Teaching Agents to Teach Themselves

Mining an agent's own past runs into reusable skills – and nearly doubling its success on the hardest support tickets


Yue Ran Kang


Colton Mikolajczyk

Faculty Advisor: James Butler · **Microsoft Team:** Sahil Bhatnagar, Chhaya Methani

15.089 Analytics Capstone · MIT Master of Business Analytics · Summer 2026

01 BUSINESS MOTIVATION

Every failed agent run escalates to a human and erodes trust in automation. Our goal: agents that **improve in production and become reliable**, giving customers **confidence to deploy Microsoft Copilot Studio agents**. Mining an agent's own transcripts delivers that improvement at **near-zero cost, with no retraining**.

02 THE CHALLENGE

Enterprise AI agents resolve employee support tickets thousands of times a day – and **start every run knowing nothing about the last one.** On deliberately hard tickets, barely one task in three succeeds.

104	120	32.5%
support queries 80 / 24 train-test split	matched runs per experiment condition	baseline success 39 of 120 runs pass

> **Can an agent read its own transcripts and write itself a skill that makes the next run better?**

HEADLINE RESULT

59.2%

task success on held-out hard tickets

~~32.5%~~ → **+26.7 points**

95% confidence interval on the gain: +15.0 to +39.2 points

The recipe: a **workflow skill** mined from **successful runs**, given **with worked examples**. All eight mined skills beat the baseline.

03 HOW WE BUILD A SKILL

a meta-skill is a skill whose job is to write other skills

1 RUN – Attempt each task five times, logging every tool call as a trajectory.

2 LABEL – **Successes** pass 8 of 10 runs; **Goldilocks** land in the messy middle, 3 to 7.

3 MINE – A meta-skill reads the trajectories and drafts a new skill, in one of two styles.

4 EQUIP – Give the agent the skill, with or without worked examples.

5 MEASURE – Re-run held-out hard tickets against the untouched baseline.

WORKFLOW SKILL
Checklist-style guidance: what to do, in what order, and when to stop.

ONTOLOGY SKILL
A map of entities, their relationships, and the actions each permits.

04 RESULTS – EVERY MINED SKILL BEAT THE BASELINE

Neobank Support · held-out hard tickets · 120 runs per bar

Worked examples separate a good workflow skill from the best one. Bar labels: gain over baseline in points · whiskers: 95% confidence interval.

Workflow beats ontology here
Support tickets are procedural – order of operations wins.

Examples help workflow, not ontology
+12.5 points for workflow; ontology stays flat.

Ontology ignores run quality
Both mining sets land on the same 42.5%.

Failure is cheap training data
Messy Goldilocks runs still earned +13 to +14 points.

05 WHAT THE AGENT ACTUALLY LEARNED

WORKFLOW · FINDING 1

Successful runs reveal the shape of a right answer – and when to stop

HELD-OUT TICKET

"Several keys on my laptop keyboard are sticking and not registering."

PASS · 4 / 5

```
status: "hold"
approval_required:
"no"
approval_status:
"not_required"
asset_id: "VDB-HW-
10890"
```

FAIL · 1 / 5

```
status: "solved"
resolution_category:

"resolved_by_requester"
```

The skill encodes the answer shape plus an exit rule: **stop once the requester will report back.**

WORKFLOW · FINDING 2

Naming the request lifecycle fixes the order of tool calls

Discover facts › Create ticket › Perform action › Update state › STOP

HELD-OUT TICKET

"I spilled coffee on my laptop. It shut off and won't turn back on."

The agent walked the lifecycle exactly and **passed 5 of 5 runs** – the lifecycle fixes both **tool-call order** and **when to stop.**

ONTOLOGY · FINDING 1

Relationship learning barely cares whether the run succeeded

42.5% mined from successful runs

42.5% mined from Goldilocks runs

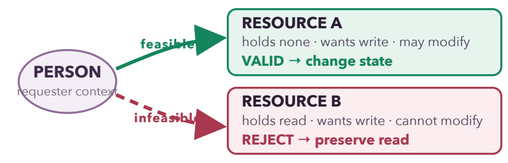
BOTH SOURCES RECOVER THE SAME STRUCTURE

```
Employee → holds  resource: READ
Employee → requests resource: WRITE
Policy + approver → PENDING | DENY
```

Both gain +10.0 points – wrong actions still observe the right relationships, **so half-broken runs stay usable.**

ONTOLOGY · FINDING 2

Mapping what is allowed keeps the agent on feasible paths



HELD-OUT TICKET

"How do I set up VPN to access company systems?"

BASELINE · FAIL – Skipped the access check, closed the ticket as solved.

ONTOLOGY · PASS – Checked access, saw a standard employee, opened an info ticket.

This ticket 1/5 → **3/5** · access & identity tickets 40.0% → **47.3%**

06 SECOND WORKSTREAM – NOT EVERY TICKET NEEDS THE BIG MODEL

can we tell easy from hard before spending a single token?

FINDING 1
The small model matches or beats the frontier model on a slice of tasks.

FINDING 2
Query-text features predict that slice **before the first model call.**

Five routers tested: gradient boosted trees, logistic regression, nearest-neighbours, and two probability blends.

Dataset	Best router	Queries routed	Accuracy retained	Cost saved
Retail	Gradient boosted trees	9.0%	99.7%	**5.5%**
Hotel	Logistic regression	8.7%	98.2%	**7.2%**
Car insurance	Logistic regression	12.9%	95.4%	**8.4%**
Neobank	Minimum probability	1.9%	99.7%	**1.3%**
Consulting	Gradient boosted trees	4.7%	99.2%	**3.9%**

Routing 2–13% of queries saves up to **8.4% of spend** at 95–99.7% accuracy.

07 IMPACT & WHERE THIS GOES NEXT

1.8× the baseline success rate on the hardest tickets

+32 more runs pass per 120, with no model retraining

8.4% peak model spend cut by routing, 95% of accuracy retained

Pick the skill style up front
Predict workflow vs. ontology before mining.

Give ontologies examples too
Example *relationships* may mirror the workflow lift.

Mine selectively
Sort easy from hard queries before mining; unlabeled data lifts least.

Test portability, then hand over
Re-run on GitHub Copilot; the harness goes to the Microsoft team.

Yue Ran Kang · **Colton Mikolajczyk** | Mentored by Sahil Bhatnagar, Chhaya Methani and James Butler

All conditions evaluated on matched held-out tasks · 10 tasks × 5 runs · 120 runs per condition · baseline 39 / 120.