WikiDo

Nate Kushman

M. Brodsky, S. Branavan, D. Katabi, R. Barzilay,
M. Rinard, N. Van Dyke, and S. Bauer

Computer Usability Is A Mess

- Average user is constantly running into tasks they don't know how to do:
  - Configure Outlook with their ISP
  - Configure Outlook with Gmail
  - Setting Facebook privacy settings to fully private
  - Configure Remote Desktop on home computer
  - Turn on wireless encryption on their home router

They Have the Web

Crowd-sourced solutions→ average task is covered

Provides only text → hard to use

What they want: Automation

Does not require any expertise, they “just run it”

→ easy to use

Can only be produced by expert programmers

→ Will never scale to the wide diversity of tasks

The Best of Both Worlds

solution easy to use

average task covered

Runs automatically

Generated by the masses

How do we enable the masses to collaborate on automating computer tasks

Our Approach

Use GUI actions as the primitive
instead of text or programming languages

Automate By Doing

WikiDo: Crowd Sourced Database of Automated Tasks

To contribute to the database:

User performs the task

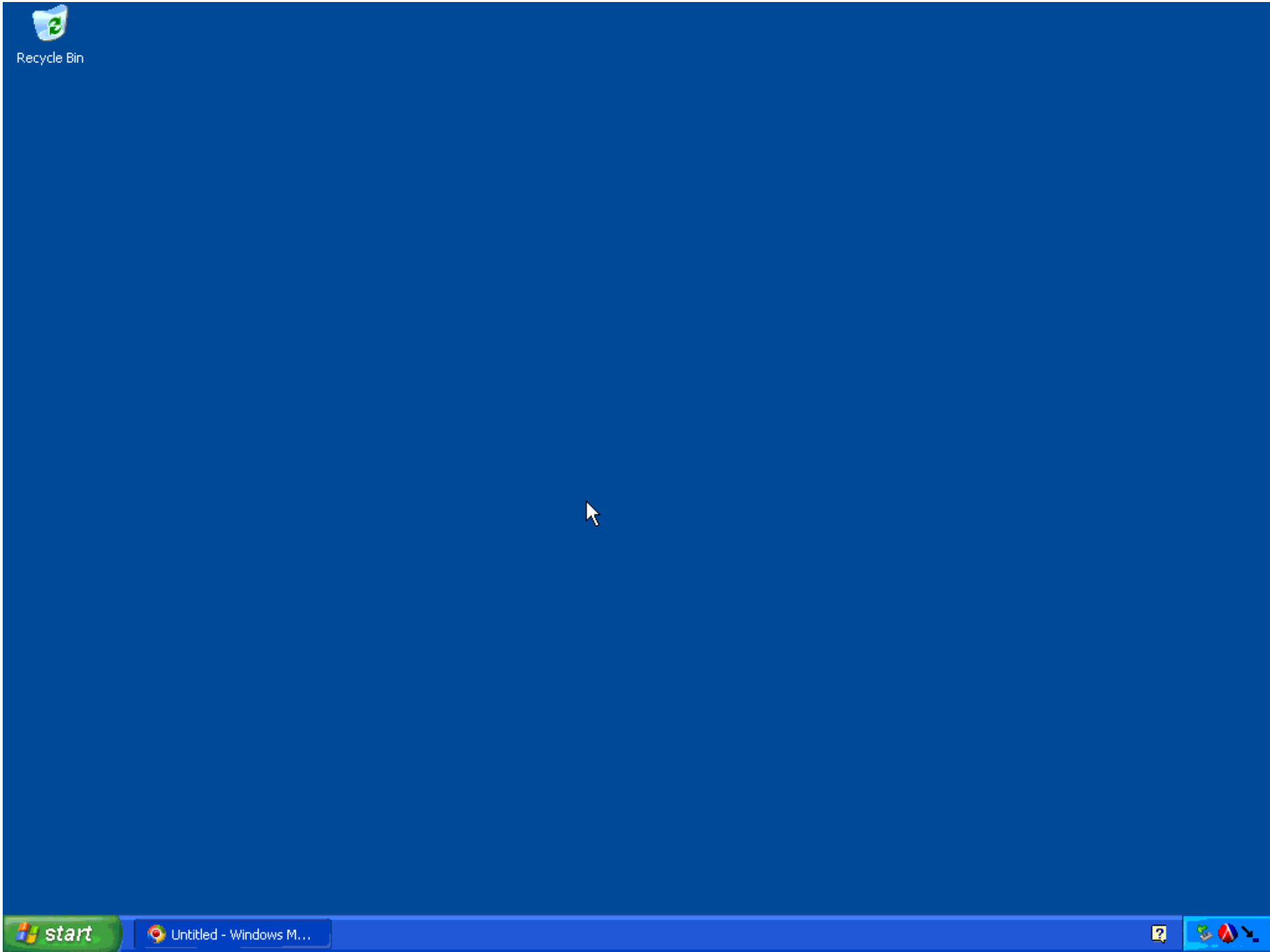
- either on their own machine or in a VM
- WikiDo records a trace of GUI actions

WikiDo merges multiple traces to create a canonical solution

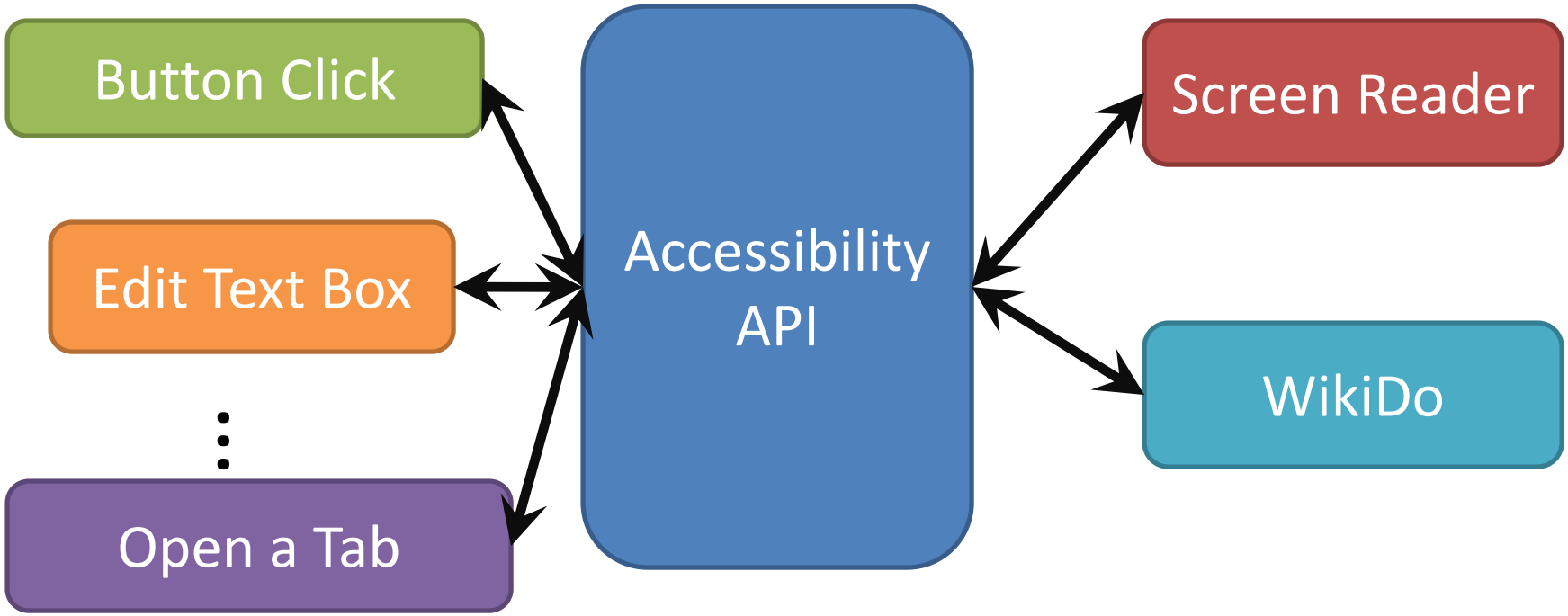
To using the database:

WikiDo replays the canonical solution

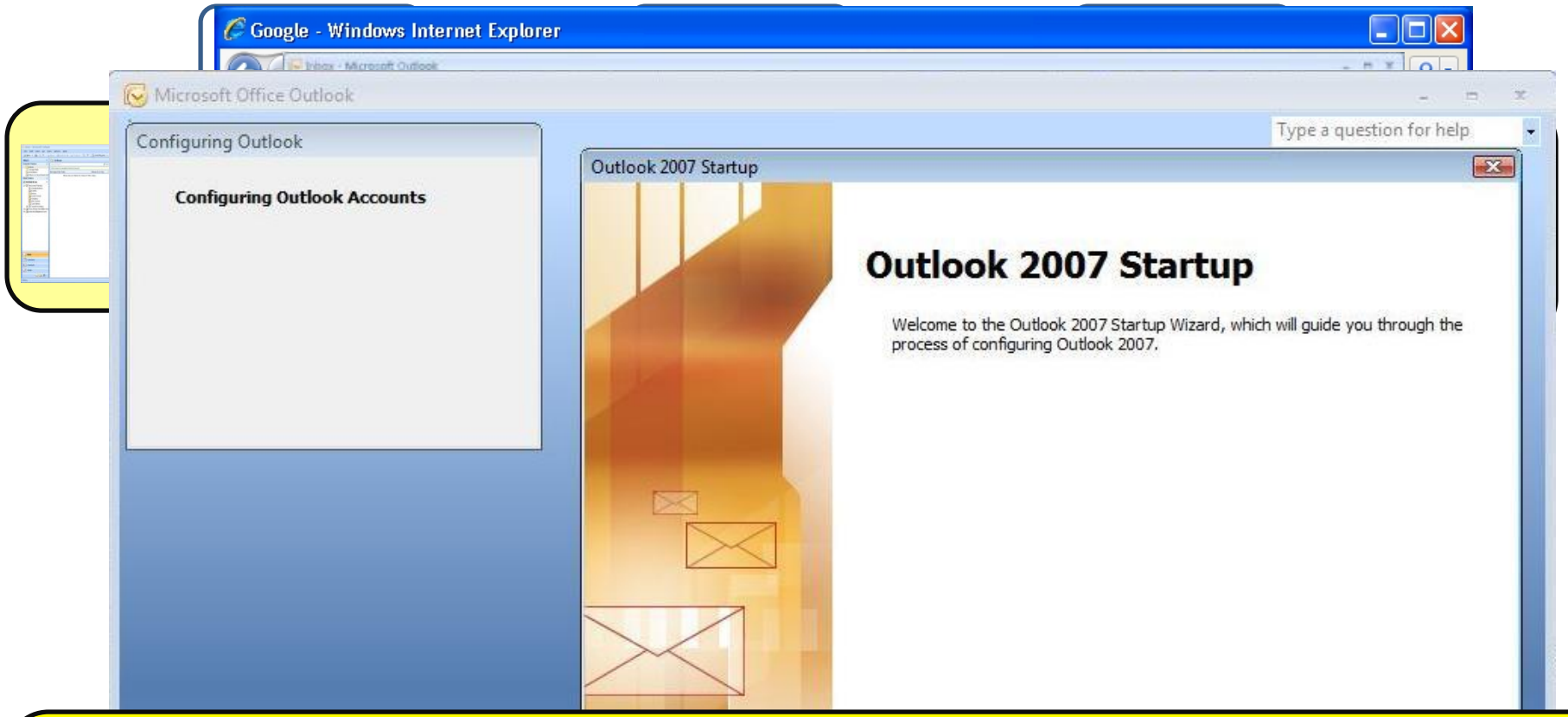
- Can also walk through step-by-step


Recycle Bin
start
Untitled - Windows M...

How Does WikiDo Record Traces?



Generating a Canonical Trace/Handling Differences Between Traces



Challenge: How do we differentiate between spurious actions and environment specific differences?

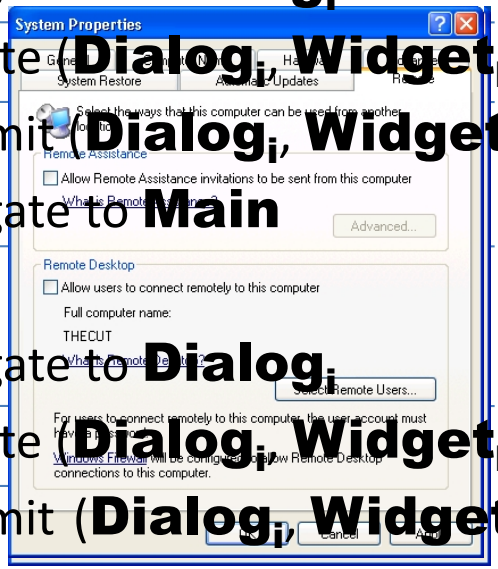
Solution Idea: Track system state and identity spurious actions as those that don't affect final state

Actions on GUI widgets can be modeled as:

- **Update:** pending change to system state
  - e.g., check box, editing a text box
- **Commit:** write pending updates to system state
  - e.g., OK/Cancel button
- **Navigate:** no change to system state
  - e.g., opening a dialog box

Transforming to Abstract Representation

Raw GUI Actions	Abstract Actions
Click **Open Dialog**	Navigate to **$Dialog_i$**
Check **Check Box**	Update (**$Dialog_i$**, **$Widget_k$**)
Click **OK**	Commit (**$Dialog_i$**, **$Widget_k$**) Navigate to **Main**
Click **Open Dialog**	Navigate to **$Dialog_i$**
UnCheck **Check Box**	Update (**$Dialog_i$**, **$Widget_k$**)
Click **OK**	Commit (**$Dialog_i$**, **$Widget_k$**) Navigate to **Main**



3 Pass Removal of Spurious Actions

Pass 1: Removing Unnecessary Updates

Start at the end → See first the final update of each widget

Go backwards → Eliminating all non-final updates

Navigate to **Dialog$_i$**

~~Update (**Dialog$_i$**, **Widget$_k$**)~~

Commit (**Dialog$_i$**, **Widget$_k$**)

Navigate to **Main**

Navigate to **Dialog$_i$**

Update (**Dialog$_i$**, **Widget$_k$**)

Commit (**Dialog$_i$**, **Widget$_k$**)

Navigate to **Main**

3 Pass Removal of Spurious Actions

Pass 2: Removing Unnecessary Commits

Walk Forwards eliminating commits with no pending updates

Navigate to **Dialog$_i$**

~~Commit (**Dialog$_i$**, **Widget$_k$**)~~

Navigate to **Main**

Navigate to **Dialog$_i$**

Update (**Dialog$_i$**, **Widget$_k$**)

Commit (**Dialog$_i$**, **Widget$_k$**)

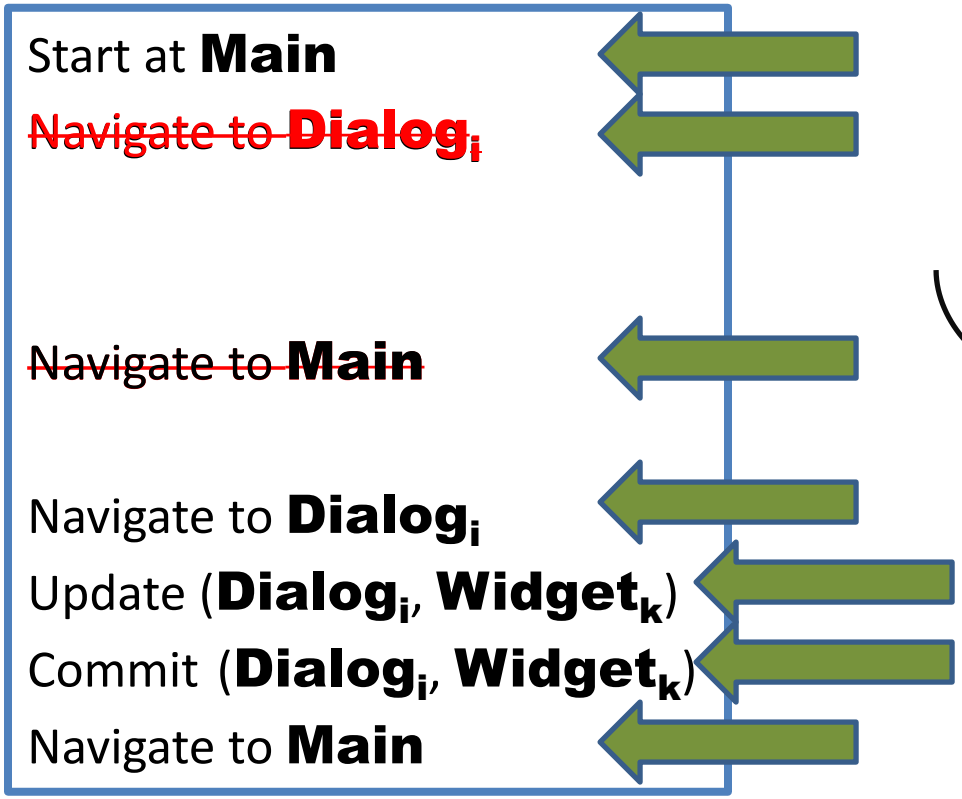
Navigate to **Main**

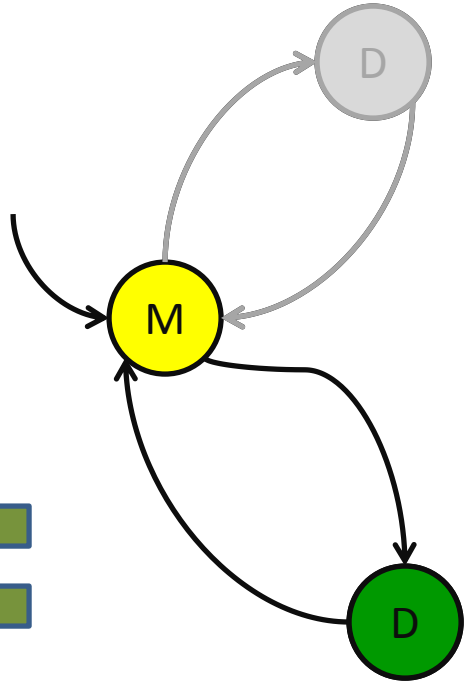
3 Pass Removal of Spurious Actions

Pass 3: Removing Unnecessary Navigation

Walk forwards → Build a navigation graph

Remove any loops which contain no commits or updates





3 Pass Mistake Removal Algorithm

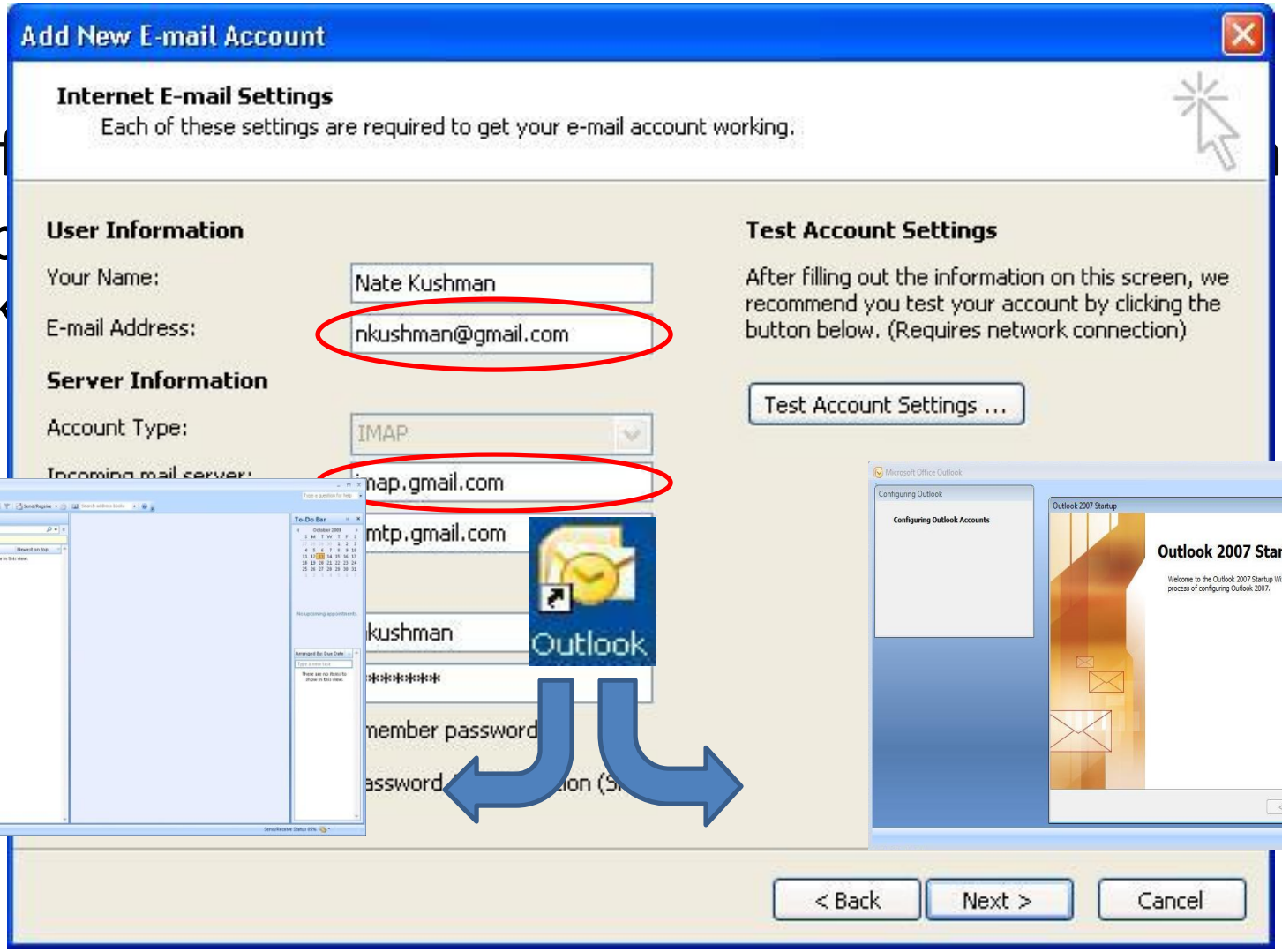
Start at **Main**

Start at **Main**
Navigate to **Dialog$_i$**
Update (**Dialog$_i$**, **Widget$_j$**)
Commit (**Dialog$_i$**, **Widget$_j$**)
Navigate to **Main**

Handling User Specific Environments

Step

- Dif
- Ob
  tak
  in



Scaling Beyond Even Crowd Sourcing

- Leverage wealth of how-to on-line documents
- Ideally we'd like to use fully automated machine translation
  - "Press OK" →LEFT_CLICK on BUTTON:OK
- State of the art English to GUI translators are correct only 37% of the time [Branavan09]

Combine Machine Learning With Crowd Sourcing

Weak Translator

?

63%

37% Correct Translations

Challenges:

-Don’t know which translations are correct

-Can’t ask humans to translate all remaining 63%

Challenge 1: Don't know which translations are correct

Solution Idea:

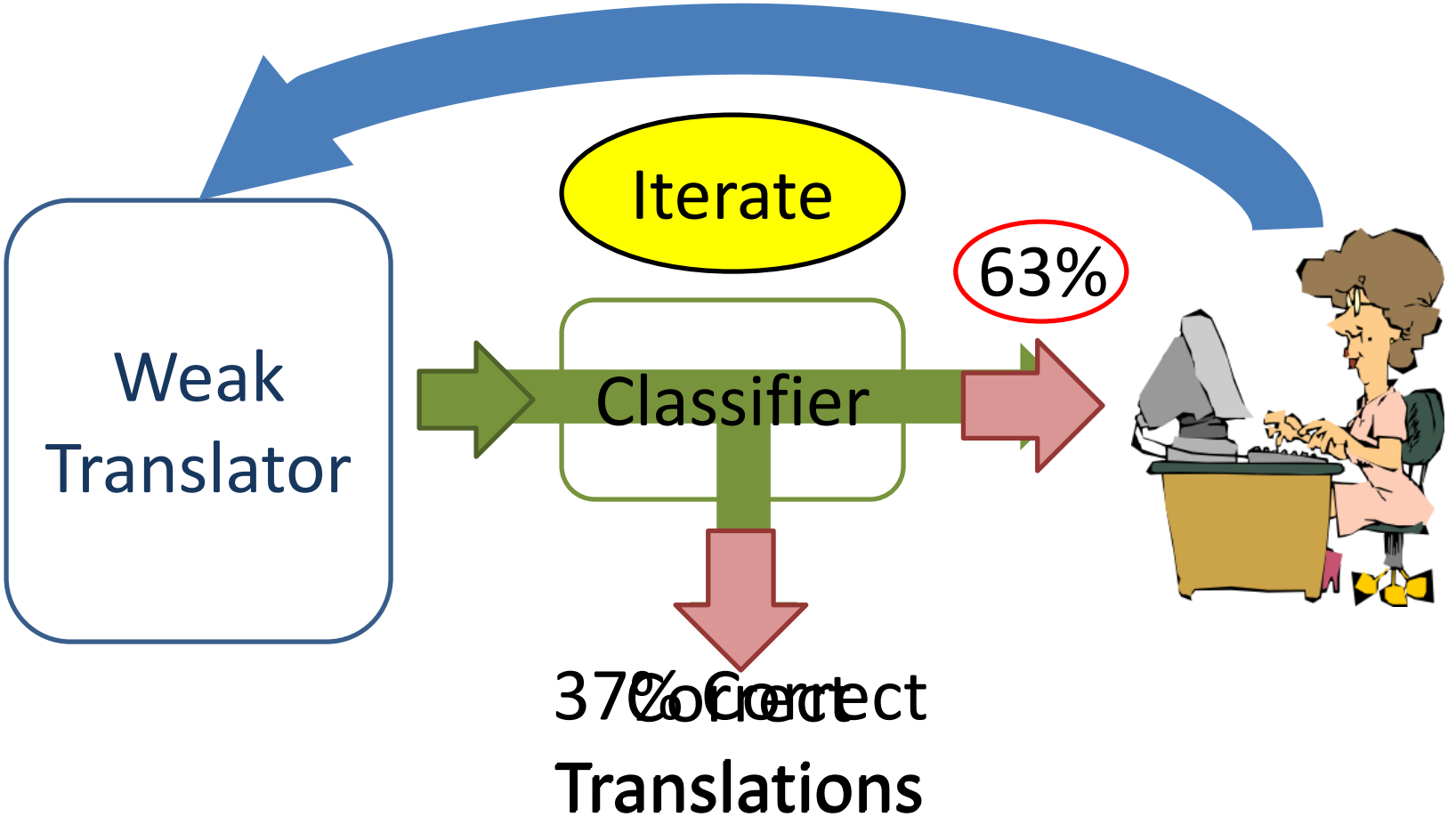
Hard Sentences Easier to Detect Than Translate

- Many features help detect hard sentences but don't help translate:
  - Unfamiliar phrases
  - Multiple translations have equal likelihood
  - etc.
- Combine these features using an ML Classifier
  - Set of features → correct or incorrect
  - Currently use a Support Vector Machine (SVM)

Challenge 2: Can't ask humans for all remaining 63%

Solution Idea:

Also use human translations to retrain ML Trans



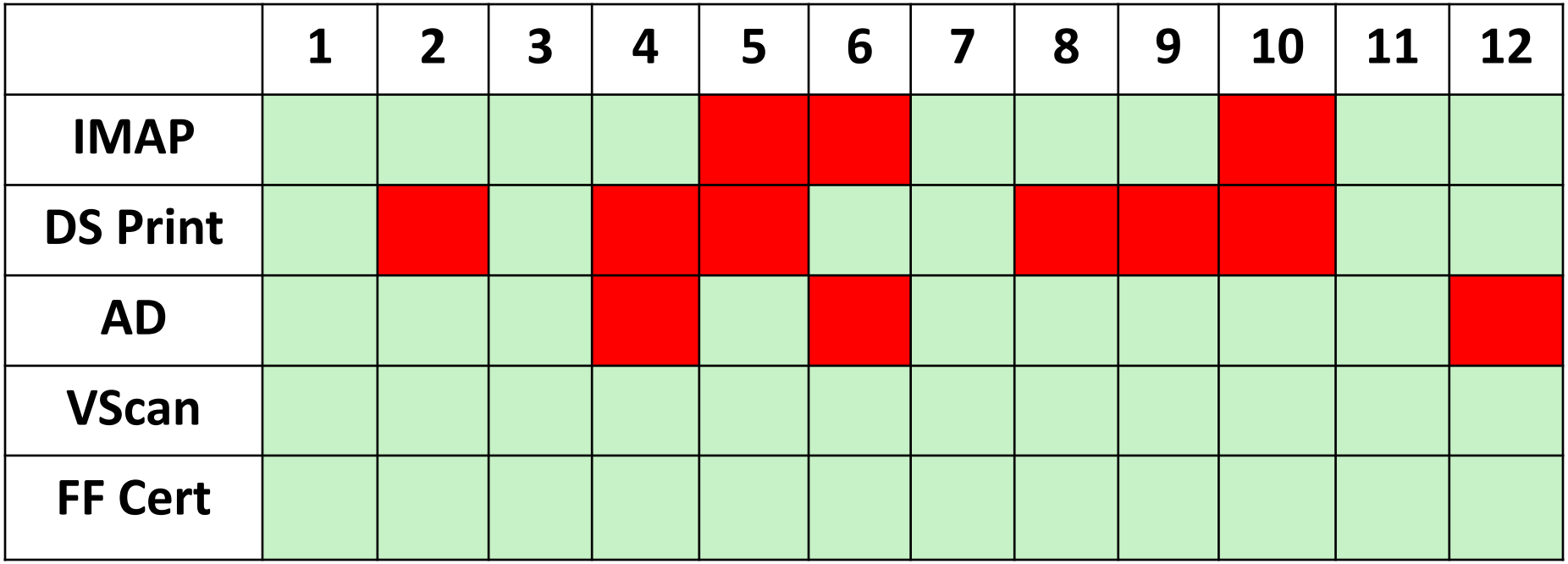
Results

Merging: Experiment Setup

- Asked 12 CS students to each perform 5 tasks
- Recorded tasks using a prototype WikiDo recorder
- Merged together the 12 recordings to create a single canonical recording

Successful Task Completion

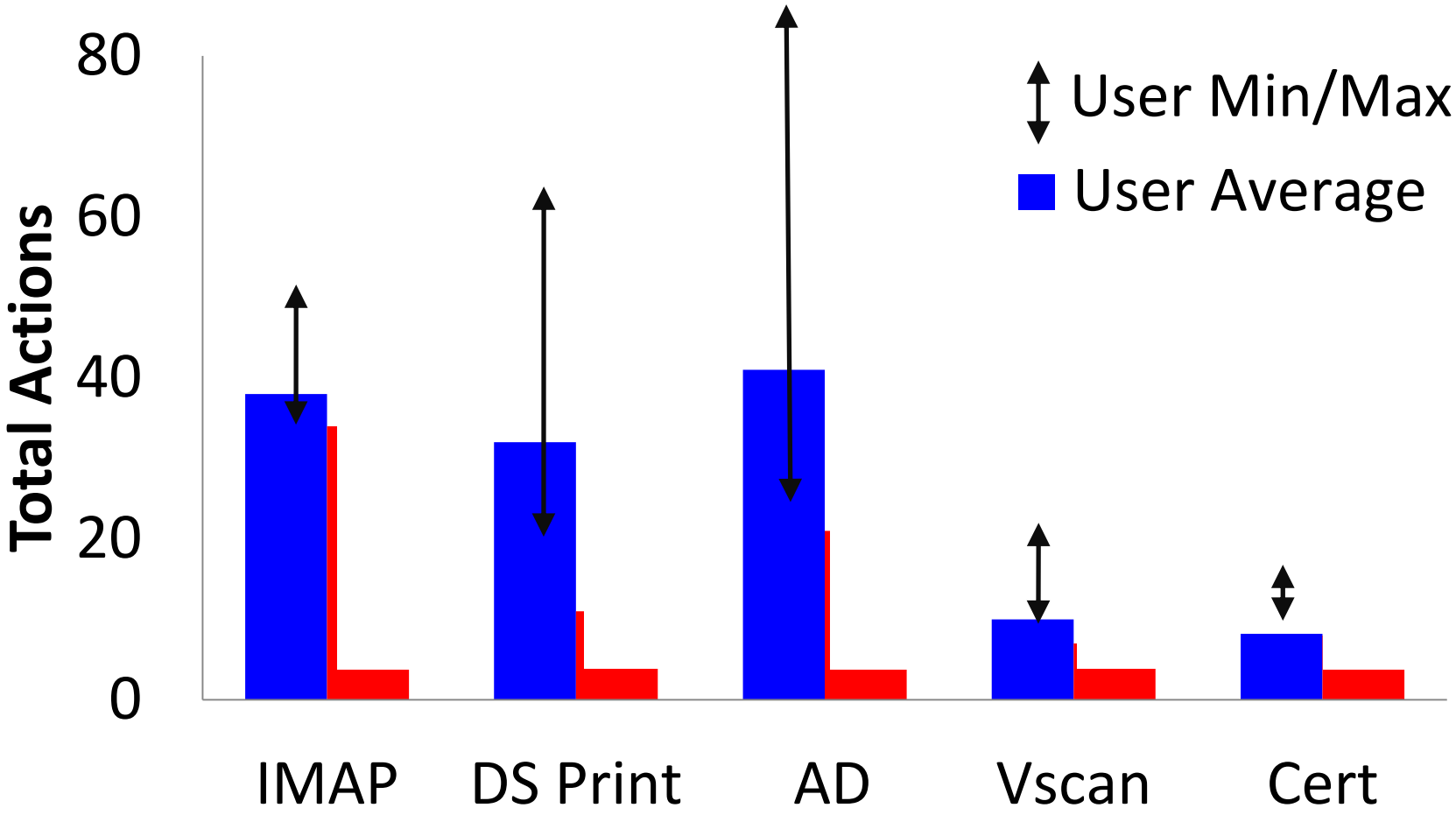
Users



	1	2	3	4	5	6	7	8	9	10	11	12
IMAP												
DS Print												
AD												
VScan												
FF Cert												

Users fail to successfully complete 20% of tasks

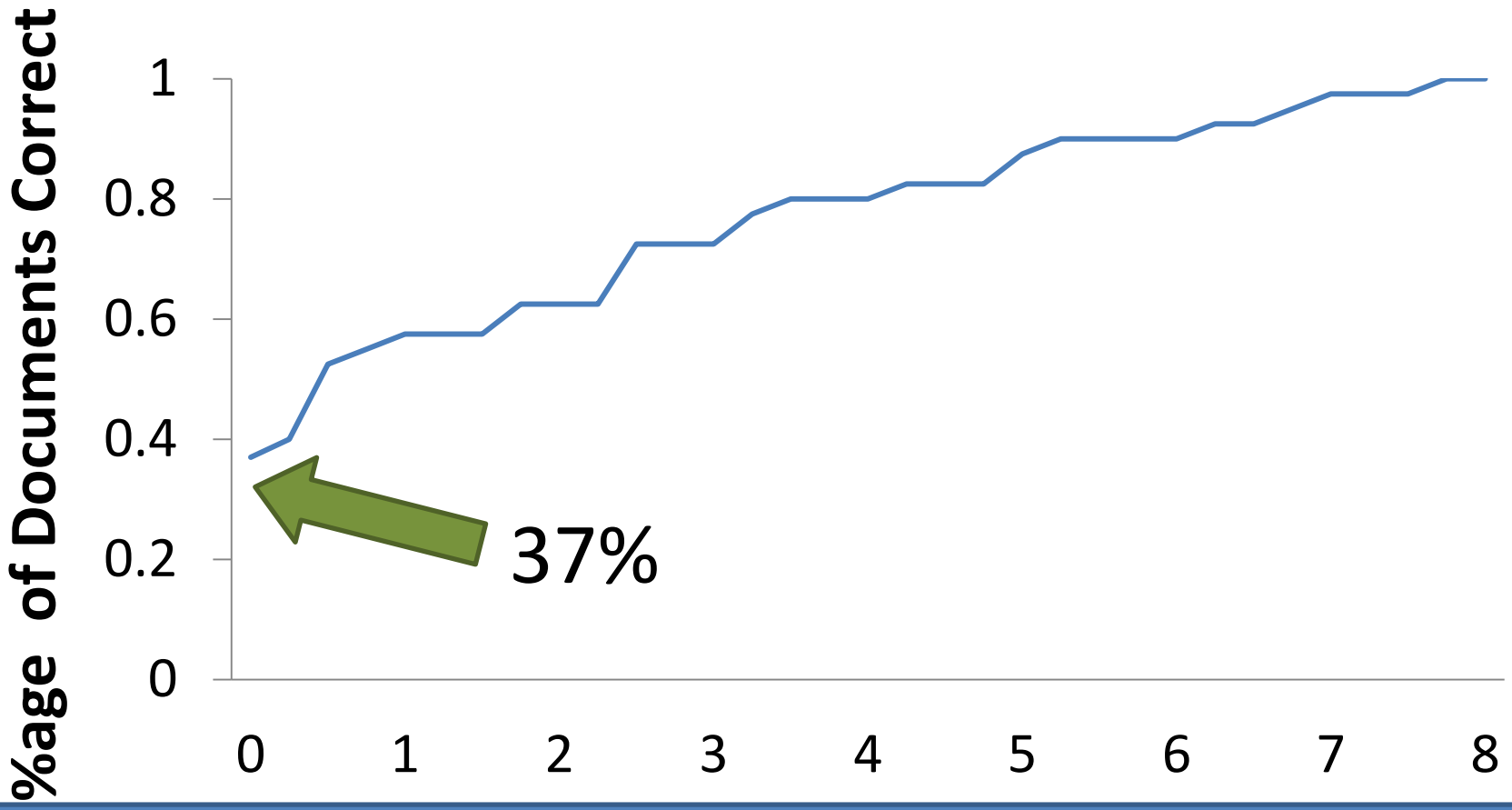
Merging: Results



Translation: Experiment Setup

- Built a prototype version of iterative translator using an ML translator built by our co-authors
- 120 articles from the Microsoft KB

If an Oracle identifies hard sentences, how much human help needed → 100% correct?



100% correct after humans perform only 7-8% of actions

Classifier Accuracy

- 94% of steps classified correct are actually correct
- 88% of steps classified wrong are actually wrong

WikiDo

- A crowd sourced databased of automated tasks
  - Contribute by doing
  - Use by playing it back
- Merge together multiple examples to create a single canonical solution
- Takes advantage of existing text by combining machine translation with crowd sourcing

Contribute to WikiDo: http://wikido.csail.mit.edu