TEACHING SPELL CHECKERS TO TEACH

Pedagogical Program Synthesis for Interactive Learning

Momin N. Siddiqui[1], Vincent Cavez[2], Sahana Rangasrinivasan[3], Abbie Olszewski[4], Srirangaraj Setlur[3], Maneesh Agrawala[2], Hari Subramonyam[2]
[1]Georgia Institute of Technology [2]Stanford University [3]University at Buffalo [4]University of Nevada, Reno

IUI '26
Paphos, Cyprus
Mar 23-26, 2026

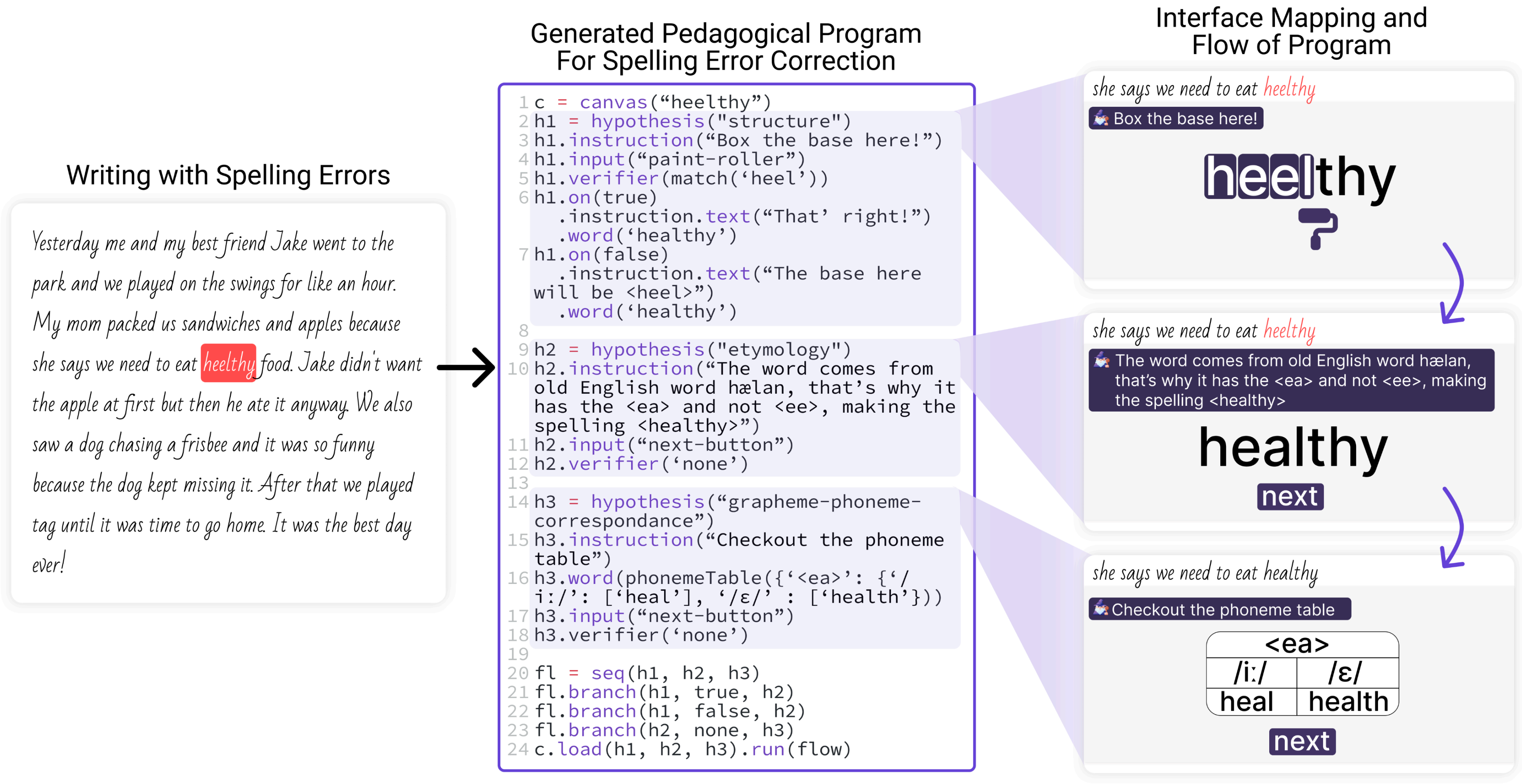
MOTIVATION & PROBLEM

Spelling taught through memorization often fails learners, particularly children with language-based learning disorders who struggle with phonological skills.

Speech-language pathologists (SLPs) use **inquiry-based** approaches targeting **phonology**, **morphology**, **meaning**, and **etymology** to teach spelling. Yet everyday writing tools simply detect and autocorrect errors, missing an opportunity for learning.

Research Question: How might we bring the rich, inquiry-based pedagogy of SLPs into everyday writing tools such that spelling errors become an opportunity to learn?

PEDAGOGICAL PROGRAM SYNTHESIS


Writing with Spelling Errors
Generated Pedagogical Program For Spelling Error Correction
Interface Mapping and Flow of Program

Yesterday me and my best friend Jake went to the park and we played on the swings for like an hour. My mom packed us sandwiches and apples because she says we need to eat heelthy food. Jake didn't want the apple at first but then he ate it anyway. We also saw a dog chasing a frisbee and it was so funny because the dog kept missing it. After that we played tag until it was time to go home. It was the best day ever!

```
c = canvas(“heelthy”)
h1 = hypothesis("structure")
h1.instruction(“Box the base here!”)
h1.input(“paint-roller”)
h1.verifier(match(‘heel’))
h1.on(true)
  .instruction.text(“That’ right!”)
  .word(‘healthy’)
h1.on(false)
  .instruction.text(“The base here
will be <heel>”)
  .word(‘healthy’)

h2 = hypothesis("etymology")
h2.instruction(“The word comes from
old English word hælan, that’s why it
has the <ea> and not <ee>, making the
spelling <healthy>”)
h2.input(“next-button”)
h2.verifier(‘none’)

h3 = hypothesis(“grapheme-phoneme-
correspondance”)
h3.instruction(“Checkout the phoneme
table”)
h3.word(phonemeTable({‘<ea>’: {‘/
i:/’: [‘heal’], ‘/ɛ/’ : [‘health’}))
h3.input(“next-button”)
h3.verifier(‘none’)

fl = seq(h1, h2, h3)
fl.branch(h1, true, h2)
fl.branch(h1, false, h2)
fl.branch(h2, none, h3)
c.load(h1, h2, h3).run(flow)
```

she says we need to eat heelthy

Box the base here!

heelthy

she says we need to eat heelthy

The word comes from old English word hælan, that’s why it has the <ea> and not <ee>, making the spelling <healthy>

healthy

next

she says we need to eat healthy

Checkout the phoneme table

<ea>	
/iː/	/ɛ/
heal	health

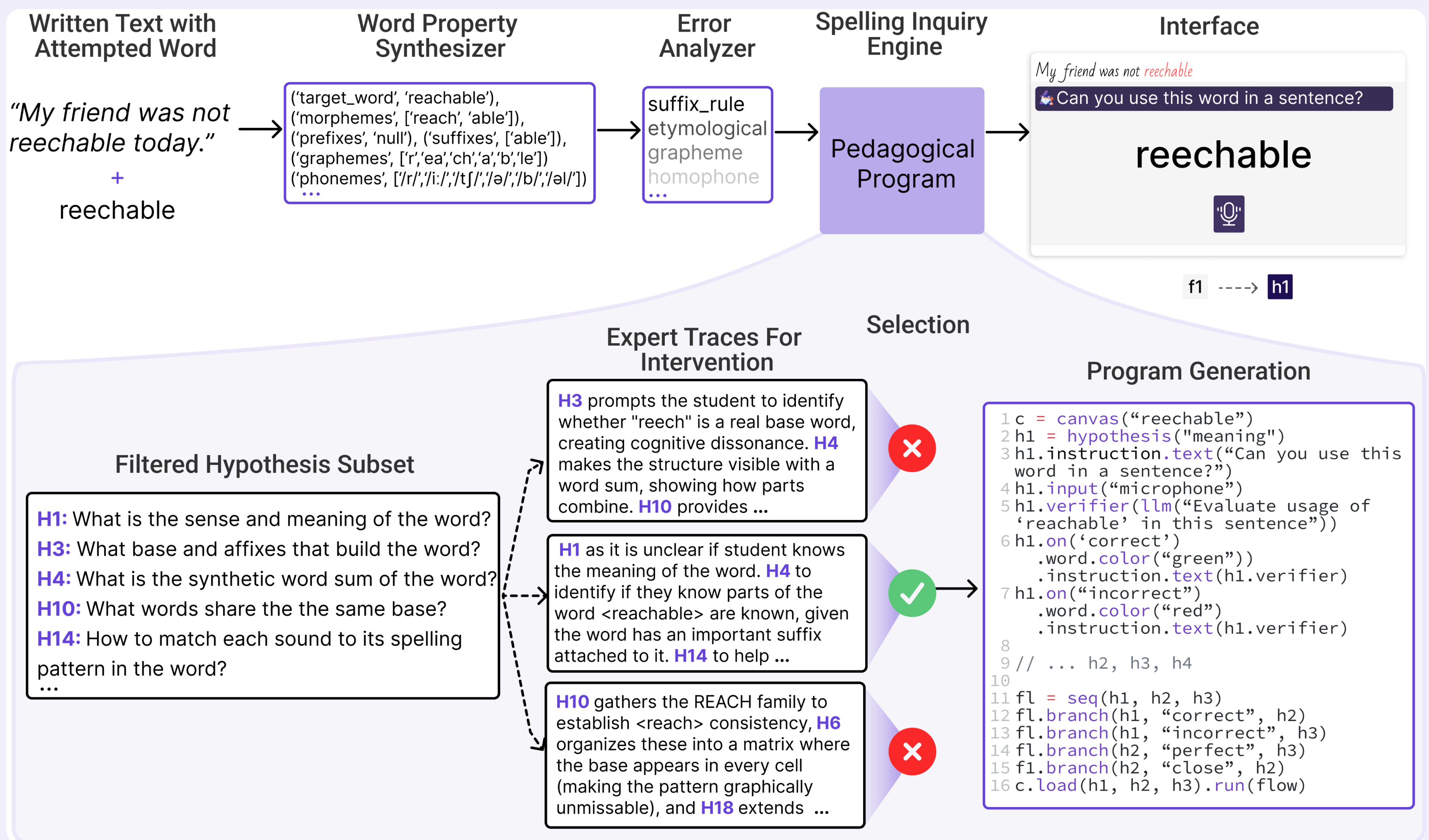
next

A novel approach that represents SLP instructional moves in a domain-specific language (DSL) and synthesizes tailored programs from learner errors in real-time.

The DSL formalizes Language Knowledge Primitives (morphemes, bases, graphemes, phonemes, etymology, related words), 18 Hypothesis Templates (each a structured quintuple: preconditions → evidential features → action base → warrant → learning effect), and Compositional Primitives (sequencing, nesting, branching, closure), enabling instruction as constrained search under uncertainty.

Example: "heelthy" → "healthy" When a learner misspells a word, SPIRE identifies the error in context, synthesizes a program of 3 to 5 inquiry steps, and renders it as an interactive interface. Here, the program asks the learner to box the base of the word, presents the etymology (Old English hælan) to explain the ⟨ea⟩ spelling, and displays a grapheme-phoneme table for guided exploration. The program branches adaptively on learner responses.

SYSTEM ARCHITECTURE


Written Text with Attempted Word
Word Property Synthesizer
Error Analyzer
Spelling Inquiry Engine
Interface
Pedagogical Program
Expert Traces For Intervention
Selection
Program Generation
Filtered Hypothesis Subset
f1 ---> h1

“My friend was not reechable today.”
+
reechable

(‘target_word’, ‘reachable’),
(‘morphemes’, [‘reach’, ‘able’]),
(‘prefixes’, ‘null’), (‘suffixes’, [‘able’]),
(‘graphemes’, [‘r’,‘ea’,‘ch’,‘a’,‘b’,‘le’])
(‘phonemes’, [‘/r/’,‘/iː/’,‘/tʃ/’,‘/ə/’,‘/b/’,‘/əl/’])
...

suffix_rule
etymological
grapheme
homophone
...

My friend was not reechable

Can you use this word in a sentence?

reechable

H1: What is the sense and meaning of the word?
H3: What base and affixes that build the word?
H4: What is the synthetic word sum of the word?
H10: What words share the the same base?
H14: How to match each sound to its spelling pattern in the word?
...

H3 prompts the student to identify whether "reech" is a real base word, creating cognitive dissonance. **H4** makes the structure visible with a word sum, showing how parts combine. **H10** provides ... ✗

H1 as it is unclear if student knows the meaning of the word. **H4** to identify if they know parts of the word <reachable> are known, given the word has an important suffix attached to it. **H14** to help ... ✓

H10 gathers the REACH family to establish <reach> consistency, **H6** organizes these into a matrix where the base appears in every cell (making the pattern graphically unmissable), and **H18** extends ... ✗

```
c = canvas(“reechable”)
h1 = hypothesis("meaning")
h1.instruction.text(“Can you use this
word in a sentence?”)
h1.input(“microphone”)
h1.verifier(llm(“Evaluate usage of
‘reachable’ in this sentence”))
h1.on(‘correct’)
  .word.color(“green”))
  .instruction.text(h1.verifier)
h1.on(“incorrect”)
  .word.color(“red”)
  .instruction.text(h1.verifier)

// ... h2, h3, h4

fl = seq(h1, h2, h3)
fl.branch(h1, “correct”, h2)
fl.branch(h1, “incorrect”, h3)
fl.branch(h2, “perfect”, h3)
fl.branch(h2, “close”, h2)
c.load(h1, h2, h3).run(flow)
```

SPIRE processes each misspelling through five chained modules.

Given "reechable" in "My friend was not reechable today":
The **Spell Checker** uses LLM-based contextual analysis to detect the error and identify "reachable" as the intended word.
The **Word Property Synthesizer** extracts its linguistic profile: morphemes [reach, -able], graphemes, phonemes, and etymology.
The **Error Analyzer** compares the attempt against this profile, diagnosing likely causes across linguistic dimensions.
The **Spelling Inquiry Engine** filters relevant hypothesis templates from the DSL, generates candidate instructional traces through three parallel LLM instances, ranks them for pedagogical soundness, and synthesizes the top trace into an executable program with branching logic.
The **Interface** Generator renders this program as interactive inline components within the writing environment.

EVALUATION

◆ **Expert Evaluation:** 10 SLP experts rated 50 synthesized conversations. Pedagogical reasoning: μ = 4.90/5 (74.7% perfect agreement). Instructional actions: μ = 4.32/5 (82% rated ≥4), with moderate variance reflecting genuine pedagogical flexibility, as effective tutoring admits multiple valid approaches.
◆ **User Study:** 7 children (ages 7–11) used SPIRE in 45-minute sessions. Ease of use and confidence: median 5/5. None had prior exposure to morphological analysis, yet all spontaneously discussed learning words beyond fixing errors.

FUTURE WORK

◆ **Cost & scalability:** Replacing commercial API calls (~$0.70/word) with local fine-tuned models to reduce latency and cost for school deployment.
◆ **Diverse populations:** Expanding evaluation to non-native speakers (Chinese, Russian, Spanish) and broader grade/proficiency ranges.
◆ **Longitudinal study:** Tracking spelling development, transfer of morphological reasoning, and sustained engagement across authentic writing tasks.

Momin N. Siddiqui, Vincent Cavez, Sahana Rangasrinivasan, Abbie Olszewski, Srirangaraj Setlur, Maneesh Agrawala, and Hari Subramonyam. 2026. Teaching Spell Checkers to Teach: Pedagogical Program Synthesis for Interactive Learning. In 31st International Conference on Intelligent User Interfaces (IUI '26), March 23–26, 2026, Paphos, Cyprus. ACM, New York, NY, USA, 18 pages.
Current address: https://doi.org/10.48550/arXiv.2512.12115
Supported by the National Science Foundation and the Institute of Education Sciences, U.S. Department of Education through Award #2229873 (National AI Institute for Exceptional Education).