adaptation. *BMVA preprint bmva:https://dx.doi.org/10.5244/C.29.24* (Oct 2013).

[10] Taboga, M. Domain shift. https://www.statlect.com/machine-learning/domain-shift, 2021. Lectures on machine learning.

[11] Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464* (Feb 2017).

[12] Wang, Z.-Y., and Kang, D.-K. P-norm attention deep coral: Extending correlation alignment using attention and the p-norm loss function. *Applied Sciences 11*, 11 (2021), 5267.

[13] Zhao, S., and Lang, H. Improving deep subdomain adaptation by dual-branch network embedding attention module for sar ship classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 15* (2022), 8038–8048.

Received 28 August 2025; Accepted 18 September 2025