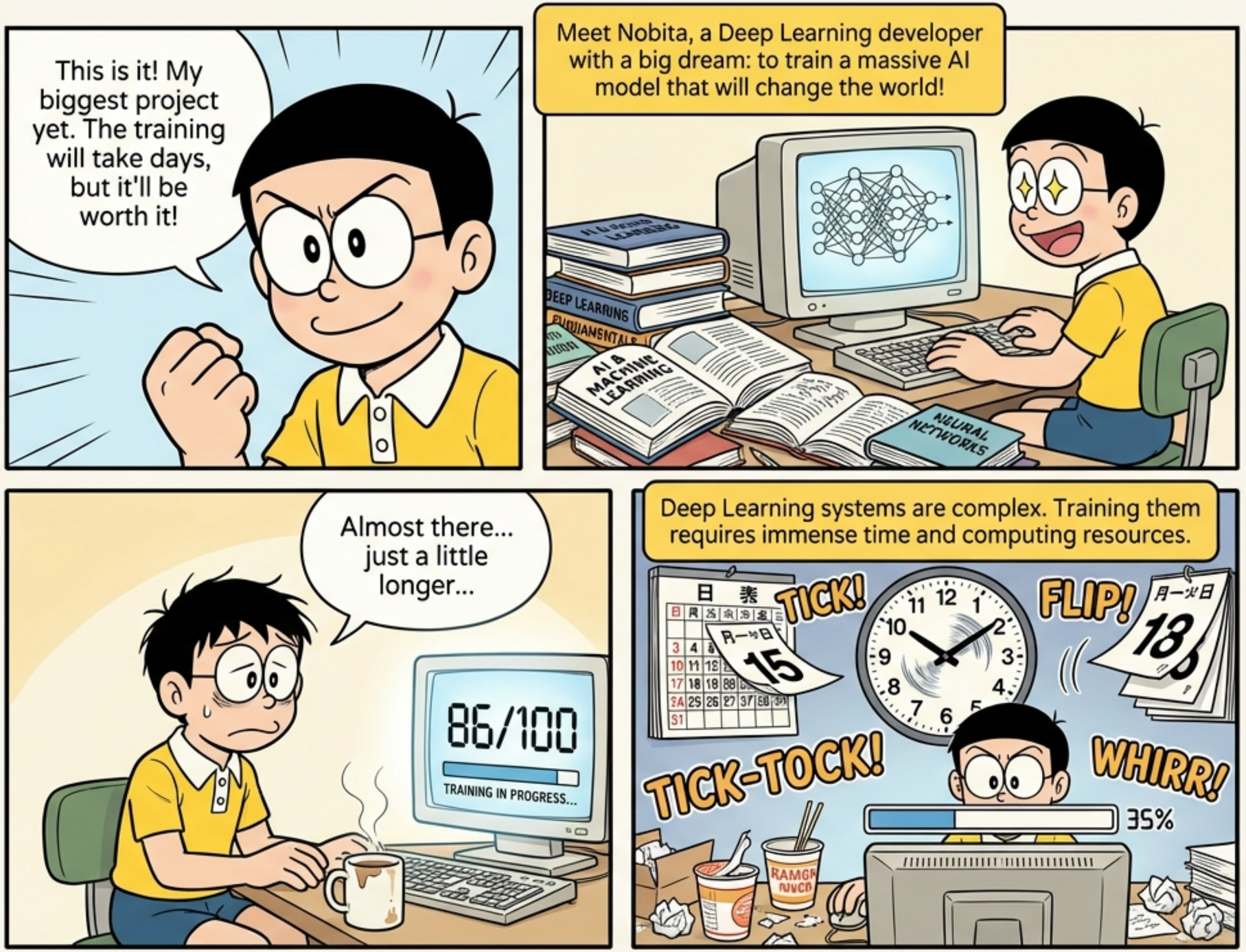
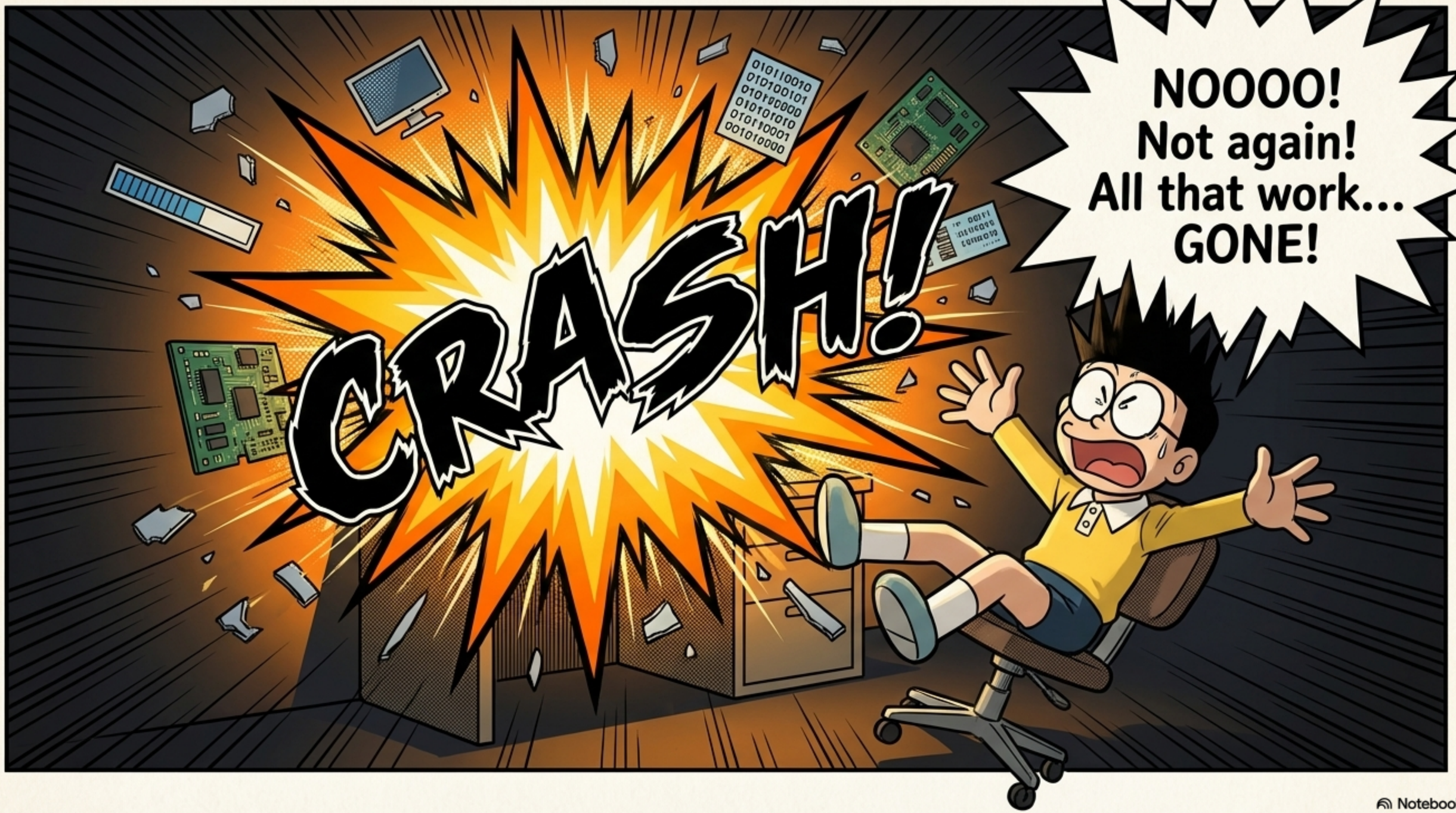
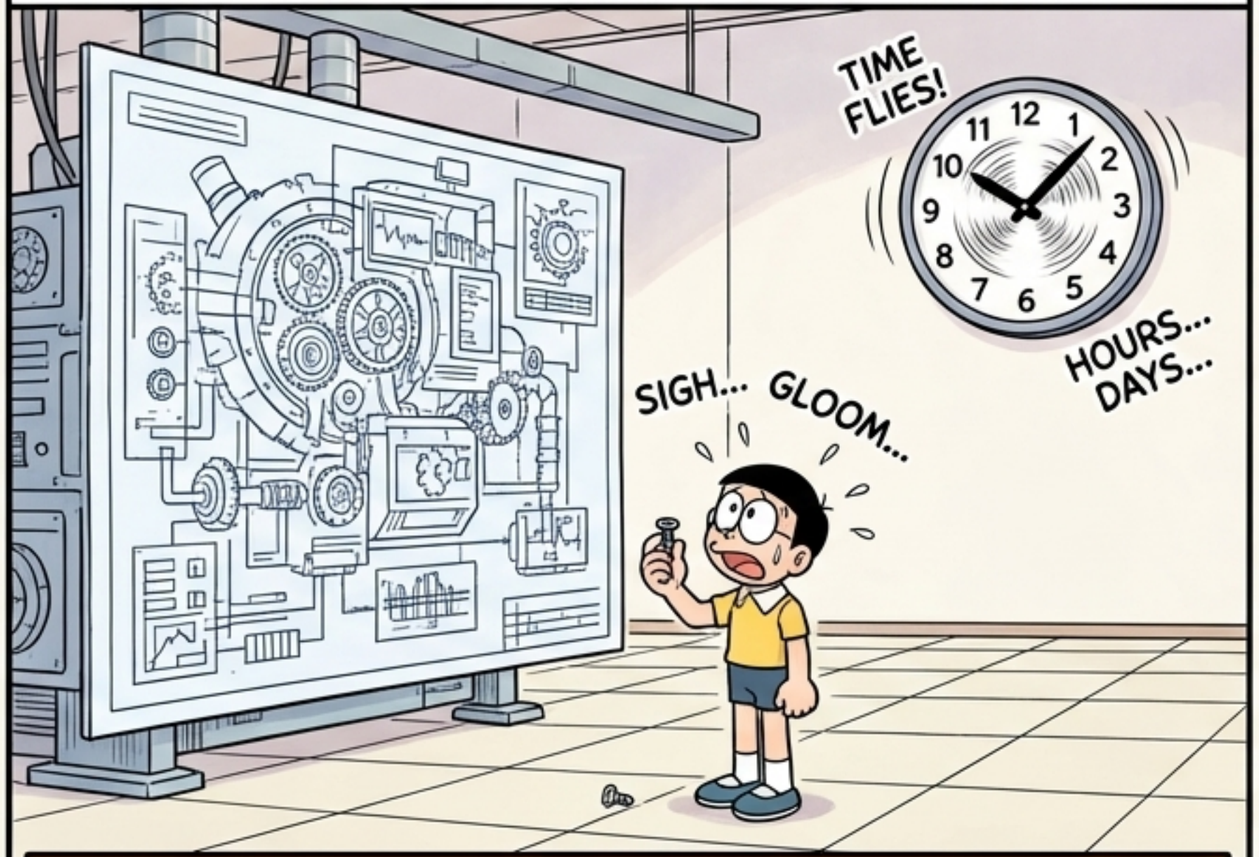

This is it! My biggest project yet. The training will take days, but it'll be worth it!
Meet Nobita, a Deep Learning developer with a big dream: to train a massive AI model that will change the world!
AI & MACHINE LEARNING
NEURAL NETWORKS
Almost there... just a little longer...
86/100
TRAINING IN PROGRESS...
Deep Learning systems are complex. Training them requires immense time and computing resources.
TICK!
FLIP!
TICK-TOCK!
WHIRR!
35%



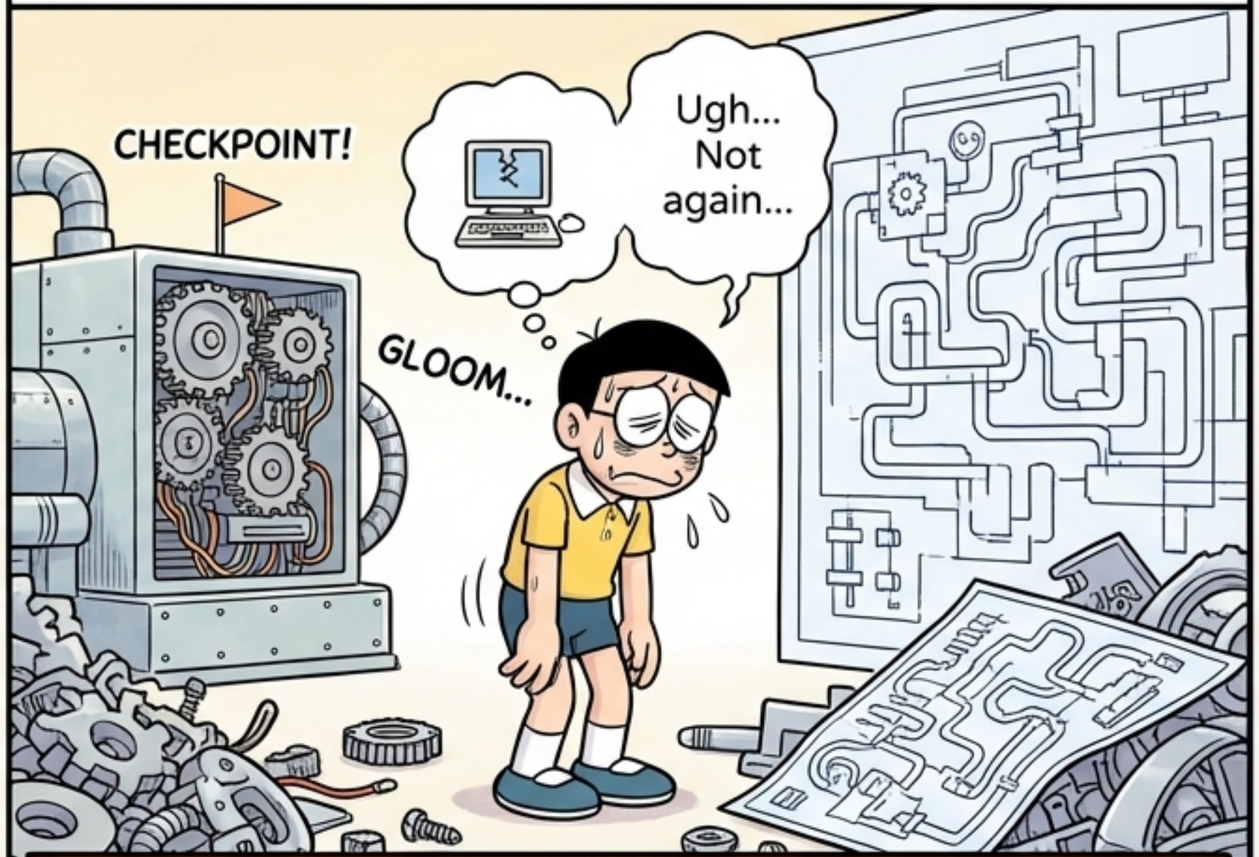
The Old Ways are Painful and Slow.

Paradigm 1: Restart-from-Scratch.

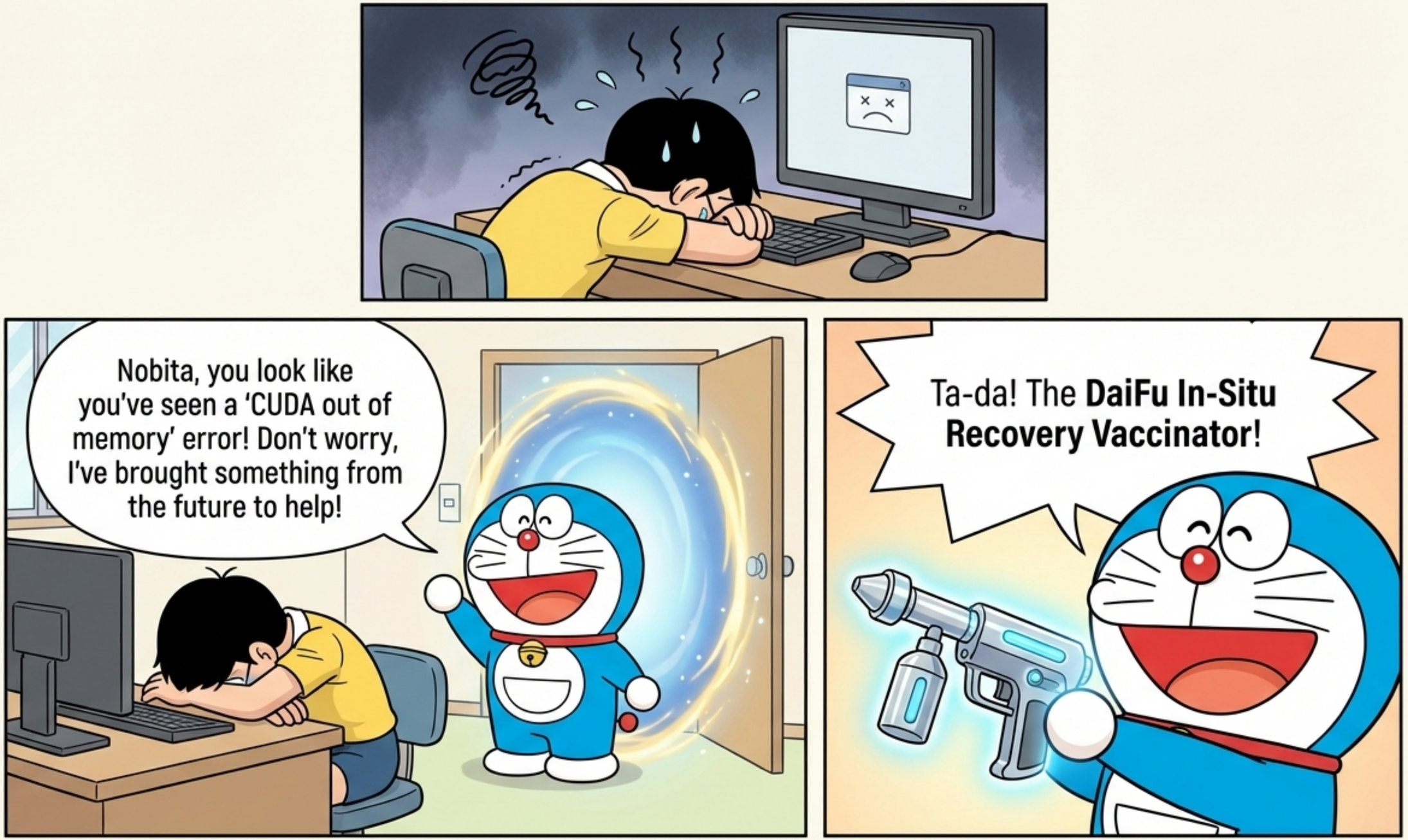


Restarting from the beginning means re-doing hours, or even days, of computation. This wastes huge amounts of resources.

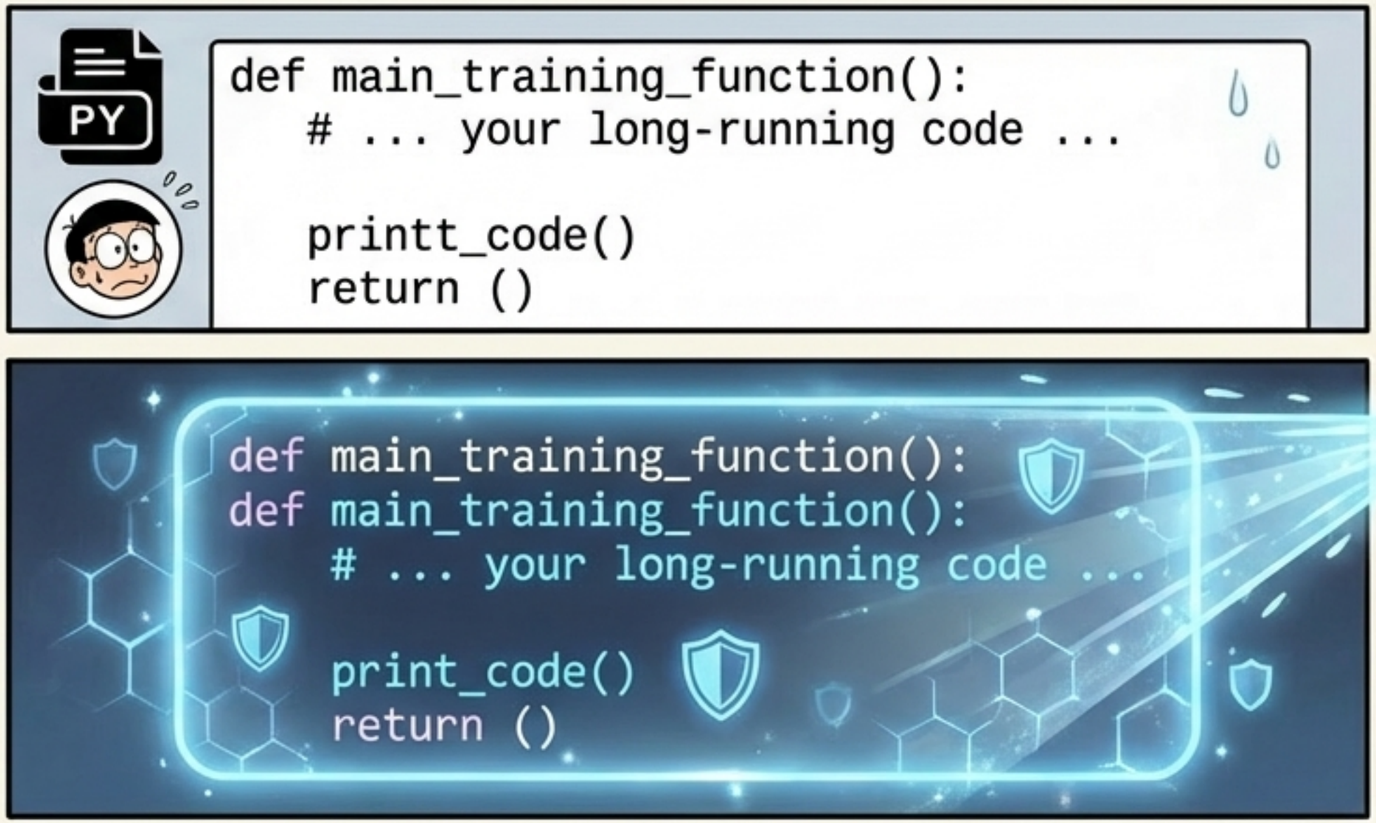
Paradigm 2: Checkpoint-Retry.

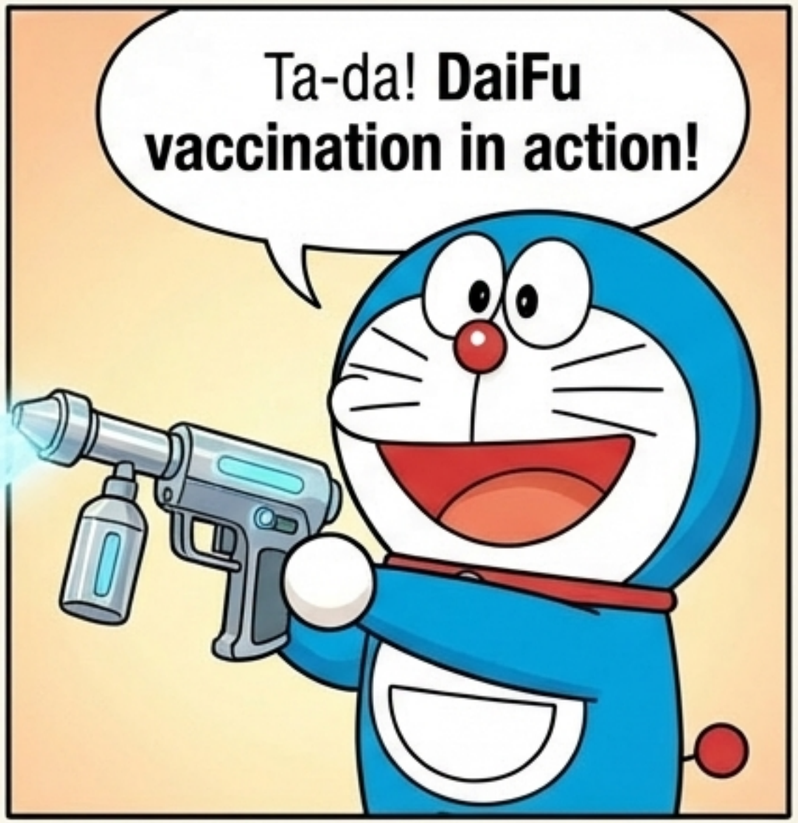


Retrying from the last "checkpoint" is better, but you still lose all the progress made since that save point. It's often a heavyweight solution for a minor error.


Nobita, you look like you've seen a 'CUDA out of memory' error! Don't worry, I've brought something from the future to help!
Ta-da! The **DaiFu In-Situ Recovery Vaccinator!**

What is “Program Vaccination”?



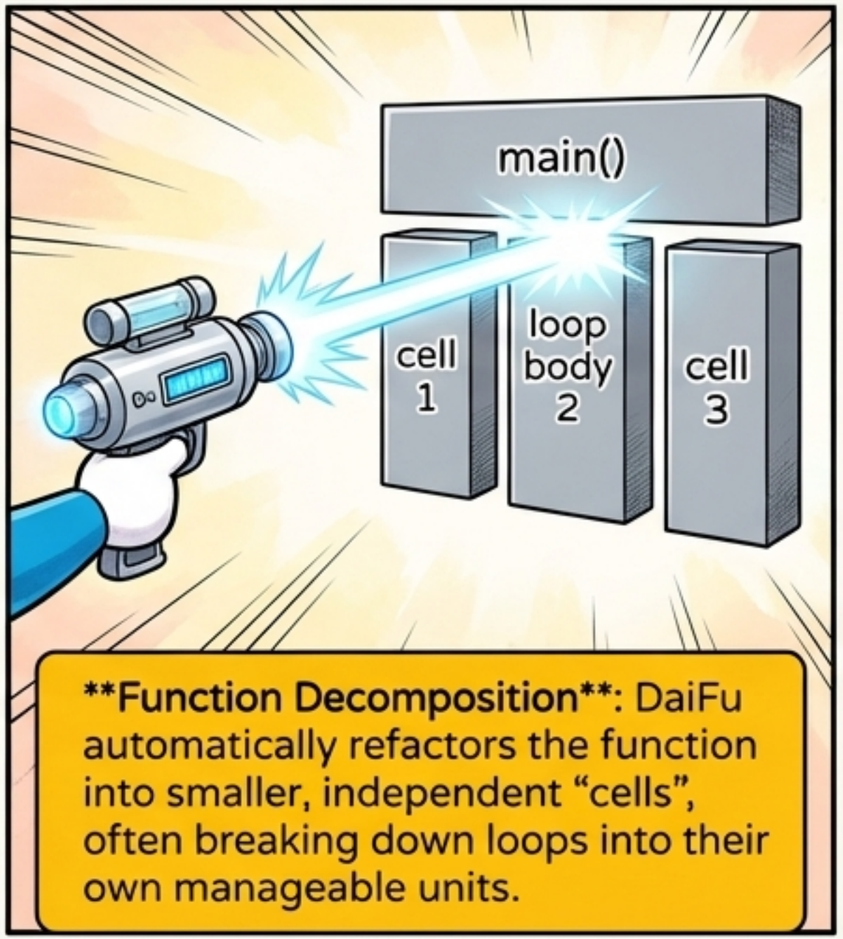


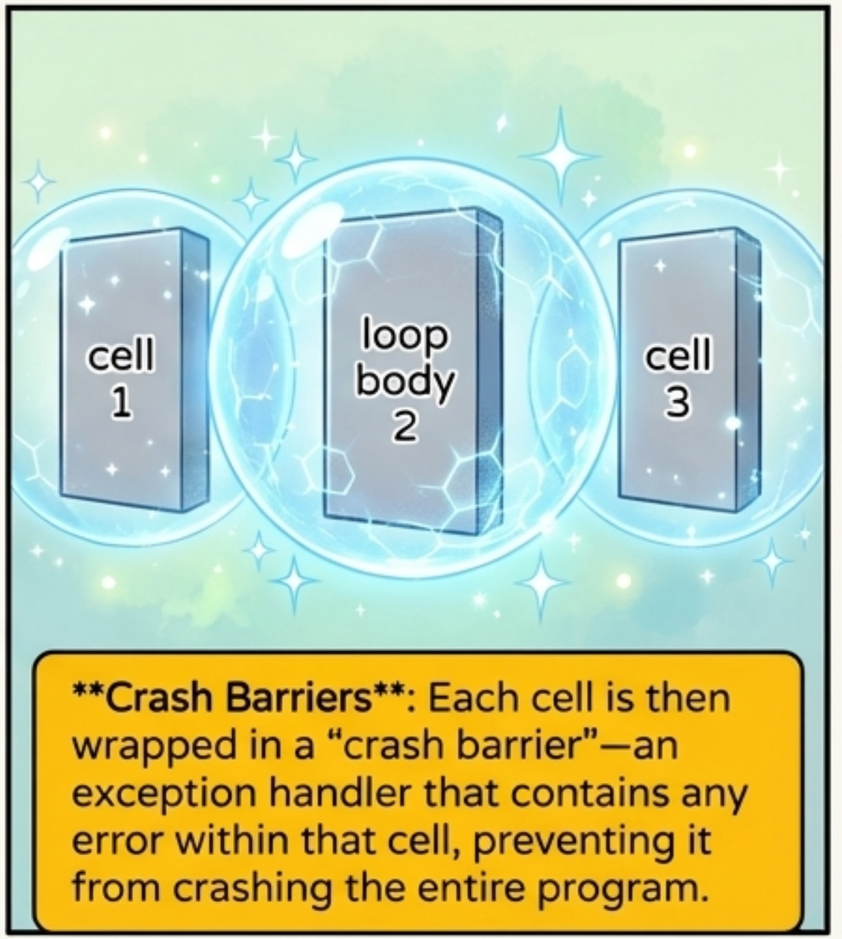
DaiFu “vaccinates” a Deep Learning system through a lightweight code transformation. With just two lines of code, it instruments the program to intercept crashes and enable instant, dynamic updates—without needing to restart.

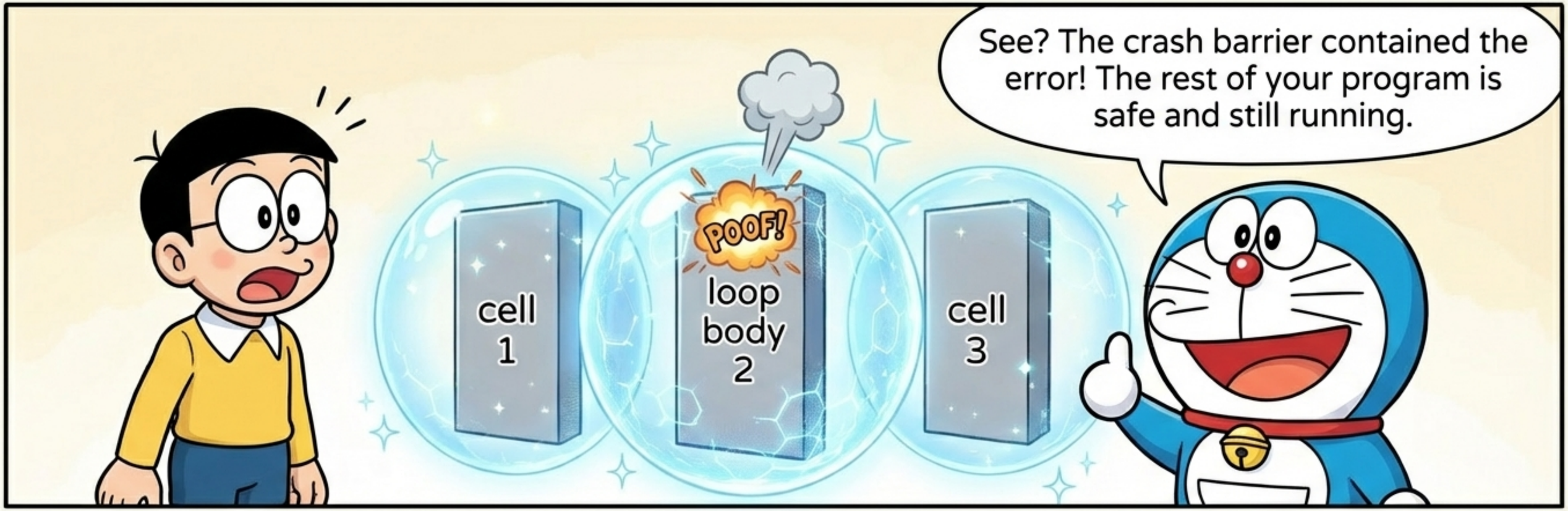
```
import daifu

@daifu.transform()
def main_training_function():
    # ... your long-running code ...
```

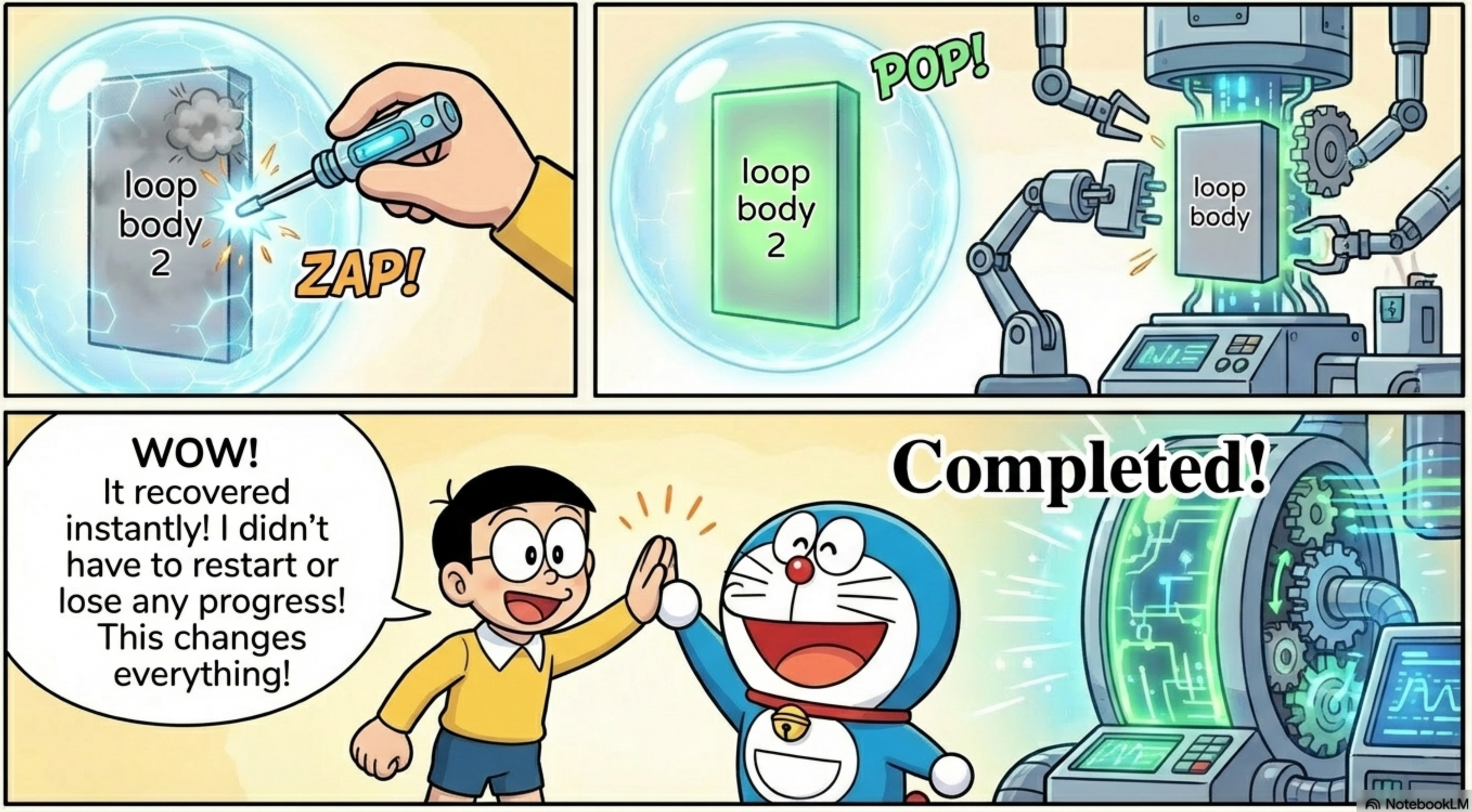
The Secret: Function Decomposition & Crash Barriers






See? The crash barrier contained the error! The rest of your program is safe and still running.
cell 1
POOF!
loop body 2
cell 3


Now for the magic of **In-Situ Recovery**! You can interact with the program *right now*, diagnose the problem, and apply a fix dynamically.
POOF!
loop body 2


loop body 2
ZAP!
POP!
loop body 2
loop body
WOW!
It recovered instantly! I didn't have to restart or lose any progress! This changes everything!
Completed!

This isn’t Magic, It’s 22nd Century Engineering!

DaiFu Vaccinator: Tech Specs

1327x
Restore Time Speedup
(Compared to state-of-the-art solutions)

< 0.40%
Runtime Overhead
(Negligible overhead during normal execution)

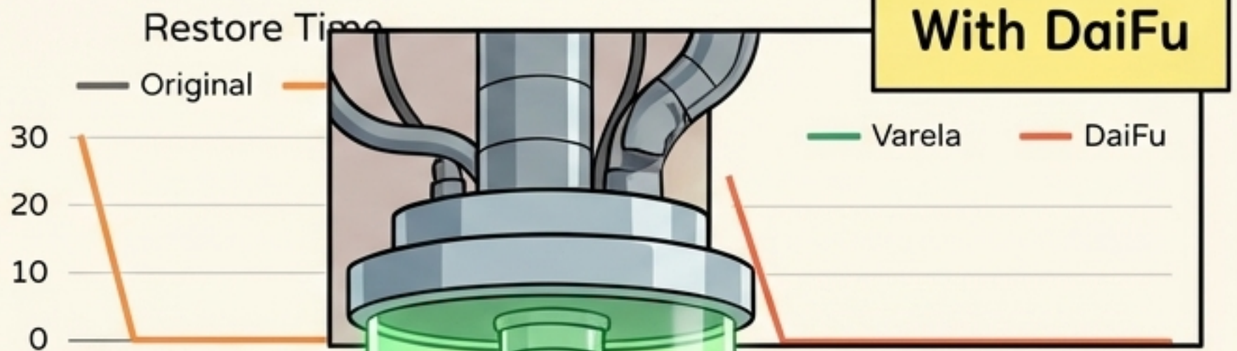
Seconds
Recovery Time
(Not minutes or hours)

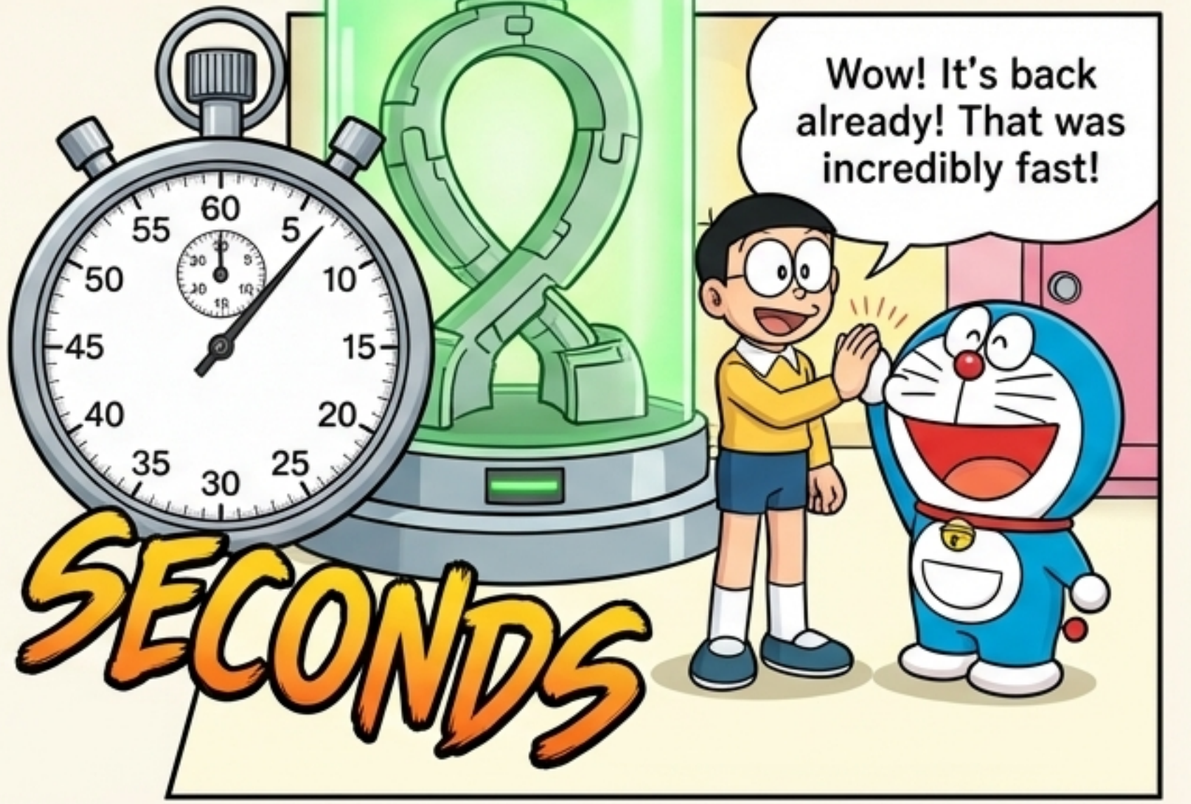
Minimal
Preparation Effort
(Automated transformation, no manual refactoring)

A Tale of Two Timelines: Crash Recovery Compared

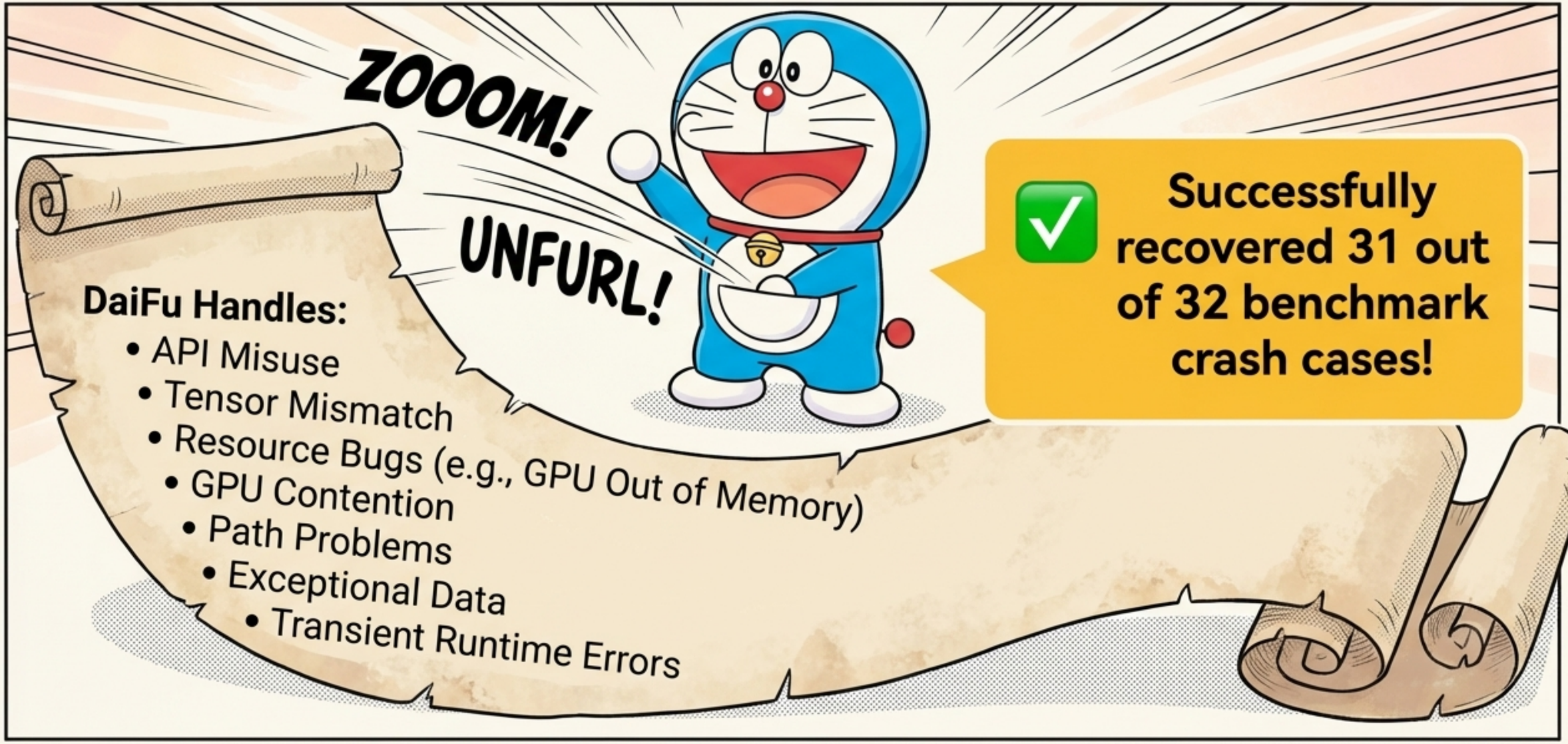
Without DaiFu

With DaiFu





Ready for Real-World Problems



Case Study: Fixing an “API Misuse” Crash In-Situ

The Bug

```
# The crash occurs here!
test_loss += criterion(output, target).data[0]
```

IndexError: invalid index of a 0-dim tensor.

The Fix

```
# Fixed in-situ with DaiFu's 'surgery' interface!
test_loss += criterion(output, target).data.item()
```

For common programming errors like misusing an API, DaiFu allows developers to diagnose the issue with the live program context and apply a code update instantly, resuming execution without losing state.

DaiFu: A New Paradigm for Crash Recovery

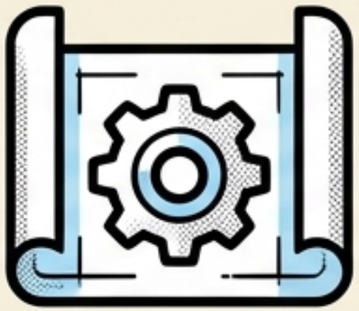
Feature	Restart / Checkpoint-Retry	DaiFu (In-Situ Recovery)
Retry Granularity	Program / Epoch / Iteration	Specific Lines of Code
Code Updates	Requires Full Program Restart	Dynamic, Instant Updates
Restore Time	Minutes to Hours	Seconds
Runtime Overhead	Can be high (e.g., 7.64%)	Negligible (<0.40%)
User Effort	Manual code refactoring	Automated (2-line decorator)

DaiFu is not just an optimization of checkpointing; it's a fundamentally different and more agile approach to handling failures in DL systems.

The Science Behind the 22nd Century Gadget

Perspective

Introduces and improves Dynamic Software Updating (DSU) to recover crashes in DL systems *in situ*—a new paradigm that achieves almost instant restoration.

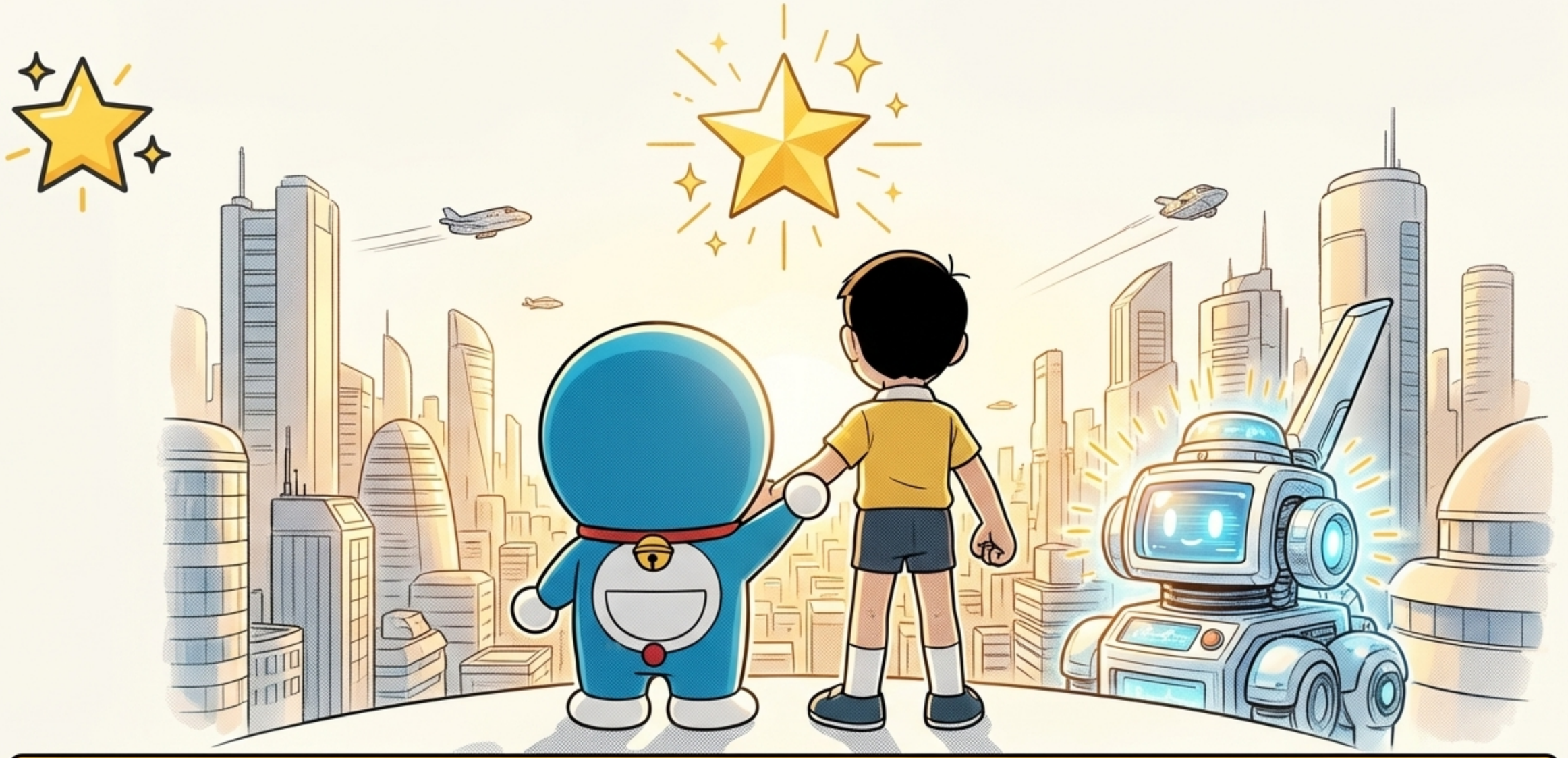


Framework

Proposes the novel DaiFu framework, which uses “Program Vaccination” to enable quick recovery while incurring low runtime overhead and minimal preparation effort.

Benchmark

Contributes a reproducible benchmark of 32 crash cases across 7 distinct scenarios, enabling comprehensive evaluation of DL crash recovery techniques.



By treating crashes not as catastrophic failures but as interceptable events, DaiFu turns hours of frustrating downtime into seconds of productive problem-solving. This accelerates development, saves vast computing resources, and empowers the next generation of AI innovation.