Detecting Failed Robot Demonstrations: Contact Force, Proprioceptive Logs, and a Label Leak Through the Gripper

Snigdha Chandan Khilar Asit Sahoo Swaraj Dhar

19 November 2026

Abstract

Imitation-learned manipulation policies are trained on teleoperated demonstrations, some of which fail. We set out to test a specific hypothesis: that contact force, which a camera does not capture, is the signal that identifies failed demonstrations. This paper reports the full sequence of experiments, including the ones that overturned the hypothesis. (i) A frozen vision encoder recovers only 1.9% and 8.8% of execution-specific wrist force on two contact-rich tasks, and the naive version of this measurement overstates it severely ($R^2 = 0.79$ on a task where time alone gives 0.76). (ii) On REASSEMBLE, a public assembly dataset with 1,161 insertion attempts and 246 labelled failures, a detector built on seven statistics of wrist force ranks failed insertions at an average precision (AP) of 0.38 against a base rate of 0.22, survives controls for part identity and duration, transfers to unseen part families, and is not improved by sequence models. (iii) The same failures are ranked at AP 0.72 by the robot's own proprioceptive logs with no force sensor; 48 of the 50 highest-ranked insertions are real failures. Most of this skill is the final state of the gripper: a single rule on it catches 59% of failed insertions at 81% precision. (iv) For the remaining failures, which end normally as far as the gripper shows, neither force nor the logs improves on knowing which part is being inserted. We conclude that success labels in teleoperated data largely record what the operator saw, that failure detectors on such data must be compared against a gripper end-state baseline, and that this dataset cannot support the claim that force reveals otherwise hidden failures. We describe the resulting method, a lightweight logs-only demonstration audit, and state plainly what it does not do.

1 Introduction

Robot manipulation policies are now commonly trained by imitation. An operator teleoperates the robot through a task many times and a policy is fitted to the recordings. Not every recording is a good example: parts slip, pegs jam, connectors fail to seat. A policy trained on such recordings can learn the failure along with the skill, so teams that collect demonstrations have an interest in finding the bad ones.

This work began with a concrete hypothesis about how to find them. Whether a screw is correctly tightened or stripped, or a connector seated or forced, is decided by the force applied, and that force is not directly visible in an image. If so, tools that judge demonstrations from video would miss an important class of failure, and a detector reading the wrist force/torque sensor would catch it. The hypothesis has three parts: (a) contact force is not recoverable from the image; (b) contact force predicts failure; (c) it predicts failures that other available signals do not.

We tested the three parts in order on public data. Parts (a) and (b) held. Part (c) did not: the robot's own proprioceptive logs, and in particular the final state of the gripper, identify failed insertions far better than force does, and force adds nothing once they are known. The reason lies in what the labels are. In the dataset we study, an operator narrates each action and the success annotation is validated afterwards, so a labelled failure is one a person observed, and the operator's response to it is written into the gripper trace.

We report every experiment, in the order it was run, because the path matters as much as the endpoint. At several points a result looked like success and passed the controls we had thought to apply. A reader benchmarking failure detection on teleoperated data is likely to meet the same points.

The paper makes five contributions.

1. A measurement of how much execution-specific contact force a frozen vision encoder recovers on three manipulation tasks, with a nested-model analysis that separates sensing from task repeatability (Section 5.1).
2. A force-based failure detector for insertions, with controls for part identity and duration, a test of transfer across part families, a comparison with sequence models, and an early-warning analysis (Sections 5.2–5.7).
3. The finding that proprioceptive logs alone rank failed insertions at AP 0.72, that most of this is the gripper's final state, and that a one-line rule recovers 59% of failures at 81% precision (Sections 5.8–5.9).
4. A negative result: for failures that the gripper does not reveal, force does not improve on part identity and duration (Section 5.9).
5. A logs-only audit procedure that follows from these results, with its requirements and limits (Section 6).

2 Related work

Failure detection for imitation policies. Most work detects failures of a deployed policy at run time. FAIL-Detect, for example, treats it as sequential out-of-distribution detection using only successful training data [5]. Our setting is different: we audit the *demonstrations* before training, where the "policy" is a human operator.

Curating demonstrations. Demo-SCORE trains a classifier on the outcomes of policy rollouts to filter unreliable demonstrations [6]. It needs a trained policy and rollouts. The audit studied here needs neither, but relies on labelled examples of failed actions.

Force in manipulation learning. A growing body of work brings force and torque into policy learning: large datasets with wrist force/torque such as RH20T [7] and REASSEMBLE [1], and policies that use force or joint-torque feedback such as FoAR [8], FACTR [9] and Adaptive Compliance Policy [10]. That literature asks whether force helps a policy act. We ask the narrower question of whether force helps identify a failed demonstration, and find that on labelled-outcome data the answer is confounded by what the labels record.

3 Data

REASSEMBLE. All failure-detection experiments use REASSEMBLE [1] in its LeRobot v3.0 copy (`robot-lev/reassemble`) [4]. A Franka FR3 arm is teleoperated with a haptic device to assemble and disassemble a board of pegs, gears, electrical connectors and a bolt. The copy contains 149 recordings and 1,433,406 frames at 30 Hz. The authors divide each recording into action segments and annotate each segment with success or failure. Operators narrate their actions while teleoperating, and the segment and success annotations are validated manually afterwards; the authors' examples of failure are an object slipping from the gripper or the gripper missing the object [1].

After removing 5 segments shorter than 10 frames and 9 with a flat force trace, 4,831 segments remain, 516 of them failures (10.7%). Table 1 gives the breakdown. Among insertions, failure rates vary by part family from 9.5% (large gear) to 36.1% (bolt): square peg 23.4%, round peg 16.2%, BNC 34.1%, bolt 36.1%, Ethernet 27.8%, USB 30.0%, waterproof connector 12.5%, D-SUB 15.9%, large gear 9.5%, medium gear 12.9%, small gear 10.7%.

Table 1: Action segments in REASSEMBLE after filtering. A further 289 segments are marked as no action.

Action	Segments	Failures	Failure rate
Insert	1,161	246	21.2%
Remove	1,100	147	13.4%
Pick	1,192	100	8.4%
Place	1,089	22	2.0%

Logged channels. Each frame holds wrist force and torque (each in two variants, plain and `.base`), a 36-value state vector, separately named joint positions (7), gripper position (2) and end-effector pose (7), and a 9-value commanded action. By correlation we identified 16 of the 36 state values as copies of the named joint, gripper and pose columns; the remaining 20 match no named column and are consistent with the joint velocities, joint efforts and end-effector velocity the authors describe. We refer to the state, named columns and action together as the *proprioceptive logs*: 61 channels as stored, 45 distinct. Wrist force in this copy is sampled at the video frame rate, 30 Hz.

Split. Whole recordings are held out: 45 of 149, drawn once with a fixed seed, giving 1,495 held-out segments with 184 failures. For insertions this is 716 training and 445 held-out segments, 98 of the latter failures (base rate 22.0%). The same split is used throughout.

Force-recoverability datasets. Section 5.1 additionally uses two LeRobot datasets from a second source, `cloudfan/intern-pour-lerobot` (pouring, 51 episodes) and `cloudfan/intern-screw-lerobot` (screwing), which record wrist force alongside video.

4 Methods

4.1 Force recoverability from vision

Sixty frames are sampled evenly from each episode and encoded with a frozen DINOv2 ViT-S/14 [2]. A ridge regression maps features to the magnitude of the measured wrist force at the same instant, with whole episodes held out (30%). Scripted tasks make force predictable from the time within the episode, so we fit three nested models in the same way: *phase* (a smooth basis over normalised time), *vision* (image features) and *both*. We report the share of execution-specific force that vision recovers,

$$\text{share recovered} = \frac{R^2_{\text{both}} - R^2_{\text{phase}}}{1 - R^2_{\text{phase}}}. \tag{1}$$

Chance is estimated by circularly shifting the force trace within each episode and refitting (15 shifts). Two further controls are a cross-episode pairing, in which each episode's frames are paired with another episode's force, and a positive control that predicts arm pose from the same features.

4.2 Failure detection

Each action segment is summarised by seven statistics of a scalar trace: mean, maximum, standard deviation, final value, largest single step, summed squared step, and overall slope. The *force detector* applies these to the wrist force magnitude (7 features). The *logs* model applies them to every proprioceptive channel and to its frame-to-frame change (14 per channel, 854 features). Two reduced variants are the *end state* (the final value of each channel, 61 features) and the *gripper rule* (a threshold on a logistic score of the final gripper values, fitted on training recordings).

All models are class-balanced, L_2-regularised logistic regressions on standardised features. The force detector uses a fixed regularisation strength; larger models choose theirs on validation recordings held out of the training set. Vision baselines pool frozen image features over 16 frames per segment (mean and standard deviation), reduce them by PCA, and use the same classifier. Sequence models are described in Section 5.5.

4.3 Evaluation and controls

Rankings are scored by average precision (AP), whose no-skill value is the base rate, and by the fraction of failures among the 50 highest-ranked segments (P@50). The first pooled experiment also reports the Matthews correlation coefficient (MCC). Intervals are 95% bootstrap intervals. Except in Table 3 and the pooled column of Table 5, which resample segments, they resample whole held-out recordings (2,000 resamples), because segments within a recording share an operator and a setup; comparisons between models are paired on the same resamples.

A ranking can exceed the base rate without sensing how an action went. We use the following controls: a *part and duration* baseline (one-hot part family, or exact part copy, plus log duration), with models credited for what they add to it; a *within-part* analysis against a floor obtained by shuffling scores within each part; a *wrist-leak* check that correlates every proprioceptive channel with every force and torque axis; and a *five-round* evaluation in which every recording is tested once, used where a single split is too small.

4.4 Validation on synthetic data

Every analysis was first executed on synthetic datasets with a known answer. The first, used before any real data, is a simulated world in which a visible property sets a fragility threshold and an invisible applied force decides whether it is exceeded; the pipeline recovers the planted ordering (vision partially informative, force decisive). Later analyses were run on synthetic LeRobot-format repositories built to produce each possible verdict, for example one in which failures appear only in the end state and one in which the only signal is a force channel concealed inside the state vector. An analysis was run on real data only after it returned the correct verdict on both.

5 Experiments and results

5.1 How much contact force does a camera recover?

Table 2 and Figure 1 give the result. On the two contact-rich tasks, vision recovers 1.9% and 8.8% of execution-specific force. Pouring is higher because its wrist force is dominated by the weight of a draining container, which the image shows; phase alone explains 76% of it.

Table 2: Force recoverability on held-out episodes. On pouring the vision gain exceeds the chance floor of +0.014 (95th percentile of 15 time-shifted refits); we do not report floors for the other two datasets.

Dataset	Task	R^2 phase	Vision adds	Share recovered
intern-pour	Pouring	0.759	+0.049	20.2%
intern-screw	Screwing	−0.000	+0.019	1.9%
REASSEMBLE	Assembly, disassembly	0.006	+0.087	8.8%

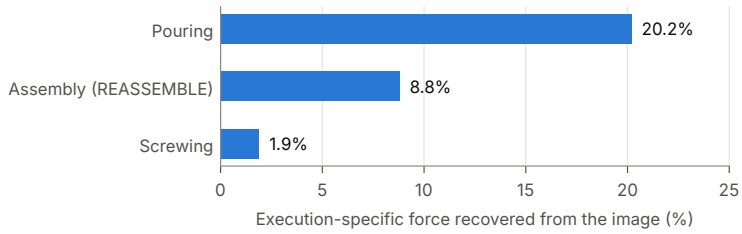

Figure 1: Share of execution-specific wrist force recovered from single frames by a frozen vision encoder with a linear readout.

The naive measurement is misleading. Regressing force on image features and reporting R^2 gives 0.793 on pouring. Pairing each episode's frames with a different episode's force still gives 0.631, and time alone gives 0.759; vision and time together give 0.809. Because every pour follows nearly the same script, image features predict force by encoding the moment in the task. Raw R^2 of force from vision on scripted tasks measures task repeatability. The positive control confirms the features are informative: the best pose channel is predicted at R^2 0.958, 0.965 and 0.942 on the three datasets. A sweep of the phase basis from 8 to 256 functions left the vision gain on pouring unchanged.

5.2 Vision against force for failure prediction, all actions pooled

The first failure experiment pooled all action types. Three classifiers were compared on the 1,495 held-out segments (base rate 12.3%): A, pooled vision features; B, vision features plus the seven force statistics; C,

force statistics alone (Table 3).

Table 3: Pooled failure prediction (DINOv2 ViT-S/14 features). MCC with 95% bootstrap intervals over segments.

Model	MCC	95% interval
A: vision	+0.025	[-0.026, +0.074]
B: vision + force	+0.126	[+0.072, +0.184]
C: force only	+0.119	[+0.067, +0.175]

Adding force to vision improved MCC by +0.101 against a permutation floor of +0.013; a label-shuffle control averaged −0.007. Ranked by score, the vision model placed 6% failures in its top 50 with DINOv2 features and 20% with ResNet-18 [3] features, neither distinguishable from the base rate, while vision plus force placed 32%. The same vision features predicted the best arm-pose channel at $R^2 = 0.962$ (median over channels 0.474).

At the time we read this as evidence that video-based scoring misses failures that force reveals. **That reading is not supported.** The vision baseline averages features over frames and so cannot represent how an action ends, which Section 5.8 shows is where the signal lies. Table 3 shows that this particular vision baseline was at chance; it does not show that a camera cannot see these failures.

5.3 Per-action structure

Failure rates differ sharply by action (Table 1), so a pooled model gains much of its apparent skill by recognising the action type. Restricting to insertions and training on insertions only, the force detector reaches AP 0.380 on the 445 held-out insertions, whereas the pooled-trained detector scores 0.227 on the same insertions, barely above the 0.220 base rate. The paired interval for the gain from per-action training is [+0.035, +0.275]. All later experiments are insertions only.

5.4 Transfer across part families

To test whether the force detector depends on having seen the part, the 11 part families were grouped into four folds. For each fold the detector was trained once with and once without that fold's families and tested on held-out insertions of those families (Table 4).

Table 4: Force detector on part families included in or removed from training (AP on held-out insertions).

Held-out families	n	Failures	Base	Seen	Unseen
Square peg	101	23	0.228	0.341	0.319
Round peg, small gear	126	18	0.143	0.203	0.206
BNC, medium gear, USB, waterproof	109	26	0.239	0.679	0.553
Bolt, D-SUB, Ethernet, large gear	109	31	0.284	0.528	0.500
Weighted	445	98	0.220	0.431	0.389

Skill above base is +0.210 [+0.151, +0.314] when the families are seen and +0.168 [+0.114, +0.281] when they are not; the loss is +0.042 [+0.018, +0.060] and 80% of skill is retained [71%, 93%]. The force detector therefore transfers across part geometries on this robot. This test was run for the force detector only.

5.5 Sequence models against summary statistics

Seven statistics discard the shape of the force trace. We compared them with models that read the three-axis trace resampled to 128 steps: per-axis statistics (29 parameters, to separate the effect of extra channels from temporal modelling), a 1D convolutional network (9,026 parameters) and YB-Mixer, a phase-only spectral mixer [11] (9,892 parameters). Networks were trained for 60 epochs with the epoch selected on validation recordings, and averaged over seeds (Table 5).

No model improves on seven statistics with an interval that excludes zero, in either setting, and the two networks are not separable from each other (paired interval [-0.097, +0.079] on insertions). At 30 Hz and a

Table 5: Sequence models against seven statistics. Differences are paired against the seven-statistic model. Network figures vary slightly between GPU runs.

	All actions pooled (base 0.123)		Insertions only (base 0.220)	
Model	AP	Difference	AP	Difference
Seven statistics	0.201		0.380	
Per-axis statistics	0.228	+0.027 [-0.003, +0.058]	0.387	+0.007 [-0.051, +0.059]
1D CNN	0.235	+0.035 [-0.011, +0.086]	0.348	−0.032 [-0.144, +0.080]
YB-Mixer	0.198	−0.003 [-0.045, +0.039]	0.339	−0.042 [-0.139, +0.037]

few hundred training failures, modelling the trace offers no measurable benefit. A 50 ms transient spans 1.5 samples at this rate, so the comparison cannot speak to high-frequency signatures.

5.6 Is the force detector reading contact, or the part and the clock?

Parts fail at different rates and failed insertions differ in length, and one of the seven statistics grows with duration. Table 6 gives the control.

Table 6: Force detector against part and duration (held-out insertions, base 0.220).

Model input	AP	95% interval	P@50
Duration only	0.251	[0.190, 0.333]	0.24
Part family only	0.282	[0.236, 0.340]	0.34
Part + duration	0.311	[0.256, 0.390]	0.32
Force, seven statistics	0.380	[0.272, 0.522]	0.48
Part + duration + force	0.477	[0.384, 0.581]	0.62

Force adds +0.166 [+0.047, +0.275] to part and duration. Force alone is not shown to exceed part and duration (+0.069 [-0.072, +0.219]). Within a single part family, the weighted skill of the force detector is +0.211 against a shuffled floor of +0.055 (95th percentile +0.100); per-family APs range from 0.208 (round peg, base 0.157) to 1.000 (USB, base 0.276, 8 failures). By these controls the detector carries information beyond part identity and duration.

5.7 Early warning

A live guardrail would need the signal before the action ends. Given the first 25%, 50% and 75% of each insertion, the force detector scores 0.228, 0.262 and 0.299, each below the whole-insertion value with a paired interval that excludes zero. Because a fraction of the action is known only in hindsight, we also trained on the first t seconds and scored only insertions still running at t (Figure 2). The median insertion lasts 15.0 s.

For the first 8 seconds AP is 0.22 to 0.26 against a base of 0.20 to 0.22. At 12 s, with 64% of insertions still running and a median of 6.2 s remaining for those that will fail, AP is 0.326 against 0.226 (skill [+0.043, +0.200]). An alarm on the top 20% of scores at that point would be right 32% of the time and catch 28% of failures. The signal is late and weak.

5.8 Proprioceptive logs without the wrist sensor

We then asked whether failures could be ranked with no force sensor, expecting a negative answer that would establish force sensing as a requirement. The result was the opposite (Table 7, Figure 3).

The logs exceed the force detector by +0.344 [+0.223, +0.439]. Because a result of this size is as likely to be a leak as a finding, we checked three explanations.

A concealed wrist sensor. No proprioceptive channel correlates above 0.95 with any force or torque axis (0 of 61).

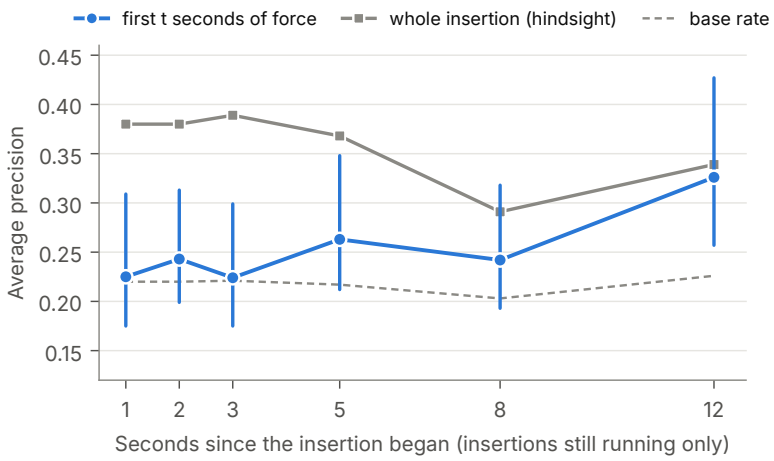

Figure 2: Force detector given the first t seconds, on insertions still running at t (point and 95% interval), with the whole-insertion detector on the same insertions.

Table 7: Ranking held-out insertions from proprioceptive logs (base 0.220).

Model input	Channels	AP	95% interval	P@50
Part copy + duration	–	0.286	[0.229, 0.367]	0.30
Joint positions	7	0.336	[0.268, 0.415]	0.42
End-effector pose	7	0.444	[0.352, 0.564]	0.52
Gripper position	2	0.665	[0.583, 0.742]	0.90
State vector	36	0.706	[0.617, 0.784]	0.90
Commanded action	9	0.712	[0.645, 0.784]	0.92
All logs	61	0.724	[0.650, 0.791]	0.96
end state only	61	0.662	[0.583, 0.733]	0.96
end minus start only	61	0.568	[0.477, 0.656]	0.72
first half only	61	0.371	[0.291, 0.480]	0.46
All logs + force	61 + force	0.731	[0.655, 0.797]	0.96

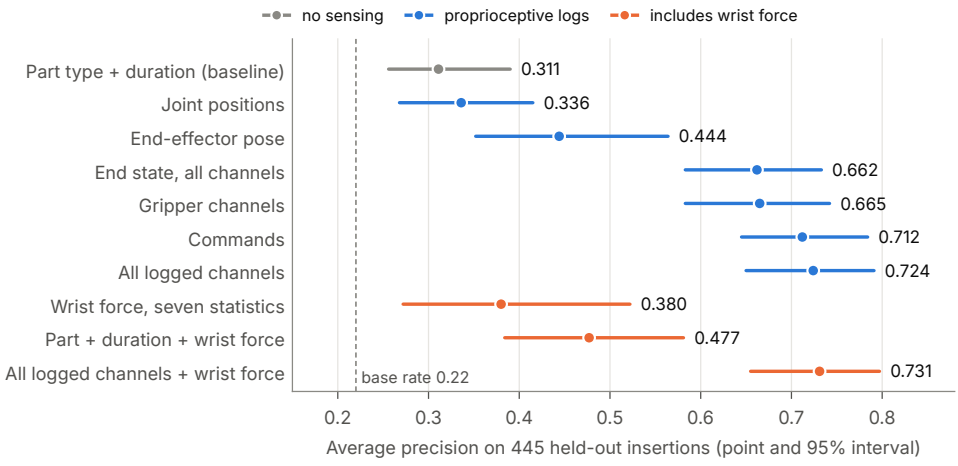

Figure 3: Average precision on held-out insertions by model input.

Part identity. Arm position could reveal which copy of a part is being inserted. Against a baseline that knows the exact part copy and the duration, the logs add +0.425 [+0.322, +0.515].

The end state. The final value of each channel alone reaches 0.662, which is 88% of the full model's skill above base; the first half of the action gives 30%. The ten single features that best separate failures on training data are all the gripper's position or its command, mostly the final value, which is lower in failures; each scores 0.61 to 0.65 on held-out data by itself.

The first two explanations are excluded and the third holds. The model reads how the gripper finishes. This is consistent with how the labels were produced (Section 3): a labelled failure is one the operator saw, and the gripper trace records what the operator did next. We have not verified against video what the lower final value corresponds to physically.

5.9 The gripper rule, and failures that look finished

The rule. A threshold on the final gripper value, fitted on training recordings, flags 55 of the 445 held-out insertions; 46 are failures (84% precision, 47% of failures). Evaluated in five rounds over all 1,161 insertions it flags 181, of which 81% are failures, and catches 146 of 246 (59%).

What remains. The original hypothesis concerned failures that are *not* otherwise apparent. We therefore removed every insertion the rule flags and asked whether anything ranks the failures that remain. On the single split this leaves 390 held-out insertions with 52 failures; force alone scores 0.293 against a base of 0.133, but adding force to part and duration gives +0.031 [-0.065, +0.122], and the combined model scores below force alone, indicating an underpowered test (54 training failures). We repeated it in five rounds with regularisation fixed in advance (Table 8, Figure 4).

Table 8: Insertions not flagged by the gripper rule: 980 insertions, 100 failures, base 0.102. Each scored by a model that never saw its recording.

Model input	AP	Skill above base
Part + duration	0.177	+0.075 [+0.045, +0.148]
All logs	0.190	+0.088 [+0.060, +0.162]
Force, seven statistics	0.224	+0.122 [+0.076, +0.206]
Part + duration + force	0.185	+0.083 [+0.064, +0.155]
All logs + force	0.210	+0.108 [+0.079, +0.194]

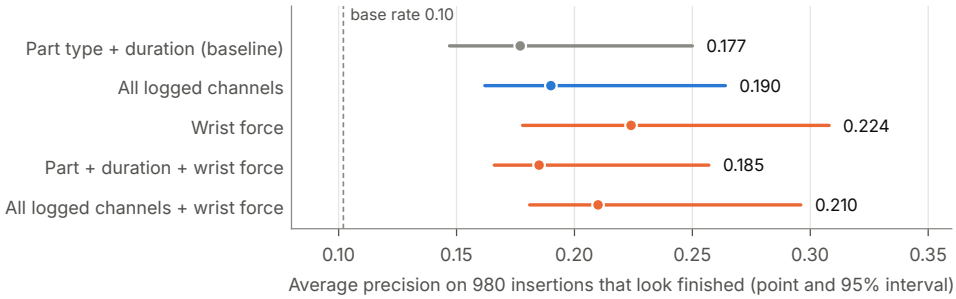

Figure 4: Insertions that look finished. Intervals are those of skill above base, drawn from the base rate.

Every input ranks these failures above chance, and none improves on part and duration: force added to part and duration gives +0.008 [-0.048, +0.076]; force alone against part and duration +0.047 [-0.046, +0.134]; force added to the logs +0.020 [-0.004, +0.056]; the logs against part and duration +0.013 [-0.044, +0.075]. Reviewing the top fifth of the ranking finds 29%, 33% and 35% of the hidden failures for part and duration, force, and both, where chance would find 20%. Across all held-out insertions, adding force to the end state changes AP by +0.004 [-0.008, +0.016].

The part of the hypothesis that mattered, that force reveals failures not otherwise apparent, is not supported on this dataset.

6 Discussion

What the labels measure. A success label in teleoperated data records an outcome that a person observed. The operator acts on that observation, and the action is logged. A classifier trained on such labels can therefore score well by reading the operator's response, with no sensitivity to how the contact

went. In REASSEMBLE the response appears in the gripper. In another protocol it may appear elsewhere, for instance in a retry, a pause, or the length of the segment.

Implications for benchmarking. The force detector of Sections 5.2–5.7 beat its base rate, survived controls for part and duration, held within part families, transferred to unseen families, and was robust to the choice of model. On that evidence it would ordinarily be reported as a working failure detector, and force as a valuable modality. It was largely re-deriving, less accurately, what two gripper channels show. We recommend that any failure detector evaluated on teleoperated data be compared against (i) a part-and-duration baseline and (ii) a proprioceptive end-state baseline, and that results be reported separately for failures the end state does and does not reveal.

What this does and does not say about force. Section 5.1 stands: a camera, under the readout tested, recovers little contact force. What is not shown is that this costs anything for identifying failed demonstrations. The force representation here is coarse, 30 Hz and seven statistics, and the labels do not mark actions that completed while damaging or stressing a part. A fair test of the original hypothesis needs data with such labels and force at a rate that preserves transients. We have not found a public dataset that provides both.

The audit as a tool. The positive result is practical. A logs-only audit, reading tabular data without video or a GPU, ranks failed insertions with 48 of its top 50 correct, and a single interpretable rule recovers most of that. Reading and segmenting all 149 recordings takes under a minute on a free cloud notebook. Its use is for datasets collected without per-action success labels, or for checking labels that exist. It requires proprioceptive logs with gripper state, action boundaries, the action type, and some labelled examples. It finds failures an operator noticed. It does not find the others.

7 Limitations

- **One dataset, one robot, one protocol.** All failure results come from REASSEMBLE. The gripper leak depends on how its operators ended actions.
- **Physical meaning unverified.** That a lower final gripper value means an unreleased or lost part was inferred from the logs and not checked against video.
- **Vision was not tested fairly for failure detection.** The pooled vision baseline could not represent the end of an action. No claim is made about what a camera can detect.
- **Given segmentation and labels.** The audit uses the dataset's own action boundaries and learns from its labels. The number of labels needed was not measured.
- **Transfer of the logs model untested.** Section 5.4 concerns the force detector only. Nothing was tested across robots.
- **Reuse of one split.** Many analyses share the same 45 held-out recordings. The gripper effect is too large for this to matter; small differences in Section 5.9 are why that analysis was repeated across all recordings.
- **Force recoverability.** One encoder with a linear readout on single frames; chance floors reported for one of three datasets; 15 shifts per floor; 60 frames per episode.
- **Earlier intervals.** The intervals in Table 3 and the pooled column of Table 5 resample segments, not recordings, and are narrower than the later convention would give.
- **No deployment.** The audit has not been used by any team on its own data.

8 Conclusion

We set out to show that contact force identifies failed robot demonstrations that other signals miss. A camera does recover little of that force, and a force detector does rank failures above chance and pass standard controls. But on the one public dataset where the claim could be tested, the robot's own logs rank

the same failures almost twice as well, most of the signal is the gripper's final state, and for the failures the gripper does not reveal, force adds nothing measurable. The useful outcomes are a simple logs-only audit, a baseline that failure-detection work on teleoperated data should include, and a clear statement of what data would be needed to test the original idea properly.

Reproducibility

All experiments ran on free Kaggle sessions against public data. Three notebooks cover the paper: `step3-force-predictability-v5` (Section 5.1; GPU), `step6-transfer-and-bakeoff-v2` (Sections 5.4–5.5; GPU) and `step7-phase0-remaining-v4` (Sections 5.3, 5.6–5.9; no GPU), with `step4-failure-prediction` for Section 5.2. The failure-detection notebooks fix their seeds and verify on load that the split reproduces 4,831 segments, 1,495 held out and 184 held-out failures. The notebooks are not yet in a public repository.

References

[1] D. Sliwowski, S. Jadav, S. Stanovcic, J. Orbik, J. Heidersberger, D. Lee. REASSEMBLE: A Multimodal Dataset for Contact-rich Robotic Assembly and Disassembly. arXiv:2502.05086, 2025.

[2] M. Oquab et al. DINOv2: Learning Robust Visual Features without Supervision. arXiv:2304.07193, 2023.

[3] K. He, X. Zhang, S. Ren, J. Sun. Deep Residual Learning for Image Recognition. CVPR, 2016.

[4] Hugging Face. LeRobot. https://github.com/huggingface/lerobot.

[5] C. Xu et al. Can We Detect Failures Without Failure Data? Uncertainty-Aware Runtime Failure Detection for Imitation Learning Policies. arXiv:2503.08558, 2025.

[6] A. S. Chen et al. Curating Demonstrations using Online Experience. arXiv:2503.03707, 2025.

[7] H.-S. Fang et al. RH20T: A Comprehensive Robotic Dataset for Learning Diverse Skills in One-Shot. https://rh20t.github.io, 2023.

[8] Z. He et al. FoAR: Force-Aware Reactive Policy for Contact-Rich Robotic Manipulation. arXiv:2411.15753, 2024.

[9] J. J. Liu, Y. Li et al. FACTR: Force-Attending Curriculum Training for Contact-Rich Policy Learning. arXiv:2502.17432, 2025.

[10] Y. Hou et al. Adaptive Compliance Policy: Learning Approximate Compliance for Diffusion Guided Control. https://adaptive-compliance.github.io, 2024.

[11] S. C. Khilar. YB-Mixer. Manuscript, 2026.