2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

Transformer-Based Robust Tactile Object Recognition under Sensor Faults through Training, Adaptation, and Correction

Masanori Muroyama, *Member, IEEE*

Abstract— **Tactile sensors embedded in robotic systems are vulnerable to hardware-level faults such as short-circuits (saturated high-value output, e.g., 5 N) and open-circuits (zero-value output), which may arise from realistic physical failures such as broken signal lines or internal disconnections. In this work, I explicitly model such hardware-induced faults based on the physical wiring structure of tactile sensor arrays and investigate their impact on object recognition accuracy. I first perform a sensitivity analysis to identify sensor regions where abnormal values (e.g., 0 N or 5 N) have particularly strong influence on recognition performance. Building on these insights, I design a robust recognition framework using a Transformer-based recognition model, and evaluate three strategies: (1) fault-aware training with simulated failure patterns, (2) selective removal of known faulty sensor data, and (3) online correction of abnormal values by converting them into statistically neutral values with minimal impact on the learned force distribution. The experiments show that while fault injection during training provides some robustness, it is difficult to generalize across all possible failure patterns. In contrast, fault detection and correction at inference time significantly restore recognition accuracy under severe fault conditions. These results highlight the necessity of integrating both hardware fault modeling and adaptive inference mechanisms for reliable tactile perception in real-world robotic systems.**

I. Introduction

Tactile sensing has a long and evolving history rooted in the desire to endow robotic systems with the sense of touch, enabling interaction with unstructured environments in a manner similar to humans. Early efforts in tactile technology began with discrete force and pressure sensors, eventually leading to the development of tactile arrays or "electronic skin" capable of detecting contact, pressure distribution, and texture. Over time, research has progressed from simple resistive and capacitive sensors to complex multimodal tactile skins capable of dynamic interaction and active exploration [1,2].

Dahiya et al. provided a comprehensive overview of tactile sensing from a biological inspiration standpoint to its application in humanoid robotics, highlighting challenges in resolution, compliance, and integration into artificial skins [2]. Ji et al. further emphasized the need for intelligent processing, distributed architectures, and energy efficiency for effective deployment of large-area tactile skins [3]. More recently, flexible tactile sensors utilizing piezoelectric, triboelectric, and hybrid mechanisms have emerged, offering high spatial resolution, fast response, and self-powered capabilities [4]. These advances reflect a continuous push toward biologically inspired, robust, and scalable tactile sensing solutions.

In recent years, tactile object recognition has benefited significantly from advances in deep learning, particularly through the application of Convolutional Neural Networks (CNNs), attention-based models, and multimodal fusion strategies. Gandarias et al. demonstrated that tactile pressure data collected from high-resolution tactile sensors can be effectively treated as image-like input and processed via CNN-based architectures for object classification tasks, achieving notable performance improvements over traditional feature-based methods [5]. Building upon this idea, Chen et al. proposed Tactile-GAT, a graph attention network that captures the spatial and relational topology of tactile sensor arrays by modeling them as graphs. This approach outperformed CNNs by leveraging inter-sensor dependencies, achieving higher accuracy in tactile perception classification [6].

In addition to unimodal tactile processing, multimodal fusion techniques that integrate visual and tactile information have shown strong potential in enhancing recognition robustness. For example, Ding et al. introduced an adaptive fusion method that dynamically balances visual and tactile cues for object classification, yielding accuracy exceeding 99% in several benchmark tasks [7]. Furthermore, Zheng et al. tackled the open-set recognition problem by proposing a Gaussian Prototype Learning framework that models uncertainty in tactile feature distributions, enabling generalization to novel objects not seen during training [8].

These approaches collectively illustrate the transition of tactile recognition research from conventional sensing and signal processing to machine learning-driven perception, with a growing emphasis on robustness, generalization, and multi-sensory integration.

In addition to robust training strategies, recent studies have explored fault-tolerant control schemes for robotic manipulators that rely on probabilistic inference models to detect and compensate for faulty sensory inputs [9]. While these methods focus on continuous control under fault conditions, my approach targets high-dimensional tactile data used in classification tasks, which requires different forms of preprocessing and robustness enhancement.

This study focuses on improving the robustness of tactile object recognition systems under realistic sensor fault conditions. Instead of relying solely on ideal training environments, I evaluate three practical fault-handling strategies: (1) training with simulated fault patterns, (2) removing known faulty sensor data, and (3) replacing abnormal values with low-impact surrogates during inference. These approaches are tested using real hardware fault scenarios to assess their individual and combined effectiveness, thereby

This research was supported by Adaptable and Seamless Technology transfer Program through Target-driven R&D (A-STEP) from Japan Science and Technology Agency (JST), Grant Number JPMJTR23R7. The author also appreciates the support from Tohoku University, Shuji Tanaka Laboratory.

The author is with the Department of Electrical and Electronic Engineering, Tohoku Institute of Technology, Sendai 982-8577, Japan (e-mail: m-muroyama@tohtech.ac.jp).

providing foundational insights for fault-tolerant tactile perception.

To support these strategies, I employ a Transformer-based architecture capable of capturing complex spatiotemporal relationships in high-dimensional tactile sensor data. Unlike CNNs or RNNs, the self-attention mechanism in Transformers enables the model to selectively focus on relevant sensor inputs across time steps. This property is particularly advantageous under conditions of partial sensor faults or data corruption. Such flexibility enhances robustness and improves recognition accuracy even in degraded conditions.

Unlike prior generic partial-observability approaches, this work explicitly integrates hardware-specific fault behaviors observed in resistive tactile arrays with a Transformer-based recognition pipeline. This device-driven modeling differentiates the present study from conventional robustness strategies.

This paper is organized as follows. Section II introduces my tactile sensor system that forms the foundation of my study, as well as its integration with the robot and the ROS2 framework. In Section III, I describe the recognition system under normal (fault-free) conditions. Section III also presents a fault-aware training strategy that anticipates potential sensor faults during the learning phase. Section IV explores an approach for handling actual sensor faults by identifying faulty sensors and adapting the model accordingly. Section V reports a method that mitigates the impact of abnormal sensor values by converting them into low-impact values. Section VI details the experimental results, and finally, Section VII summarizes the findings and concludes the paper.

II. Tactile Sensor Data Acquisition System with ROS2 based Robot Control

To collect tactile data for this study, I constructed a sensing system based on two commercially available resistive tactile sensor arrays (MMS1011 by Alpha (Taiwan)) connected to an ESP32 DevKit microcontroller (Figure 1). Each array consists of 11 columns and 10 rows, yielding 110 sensing elements per array and a total of 220 tactile sensor elements across both arrays. The sensor arrays are scanned sequentially by the ESP32 using GPIO-controlled multiplexing logic. For each array, dedicated 4-bit address lines are used to select specific columns and rows. Analog voltages representing the resistive responses of the tactile elements are read through two analog input channels, one for each array. The scanning procedure is optimized for low-latency operation using microsecond-level timing control. One complete scan cycle of all 220 sensors is completed approximately every 100 milliseconds, supporting continuous tactile sensing with relatively high temporal resolution. To facilitate real-time remote monitoring and processing, the ESP32 connects to a local Wi-Fi network and transmits the scanned tactile data over UDP (User Datagram Protocol). Each packet includes a unique device ID, a timestamp, and the sensor readings formatted as a CSV-formatted string. The data are sent to a host PC configured to listen on a specific IP address and port.

On the PC side, a ROS2 (Galactic) node is responsible for receiving the UDP packets and decoding the tactile data. This ROS2-based system provides modular integration for further data processing, classification, and storage. Whether the received tactile data should be saved to file is determined in coordination with the robot's manipulation behavior, particularly during grasping.

The robot platform used in this study is the Mercury B1 semi-humanoid robot developed by Elephant Robotics. It is also controlled via ROS2, with a dedicated node that listens for control commands and executes pre-defined grasp motions. The timing of data saving is synchronized with the robot's grasp execution, allowing for selective logging of tactile data during contact-rich interactions. This integration between the tactile sensing system and the robot controller ensures precise data labeling and enhances the quality of supervised learning for downstream classification tasks.

III. Preliminary Study of Fault-Robust Classification by Training with Simulated Sensor Faults

This section presents a foundational investigation into building classification models that remain effective under sensor failure conditions. I design a Transformer-based deep learning architecture and introduce a training methodology that simulates hardware-level sensor faults. The objective is to evaluate how the model generalizes under realistic failure

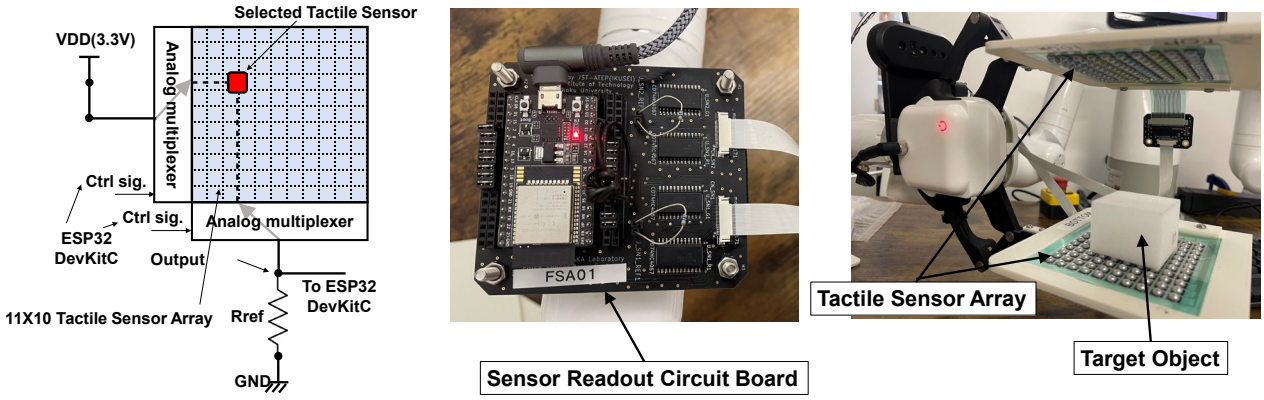

Figure 1. (left)Sensing system block diagragm, (center)sensor readout circuit board, (right)robot finger tips w/ two tactile sensor arrays.

patterns and to assess the feasibility of fault-aware training with a limited number of synthetic combinations.

A. Model Architecture

Let the input to the model be a 3D tensor $\mathbf{X} \in \mathbb{R}^{T\times C}$, where $T = 10$ is the number of time steps and $C = 220$ is the number of tactile sensor elements per time step. The input sequence is reshaped for spatial convolution as:

$\mathbf{X}_{\text{reshaped}} \in \mathbb{R}^{T\times H\times W\times D}$ where $H = 11, W = 10, D = 2,$ and $H \cdot W \cdot D = C$.

I apply two stacked **TimeDistributed** 2D convolution layers:

$$\mathbf{Z_1} = \text{Conv2D}_{(3\times3,16)}(\mathbf{X}_{\text{reshaped}}),$$

$$\mathbf{Z_2} = \text{Conv2D}_{(3\times3,32)}(\mathbf{Z_1}).$$

After flattening the spatial dimensions, I obtain:

$$\mathbf{F} = \text{TimeDistributed}(\text{Flatten})(\mathbf{Z_2}) \in \mathbb{R}^{T\times d},$$

e.g., d=number of output features.

The two Conv2D layers are applied independently to each of the 10 frames using TimeDistributed wrappers. Each frame is first processed by Conv2D layers, and the resulting feature map is flattened into a fixed-length feature vector. These 10 frame-wise feature vectors are then stacked in temporal order to form a sequence input to the Transformer encoder. In this structure, the Transformer operates across time steps rather than across spatial dimensions, enabling temporal integration of spatial features extracted by the Conv2D layers.

This is followed by two Transformer Encoder Blocks. Each block includes:

1. Multi-Head Self-Attention (MHSA):

$$\text{MHSA}(\mathbf{F}) = \text{Concat}(\text{head}_1, \text{head}_2, \text{head}_3, \text{head}_4)\mathbf{W}^O.$$

each head uses dimension $d_k = 64$, and queries / keys / values are projected as:

$$\text{head}_i = \text{Attention}(\mathbf{F}\mathbf{W}_i^Q, \mathbf{F}\mathbf{W}_i^K, \mathbf{F}\mathbf{W}_i^V).$$

Here, $\mathbf{F}\mathbf{W}_i^Q$, $\mathbf{F}\mathbf{W}_i^K$, and $\mathbf{F}\mathbf{W}_i^V$ denote the linear projections of the input sequence F into query, key, and value spaces for the *i*-th head, using learned projection matrices. The output projection matrix $\mathbf{W}^O$ is also learned and used to project the concatenated multi-head output back into the model's hidden space.

2. Feedforward Network (FFN):

$$\text{FFN}(\mathbf{x}) = \text{Dense}_{128}\left(\text{ReLU}(\text{Dense}_d(\text{x}))\right).$$

3. Residual and LayerNorm:

$$\mathbf{F} \leftarrow \text{LayerNorm}\left(\mathbf{F} + \text{Dropout}(\text{MHSA}(\mathbf{F}))\right),$$

$$\mathbf{F} \leftarrow \text{LayerNorm}\left(\mathbf{F} + \text{Dropout}(\text{FFN}(\mathbf{F}))\right).$$

After Transformer encoding, I apply temporal pooling:

$$\mathbf{g} = \text{GlobalAveragePooling1D}(\mathbf{F}) \in \mathbb{R}^d,$$

Then a classification head follows:

$$\mathbf{z} = \text{Dropout}(0.1)\left(\text{ReLU}(\text{Dense}_{128}(\mathbf{g}))\right),$$

$$\hat{\boldsymbol{y}} = \text{Softmax}(\text{Dense}_7(\mathbf{z})).$$

B. Training Procedure and Evaluation Metrics

- Loss Function: Categorical Cross Entropy

$$\mathcal{L} = -\sum_{i=1}^{K} y_i \log \hat{y}_i.$$

- Optimization: Adam with learning rate $\eta = 10^{-4}$, batch size = 32, max epochs = 50
- Metrics:
  - Accuracy: $\frac{1}{N}\sum_{i=1}^{N} 1\{\hat{y}_i = y_i\}$
  - Macro Precision / Recall / F1-score: averaged across all classes

C. Fault-Robust Training Method

To simulate sensor failures, for each input sample during training:

- Randomly select k sensor elements from 220 (e.g., $k =$ 5 or 10)
- Replace each selected sensor element with either:

  0 → open-circuit simulation

  5 → short-circuit simulation
- Assignment is random with equal probability

This produces a dataset $D_{\text{faulty}} = \{(\tilde{\mathbf{X}}, \boldsymbol{y})\}, where\ \tilde{\mathbf{X}}$ contains simulated sensor faults.

The model learns to generalize to such corrupted inputs without explicit labels for which sensors are faulty. The objective becomes:

$$\min_{\theta} \mathbb{E}_{(\tilde{\mathbf{x}},y)\sim D_{\text{faulty}}}\left[\mathcal{L}(f_\theta(\tilde{\mathbf{X}}), \boldsymbol{y})\right].$$

The number of possible fault combinations is exponential:

$$\sum_{k=1}^{n}\binom{n}{k}\cdot 2^k = 3^n - 1.$$

For n=220, this results in approximately 10^{105} possible fault patterns. Therefore, exhaustive training is infeasible, and this study sample fault cases randomly to demonstrate the feasibility of robust learning.

IV. Fault-Robust Classification via Sensor-Fault-Aware Model Adaptation

In sensor-based classification systems, hardware-level failures such as open-circuit disconnections, saturated outputs, or fixed-value faults can significantly degrade recognition accuracy. These types of faults distort the input features in a nontrivial way, reducing the model's ability to reliably distinguish between classes. To address this, I propose a model

adaptation strategy that incorporates knowledge of faulty sensor positions to modify the input space prior to training or inference.

Let $\mathbf{x}_i \in \mathbb{R}^{T\times D}$ be the time-series input sample, where $T = 10$ is the time length and $D = 220$ is the number of tactile sensor elements. Suppose I have a known set of faulty sensor indices:

$$\mathcal{F} = \{d_1, d_2, \ldots, d_k\} \subset \{1,2,\ldots,D\}.$$

I define a feature selector function $\phi: \mathbb{R}^{T\times D} \to \mathbb{R}^{T\times(D-k)}$ that removes the faulty features from each time step:

$$\tilde{\mathrm{x}}_i = \phi(\mathrm{x}_i) = \mathrm{x}_i[:, \{1,2,\ldots,D\} \setminus \mathcal{F}].$$

The new input $\tilde{\mathrm{x}}_i$ is then passed to a modified Transformer model f_θ trained on the same architecture but adapted to the reduced feature dimension:

$$\hat{y}_i = f_\theta(\tilde{\mathrm{x}}_i).$$

The model parameters θ are learned by minimizing the categorical cross-entropy loss on the training dataset:

$$\mathcal{L}(\theta) = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c} \log \hat{y}_{i,c},$$

Where $y_{i,c} \in \{0,1\}$ is the ground truth one-hot vector and $\hat{y}_{i,c}$ is the predicted softmax probability.

Evaluation is performed under varying numbers of faulty sensor $|\mathcal{F}| = \{0,5,10,\ldots,200\}$, and the model is retrained for each fault configuration accordingly. The overall classification performance is assessed using macro-averaged metrics such as accuracy, precision, recall, and F1-score.

This method structurally avoids the influence of faulty sensor elements by removing them from the input representation altogether. Importantly, it assumes access to accurate fault localization, which can be realized in practice through self-diagnosis routines or external fault detection modules in robotic systems.

This approach provides a simple yet effective way to adapt the model to specific fault scenarios, especially when real-time correction is not feasible. It lays the groundwork for scalable robustness in robotic tactile systems.

V. Fault-Robust Classification via Online Correction of Abnormal Sensor Values

In real-world scenarios, it is often impractical to know in advance which sensor elements are faulty or to retrain a new model every time sensor faults occur. To address this limitation, I propose a lightweight, runtime correction strategy that replaces abnormal sensor values with neutral values, thereby mitigating their impact without altering the model architecture or requiring prior fault localization.

A. Abnormal Value Detection

In tactile sensor arrays, typical sensor readings are bounded within a range, e.g., [0, 5] Newtons. However, faulty sensors may output persistently abnormal values such as 0 (indicative of open circuits) or 5 (indicative of short circuits). These values significantly differ from normal distributions and can be regarded as outliers in context.

To correct such abnormal values during inference, I define a simple detection rule based on fixed thresholds:

- Values equal to 0.0 or 5.0 are considered abnormal.
- All other values are left unchanged.

Formally, for each input sample $\mathbf{x}_\mathrm{i} \in \mathbb{R}^{T\times D}$, the corrected sample $\mathbf{x}_i^{corr}$ is defined as:

$$\mathbf{x}_i^{corr}[t,d] = \begin{cases} v_{\mathrm{neutral}} & \text{if } \mathbf{x}_\mathrm{i}[t,d] \in \{0.0, 5.0\} \\ \mathbf{x}_\mathrm{i}[t,d] & \text{otherwise} \end{cases},$$
$$\forall t \in [1,T], \forall d \in [1,D].$$

Here, v_{neutral} is a constant scalar chosen to minimize distortion. Based on empirical evaluation, a value near the median of the initial value or normal range (e.g., $v_{\mathrm{neutral}} = 0$ or 2.5) may be effective.

B. Runtime Integration and Advantages

This correction is applied at inference time, just before feeding data into the pre-trained classification model f_θ. No retraining or architectural changes are needed:

$$\hat{y}_i = f_\theta(\mathbf{x}_i^{corr}).$$

This strategy offers several practical benefits:

Scalability: It can be applied to any trained model without modifying the training pipeline.

Low computation cost: The replacement rule is simple and deterministic.

No fault localization required: Unlike Section IV, this method does not rely on prior identification of faulty sensor indices.

C. Limitations

The effectiveness of this approach depends on the ability to distinguish between naturally occurring edge values and true faults. Overcorrecting normal signals that happen to be near 0 or 5 can degrade performance. Moreover, since all abnormal values are replaced with a single constant, the corrected data may lose informative structure, especially if multiple sensor elements are faulty.

Despite these limitations, this method provides a robust fallback mechanism when no fault labels or diagnostic tools are available, making it suitable for deployment in autonomous systems where runtime adaptability is critical.

VI. Experimental Results

A. Environment

All experiments were conducted on a Linux-based workstation with GPU acceleration. The system was configured as follows:

- Operating System: Ubuntu 20.04LTS
- Python: version 3.11.8
- TensorFlow w/ Keras API: 2.19.0
- CUDA: 12.5

- cuDNN: 9.3

This environment enabled efficient training and evaluation of Transformer-based models on large-scale time-series tactile sensor datasets.

B. Classification Target Application Description

The experiments in this study are centered on a practical and industrially relevant application of tactile sensing: the inspection and quality control of food products, specifically cheese. This application was selected to evaluate the ability of the proposed system to detect subtle physical variations that are difficult to identify using visual modalities alone.

- The classification task is formulated as a 7-class tactile recognition problem:
- Label 0: No object grasped (i.e., free space or failed grasp).
- Label 1: Normal, defect-free cheese products.
- Labels 2–6: Various abnormal products, each representing different defect types such as:
  - Embedded foreign materials,
  - Internal voids or cavities,
  - Surface deformations,
  - Variations in hardness or consistency.

These defects were intentionally introduced to simulate realistic quality issues, including both external irregularities and internal inconsistencies not observable by external cameras. For instance, materials with altered internal density or embedded foreign matter can exhibit similar visual appearances but differing tactile pressure distributions.

To detect such nuanced characteristics, a high-resolution tactile sensor array was employed. The array captures rich time-series pressure distributions during robotic grasping, enabling force-based analysis of object integrity. This approach reflects a realistic use case in industrial automation, where machine learning-based tactile perception can support or augment visual systems to ensure reliable product quality assurance and food safety.

C. Input Data Preparation and Data Preprocessing

In this study, a high-resolution tactile sensor array was employed, consisting of resistive-type force sensors arranged in two 11×10 grids, yielding a total of 220 sensor elements. These tactile sensors output raw analog voltage signals, which were digitized by a microcontroller and transmitted to the host system as digital values. However, the raw digital values do not directly represent physical force and must be converted to meaningful units for downstream machine learning tasks.

Each sensor element was individually calibrated using a reference force measurement system. Specifically, known forces in the range of 0 to 5 N were applied using a ZTA-20N digital force gauge and an MX2-500N motorized test stand (IMADA Co., Ltd.). For each element, a lookup table was created to map raw digital values to corresponding physical forces. These calibration tables were used to perform linear interpolation during preprocessing, enabling precise conversion of incoming sensor data into Newtons while compensating for sensor-to-sensor variability due to manufacturing inconsistencies.

After calibration, each force measurement was normalized to the range $[0.0, 1.0]$ by dividing by the maximum measurable force of 5.0 N. This ensured that all 220 sensor elements shared a consistent scale, thereby improving the stability and convergence of the neural network training.

Each data sample consisted of 100 consecutive time frames, with each frame containing 220 normalized force values. These sequences were stored in CSV format and labeled according to the corresponding object class. To prepare the data for training with fixed-length inputs, each 100-frame sequence was segmented into 10 non-overlapping sub-sequences, each containing 10 frames, resulting in input tensors of shape $(10, 220)$ for each training sample.

Tactile data were collected across seven object classes (Label 0 to Label 6), including both normal (e.g., correctly formed cheese products) and abnormal cases (e.g., contamination, deformation, and internal cavities). For each class, 200 distinct sequences were recorded, introducing small variations in object placement and contact force to enhance diversity. In total, the dataset comprised:

- 1,400 labeled CSV files, and
- 14,000 segmented sequences of 10-frame tactile data.

This dataset was subsequently used for model training, validation, and testing, as described in Section VI. Importantly, all preprocessing steps were completed prior to any fault injection or model training, ensuring a clean, consistent input structure for both baseline and fault-aware learning scenarios.

To rigorously evaluate the classification model while avoiding data leakage, the dataset was split into three distinct subsets:

- Training set: 80% of the preprocessed data samples
- Validation set: 20% subset of the training data, used internally for hyperparameter tuning and early stopping
- Test set: Remaining 20% of the original data, reserved for final performance evaluation

Each split was performed at the file level to ensure that data from the same original sequence (e.g., a single 100-frame sample) did not appear across different subsets. This approach prevents any overlap or redundancy between training and evaluation data, ensuring a fair assessment of generalization capability.

Importantly, all data used in training and validation were collected under normal, fault-free operating conditions. No sensor failures or anomalies were present in these subsets. The model thus learned to recognize the intrinsic characteristics of each class based solely on clean tactile input.

Sensor fault conditions were introduced only during the testing phase, after the model was fully trained. Synthetic fault injection was applied exclusively to the test set, simulating open-circuit or short-circuit conditions. This separation guarantees that the model had no prior exposure to faulty

Figure 2. Recognition performance comparision for fault free, open fault, short fault cases.

sensor data during training, thereby enabling an unbiased evaluation of robustness under previously unseen failure conditions.

D. Impact Analysis of Open and Short-Type Sensor Faults in Tactile Object Classification Using Transformer-Based Models

The classification results under various sensor fault conditions are summarized in Figure 2. In the notation used, "OF" denotes Open Faults, where faulty sensor outputs are replaced with 0 N, simulating open-circuit disconnections. Conversely, "SF" refers to Short Faults, where outputs are fixed at 5 N, simulating short-circuit saturation. 5 N corresponds to the maximum measurable range of the tactile sensor. The numeric suffix indicates the number of faulty sensors. All results represent averages over multiple evaluation runs to ensure robustness.

Under fault-free conditions, the Transformer-based model achieved high performance across all evaluation metrics (accuracy, precision, recall, and F1-score). In fact, the model trained in this configuration was successfully deployed in a real-time inference system, where it demonstrated reliable operation and consistent classification accuracy.

When sensor faults were introduced, performance degradation varied significantly depending on the fault type. Open Faults (OF), where sensor values are zeroed out, caused only a moderate decline in performance. Even with several faulty sensor elements, the model retained relatively high accuracy, suggesting that the model exhibits inherent tolerance to such failure modes.

In contrast, Short Faults (SF) had a much more severe impact. Fixing sensor values to 5 N resulted in a dramatic drop in classification performance. In the SF5 condition, for example, the accuracy dropped to approximately 13%, which is near the theoretical accuracy for random guessing in a 7-class classification task ($1/7 \approx 14.29\%$). This indicates that the model loses meaningful discriminative power even under a small number of short-type faults.

Figure 3 shows the average spatial force distribution during object grasping under three conditions: normal (no faults), with 10 open-type sensor faults (0 N), and with 10 short-type faults

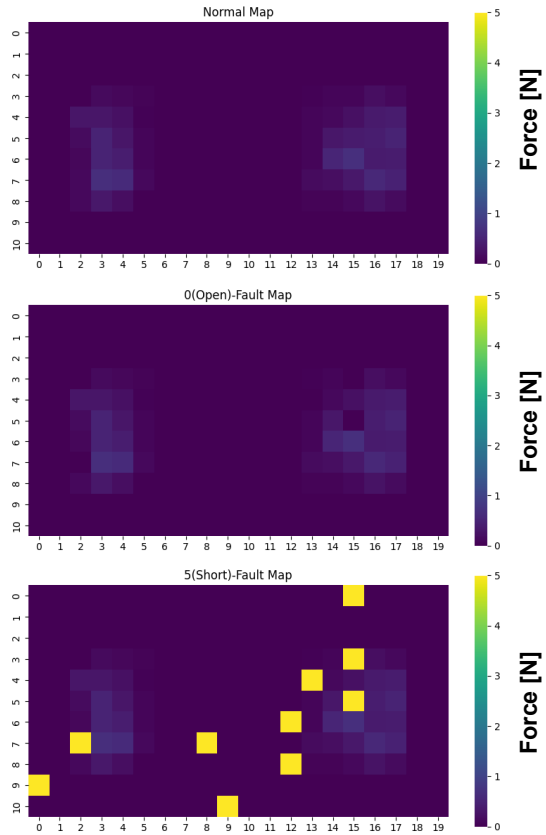

Figure 3. Comparison of spatial force distribution maps under different sensor fault conditions. From top to bottom: (1) normal condition with no sensor faults, (2) condition with 10 open-type sensor faults (values fixed at 0 N), and (3) condition with 10 short-type sensor faults (values fixed at 5 N). The heatmaps visualize the average force distribution across the tactile sensor array, highlighting the impact of fault types on spatial sensing characteristics.

(5 N). As shown in the maps, the influence of open-type faults is relatively small, producing only slight gaps in the distribution. In contrast, short-type faults introduce unnaturally high force values that strongly distort the overall force pattern. Because these 5 N spikes are much larger than the surrounding normal forces, they create disproportionately large ΔN values, which in turn dominate the attention weights in the Transformer. This effect is especially severe in my gentle-grasping scenario, where a 5 N reading is clearly abnormal and easily overwhelms the model's ability to focus on the true contact region.

These findings highlight the critical need for fault-robust strategies in tactile sensing systems. Specifically, Short Faults introduce strong, non-informative activations that overwhelm the attention mechanism, distorting the internal feature representations. This motivates the development of methods that can anticipate, detect, or compensate for such degradation during training or inference.

E. Results of Fault-Aware Training with Simulated Sensor Failures

I present here the evaluation results of the proposed fault-aware training strategy, in which the model is exposed to sensor faults during training. Figure 4 summarizes the results

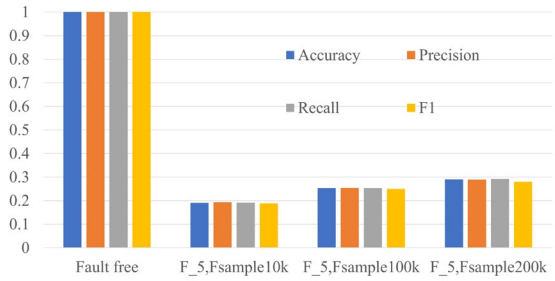

Figure 4. Recognition performance evaluation for fault pretrained model.

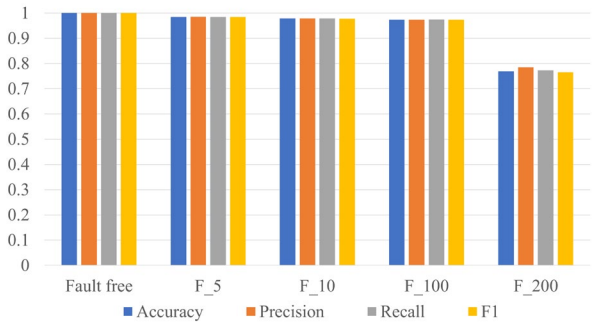

Figure 5. Recognition performance evaluation for fault posttrained model.

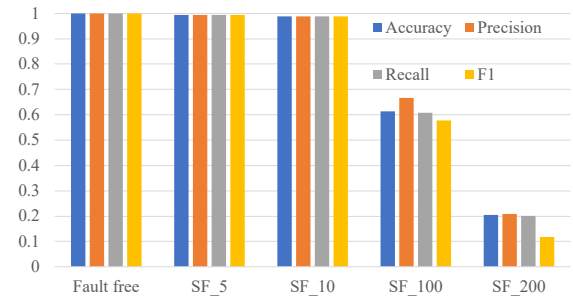

Figure 6. Recognition performance evaluation for fault correction model.

under simulated fault conditions. In the figure, F_5 denotes an evaluation scenario where five sensor faults (either 0 N or 5 N) are randomly injected during testing, and Fsample refers to the number of fault-injected samples added to the training data. For example, "10k" represents 10,000 such synthetic samples, while "100k" and "200k" indicate increasingly larger datasets with more diverse degradation patterns.

Although the number of additional samples is large relative to the original dataset of 14,000 samples, it still covers only a small fraction of all possible fault configurations. Nevertheless, the experimental results demonstrate a clear and consistent trend: incorporating a greater volume and variety of fault patterns into training leads to improved model robustness under fault conditions.

This indicates that even with limited coverage of fault combinations, the model is capable of learning more generalized representations that are resilient to certain classes of sensor degradation. However, further work is needed to design efficient sampling and augmentation strategies that can scale to more complex or variable real-world fault scenarios.

F. Results of Model Adaptation Based on Identified Sensor Faults

Figure 5 presents the classification performance of the sensor-fault-aware model adaptation strategy, which assumes prior knowledge of faulty sensor locations. These locations are explicitly excluded from the input feature space before training. The notation used follows the same convention as before; for example, F_5 and F_200 indicate fault scenarios with 5 and 200 faulty sensors, respectively, including both open-circuit (0 N) and short-circuit (5 N) types.

The results clearly demonstrate that identifying and removing the faulty sensor elements significantly improves classification performance. Even under severe degradation scenarios with hundreds of faulty inputs, retraining the model using only the reliable sensor elements allows the system to retain high accuracy. This confirms that the model can structurally adapt to reduced input dimensions while maintaining robustness.

However, this method entails several practical challenges. First, it requires accurate fault localization, which may not always be achievable in real time. Second, preprocessing to exclude faulty features must be implemented as part of the data pipeline. Finally, the model must either be retrained or fine-tuned for each new fault configuration, which may introduce latency or computational overhead in resource-constrained systems.

Despite these limitations, the method offers a strong foundation for model-level fault mitigation when prior diagnostic information is available. It is particularly well-suited for applications in which sensor degradation accumulates gradually over time and periodic recalibration or retraining is feasible.

G. Results of Fault Correction via Abnormal Value Replacement

Figure 6 presents the results of the proposed fault-correction method, in which abnormal sensor readings caused by short-type faults (i.e., stuck at 5 N) are replaced with neutral values simulating open-type faults (i.e., 0 N). This transformation effectively mitigates the disruptive effect of high-valued faulty readings without requiring prior knowledge of fault locations or model retraining.

The results indicate that this replacement strategy achieves classification performance comparable to the open-fault (OF) condition with the same number of faulty sensors. Specifically, replacing short faults with zeros significantly improves the model's robustness, maintaining stable accuracy even as the number of corrected sensor elements increases. This result suggests that abnormal-value suppression functions as an effective attention regularization mechanism, preventing faulty sensors from dominating the Transformer's feature extraction process. This aligns with previous observations that open faults have a much smaller impact on model performance than short faults, which tend to overwhelm the input distribution.

It is important to note, however, that this correction method is highly specific to the tactile sensor used in this study—namely, a resistive-type array where the signal value range spans from 0 to 5 N. In such systems, sensor faults often manifest as digital outputs stuck at these extreme values, allowing for straightforward identification and transformation. In contrast, different tactile sensors or sensing principles (e.g., capacitive, optical, piezoelectric) may exhibit fault behaviors with different characteristics and ranges. Therefore, the definition of what constitutes an "abnormal value" and what value it should be converted to must be carefully tailored to each sensor type and application context.

Despite this limitation, the proposed method offers a lightweight and practical inference-time solution for real-world deployments, especially when real-time performance is essential and sensor degradation cannot be completely prevented.

VII. Conclusion

This study addressed the critical challenge of sensor faults in tactile object recognition systems, particularly in scenarios involving hardware-specific failure modes such as open circuits and short circuits, which are common in resistive-type sensor arrays. My analysis showed that even a small number of short-type faults can collapse classification performance to chance level, underscoring the importance of incorporating fault-aware design into learning-based tactile perception.

To tackle this issue, I proposed and evaluated three complementary strategies, each corresponding to a different level of fault knowledge and system integration. First, I introduced a fault-aware training method that injects synthetic fault patterns (0 N or 5 N) during learning, enabling partial robustness without requiring prior knowledge of fault locations. Second, I developed a model adaptation strategy that explicitly identifies and removes faulty sensor elements, allowing the model to be retrained or fine-tuned with a reduced but reliable input space. Third, I proposed a lightweight inference-time correction method that converts abnormal values such as 5 N to low-impact values like 0 N without modifying the model architecture.

Among these approaches, the correction method proved especially effective in scenarios where faults can be detected at runtime, but model retraining is impractical. Importantly, all three methods are grounded in hardware-specific fault behavior characteristic of resistive tactile sensors, where damage often results in fixed output values. Thus, the proposed strategies are applicable to a wide range of real-world robotic systems employing similar sensor architectures.

In summary, these findings demonstrate that incorporating sensor-fault considerations into training, model adaptation, and inference-time processing is essential for achieving robust tactile object recognition in practical environments. In future work, I will explore integrating these methods, extending the fault model to include noise, drift, and intermittent failures, and developing automatic fault-detection mechanisms that leverage temporal and spatial consistency within the tactile data.

References

[1] S. Luo, J. Bimbo, R. Dahiya, and H. Liu, "Robotic Tactile Perception of Object Properties: A Review," Mechatronics, Vol. 48, pp. 54-67, 2017.

[2] R. S. Dahiya, G. Metta, M. Valle, and G. Sandini, "Tactile Sensing—From Humans to Humanoids," IEEE Transactions on Robotics, Vol. 26, No. 1, pp. 1–20, 2010.

[3] S. Ji, Y. Kim, and H. Son, "Directions Toward Effective Utilization of Tactile Skin: A Review," IEEE Sensors Journal, Vol. 13, No. 11, pp. 4121-4138, 2013.

[4] J. Tang, Y. Li, Y. Yu, Q. Hu, W. Du, and D. Lin, "Recent Progress in Flexible Piezoelectric Tactile Sensors: Materials, Structures, Fabrication, and Application," Sensors, Vol. 25, No. 3, pp. 1-28, 2025.

[5] J. M. Gandarias, A. J. García-Cerezo, and J. M. Gómez-de-Gabriel, "CNN-based Methods for Object Recognition with High-Resolution Tactile Sensors," IEEE Sensors Journal, Vol. 19, No. 16, pp. 6872-6882, 2019.

[6] L. Chen, Y. Zhu, and M. Li, "Tactile-GAT: tactile graph attention networks for robot tactile perception classification," Scientific Reports, 27543, pp. 1-13, 2024.

[7] Z. Ding, G. Chen, Z. Wang, and L. Sun, "Adaptive visual–tactile fusion recognition for robotic systems," Frontiers in Neurorobotics, Vol. 17, pp. 1-14, 2023.

[8] W. Zheng, H. Liu, D. Guo, and F. Sun, "Robust tactile object recognition in open-set scenarios," Frontiers in Neuroscience, Vol. 16, pp. 1-14,2022.

[9] M. Baioumy, C. Pezzato, R. Ferrari, C. H. Corbato, and N. Hawes, "Fault-tolerant Control of Robot Manipulators with Sensory Faults using Unbiased Active Inference," 2021 European Control Conference, pp. 1119-1125, 2021.