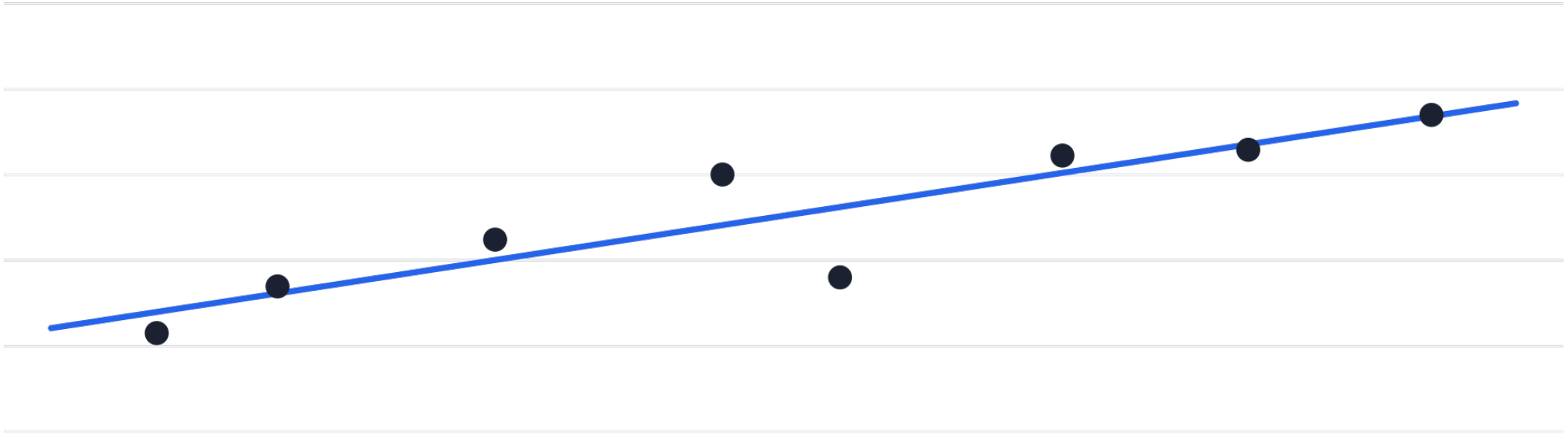
SOURCE #3 OF EVALUATION ERROR

Open-Book Overfitting

You looked at the data. From that moment the data is training data, and the score it gives you is no longer a measurement.

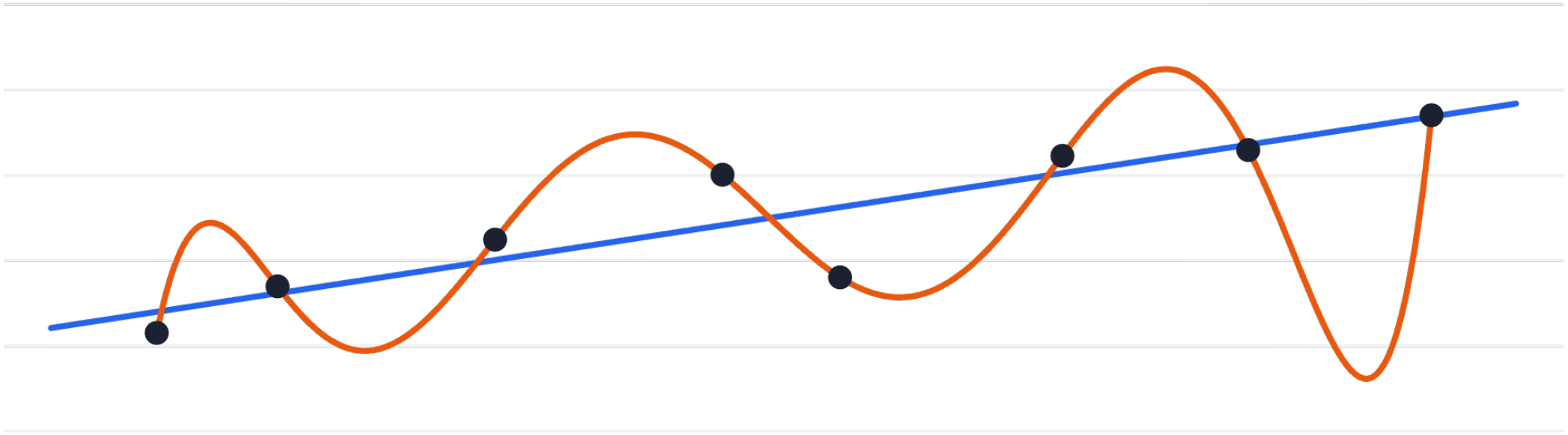
Overfitting in traditional ML (1/3):
We train a model and keep going, going, going



Overfitting in traditional ML (2/3): a curve through every point

Straight line: average error 0.8

Bent curve: average error 0.0, through every point



On the points we can see, the bent curve is perfect.

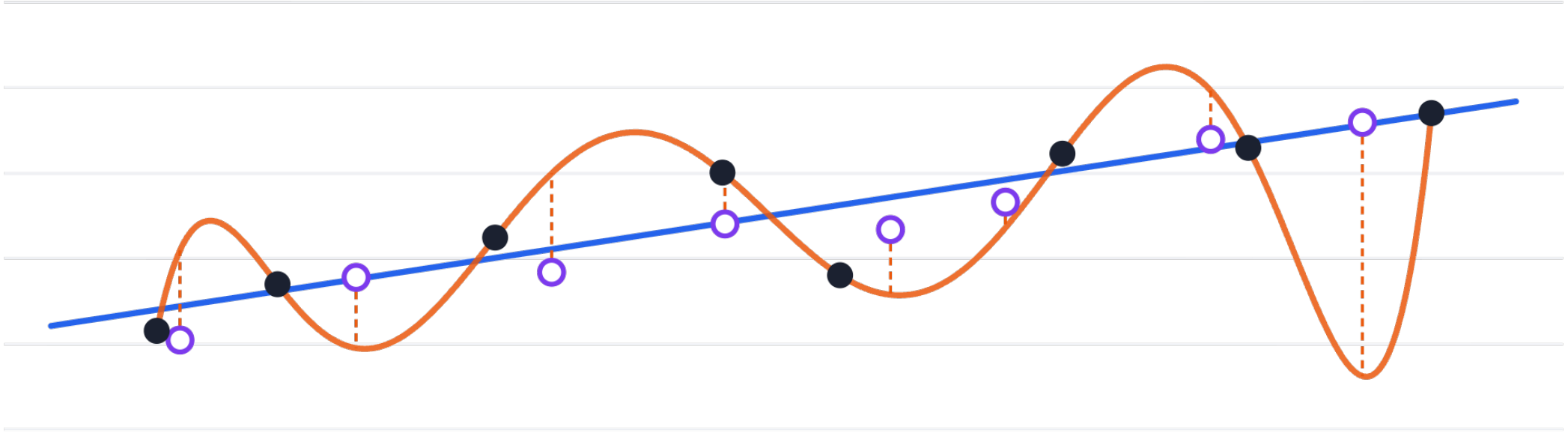
Overfitting in traditional ML (3/3): new points from the same process

Straight line: average error 0.5

Bent curve: average error 2.6

Dark: the 8 points it was fitted to

Hollow: 8 new points



The bent curve memorized the noise in eight particular points. It describes the sample, not the process. That is overfitting, and the cure is the same everywhere: judge the model on points it never saw.

How does that look in gen AI?

Few-shot prompt V1 on five support tickets

PROMPT V1

```
Classify the customer
ticket into exactly one
category: Billing,
Shipping, Returns, Account,
Other.

Examples:
"My invoice shows the wrong
VAT number." → Billing
"When will my order
arrive?" → Shipping
"I'd like to exchange this
for a medium." → Returns
"How do I change my email
address?" → Account

Reply with the category
only.
```

Ticket	True label	V1 says
"I was charged twice for order #4471."	Billing	Billing, correct
"Tracking says delivered but nothing arrived."	Shipping	Shipping, correct
"The shoes are the wrong size, I want to send them back."	Returns	Returns, correct
"I returned a jacket two weeks ago and the refund is still not on my card."	Billing	Returns, wrong
"It's a gift, can you ship it to my sister's address instead of mine?"	Shipping	Account, wrong

Measured on 100 labeled tickets: **80% correct**.

Prompt V2: we refine the categories - and the examples

PROMPT V2

```
Classify the customer ticket into exactly one
category.

Billing: charges, invoices, payments and refunds.
Shipping: delivery, tracking, and where or when an
order is delivered.
Returns: sending an item back or exchanging it.
Account: login, password, profile and contact
details.
Other: anything else.

If a ticket touches two categories, choose the one
that describes what the customer wants to happen
next.

Examples:
"My invoice shows the wrong VAT number." → Billing
"When will my order arrive?" → Shipping
"I'd like to exchange this for a medium." → Returns
"How do I change my email address?" → Account

Reply with the category only.
```

THE TWO TICKETS V1 GOT WRONG

"I returned a jacket two weeks ago and the refund is still not on my card."

V1: Returns, wrong V2: Billing, correct

"It's a gift, can you ship it to my sister's address instead of mine?"

V1: Account, wrong V2: Shipping, correct

SAME 100 TICKETS

90%

Then we ship V2

Accuracy on our 100 labeled tickets, and accuracy on the tickets customers actually send.

PROMPT V1

80%

Not bad. Not good enough.

PROMPT V2, SAME 100 TICKETS

90%

Category definitions and a tie-break rule. Ship it.

CUSTOMERS STILL SEE

80%

Where did the ten points go?

This happens even with the best intentions

And part of the issue is that executives don't ask the question and do not discourage inaccurate reporting

The same loop, elsewhere: a trace analyzer

An LLM reads agent traces and flags the ones where the agent went wrong.
You label 40 traces by hand and start iterating.

1

Run the prompt. It misses a third of the errors.

→

2

Open the misses. Read them. Understand why the model did not flag them.

→

3

Add an instruction: "an empty tool result that is never retried counts as an error."

→

4

Run again. Better. Open the new misses. Repeat five times.

Learning the method, or memorizing past exams

Studies the textbook: learns the method

Past exams **78%** · new exam **76%**

Drills past exams: memorizes the answers

Past exams **97%** · new exam **64%**

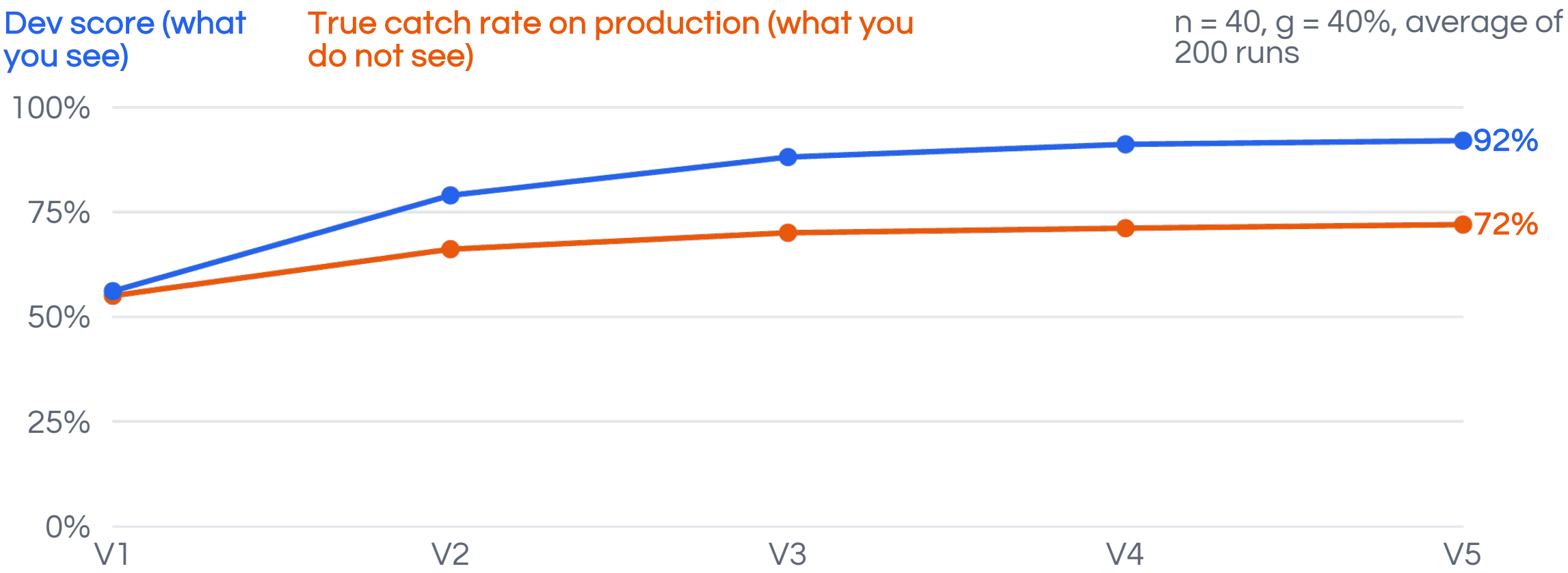
The prompt is the student. The dev set is the pile of past exams, and every patch is one more memorized answer. Production is the new exam.

Open-book hill climbing: the loop we simulate

Ingredient	In the simulation
Failures	30 patterns with a long-tailed frequency: a few are common, most are rare
Prompt v0	Catches each pattern with some probability between 30% and 80%
Dev set	n traces sampled from the same distribution as production
One round	Run on the dev set, read every miss, patch for each one
One patch	Always handles that trace; with probability g also fixes its pattern
What we track	The dev score you would see, and the true catch rate on production you would not

Defaults on the next slides: n = 40, g = 40%, four rounds (V1 to V5), averaged over 200 runs.

The dev score drifts away from the truth



Four rounds of read-and-patch, V1 to V5. You would ship V5 believing you are **20 points** better than you are.

AI hill climb (1/4): start with a plain prompt

PROMPT V1

```
Classify the customer support
ticket into exactly one
category: Billing, Shipping,
Returns, Account, Other.
Reply with the category only.
```

WHAT THE AI OPTIMIZER IS SHOWN: DEV TICKETS V1 GETS WRONG

Dev ticket	Expected
"Forget the exchange, I just want my $110 back on my Amex."	Billing
"Charged $9.95 for delivery but your page said free shipping over $75."	Shipping
"I'm getting order notification emails for someone else's account!"	Shipping
"tracking number hasnt updated in 6 days. still says 'in transit'"	Other
"I entered a wrong birthday at signup so I never get the birthday discount."	Billing

and 2 more

V1 · 19 words · dev **85%** · held-out **83%**

AI hill climb (2/4): a rule for every miss

Rule the AI added to the prompt	Written for this dev ticket
Refunds owed, a refund instead of an exchange, and late-return exceptions are Billing	"Forget the exchange, I just want my $110 back" · "My return window closed yesterday"
Delivery fees and free-shipping thresholds are Shipping	"Charged $9.95 for delivery but your page said free shipping"
Birthday and loyalty perks are Billing, even when the fix is editing the profile	"I entered a wrong birthday at signup"
Notification emails are Shipping, even if described as a privacy concern	"notification emails for someone else's account"
A tracking page stuck on "in transit" is Other	"tracking number hasnt updated in 6 days"

Its own note on the change: *"the gain is in-sample and could cost accuracy on unseen tickets."*

V2 · 378 words · dev 91% · held-out 77%

AI hill climb (3/4): patching the patches

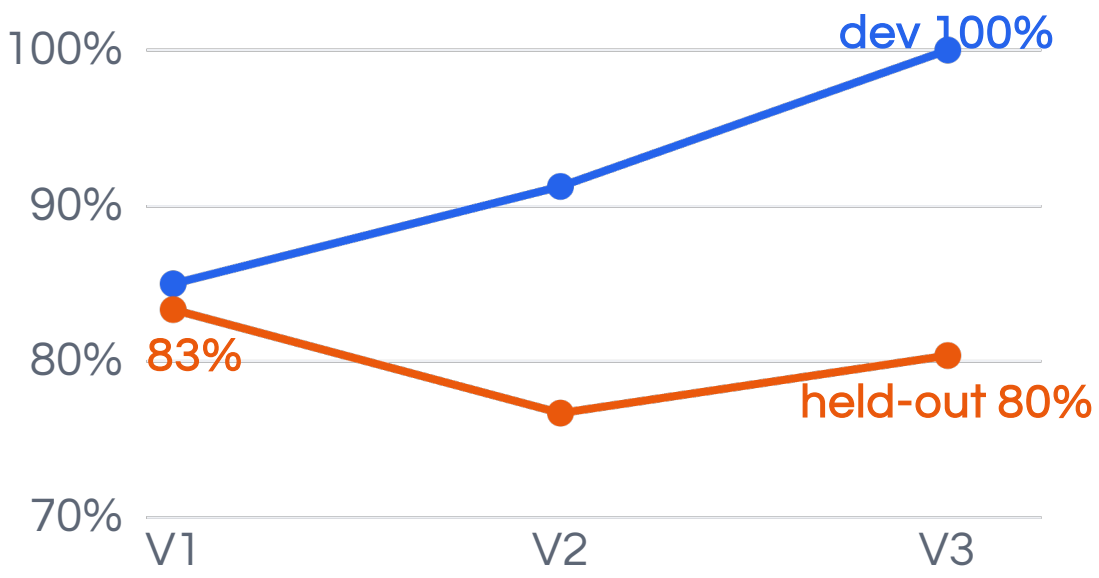
The round-1 rules broke three other dev tickets. The AI adds exceptions to its own rules, plus six worked examples. Compare three of them with the dev tickets they were written for:

Dev ticket	Example the AI added	Label
"The bag handle's stitching came undone on first use. I'd like my money back."	"The zipper broke on first use, I want my money back."	Returns
"Got the wrong size AGAIN … I'd rather not do another exchange. Tell me how to send it back"	"Wrong size again, I don't want another exchange. How do I send it back?"	Returns
"My account says 340 loyalty points, but the receipt … said I earned 100 more."	"My account shows 200 loyalty points but my receipt said I earned 50 more."	Account

The student is memorizing the exercises, with the numbers changed.

V3 · 695 words · dev 100% · held-out 80%

AI hill climb (4/4): perfect on dev, worse on new tickets



The textbook picture, in quality instead of loss

Try it yourself

reliableai.github.io/ai-design-2026-pub/playground/open_book_overfitting.html

Suggested exercise: run the loop with g = 0 and g = 100% and explain, in one sentence each, why the gap is not zero in either case.

Freeze one held-out set, and keep growing the test set with an LLM judge

1 Hold out a test set you never iterate on

Tune against the dev set; run the held-out set only to check the result.

2 Grow the test set continuously, labeled by an LLM judge

Label aggressively, so new cases arrive faster than a prompt can memorize them.

3 Use human labels only to calibrate the judge

Human ground truth checks that the judge agrees with people; it is not the test set.

OPEN-BOOK OVERFITTING

Thank you

X - LinkedIn