Research methods for HCI

Interactive Machine Learning

MAS.S62

Outline

1. Needfinding
2. Rapid prototyping
3. Mental models
4. Experimentation

*Much of this talk is based on lectures from Scott Klemmer's HCI course on Coursera.

Needfinding

Forms of evaluation designed to ascertain the needs of potential end-users.

- usually applied early in the design process

Needfinding: Choosing subjects

Usually choose subjects representative of target population (maybe current users),

but non-users could be helpful to understand what obstacles are preventing use by certain groups.

Needfinding: Choosing subjects

extreme users - use something extremely much or extremely little or in some other extreme way

lead users - those who innovate to improve their own user interfaces

- Lead users experience the interaction *and* reflect on their own needs.
- Innovations can directly map to design improvements.

Needfinding: Choosing subjects

personas - abstract users modeled after a category of observed people

A persona is specified to include demographic info, motivation, beliefs, behavior, goals, reasons for using the technology.

Give them a name, a story, a picture, etc. to ground them and elicit empathy for the user.

- Empathy can lead to insights.

Needfinding: User Observation

Pay attention to all artifacts

- e.g., post-it notes giving instructions that aren't apparent or avoid common errors

Look for workarounds and hacks.

"Errors" tell you crucial information about where you might redesign to make the system more intuitive to the user.

Self-report can lead you astray. Weigh what people do more than what they say.

Needfinding: Interviewing, Bad questions

What would you do / like / want in a hypothetical scenario?

- too hard to imagine and respond realistically

How often do you do X?

- unreliable; be more concrete (# times in a specific time window)

How much do you like X (on an absolute scale)?

- attach words to Likert scales

Needfinding: Interviewing, Good questions

Open-ended questions

Grounded questions

- e.g., “I see that you do use function X much more than function Y. Are there reasons for this difference that you’re aware of?”

Allow silence after questions and short responses

- often great information comes after giving silence

Needfinding: Other methods of self report

diaries - for long-term processes or unpredictably random events

experience sampling - ping/beep people at times to get them to give experience information

Rapid prototyping

Fail early, fail fast, fail often

Rapid prototyping

Fail early, fail fast, fail often

Get your design in the hands of potential end-users and stakeholders as quickly as possible,

involving users throughout prototyping.

Rapid prototyping

Low fidelity – users less reserved in feedback

High fidelity – more believable system creates more natural interaction

Rapid prototyping: storyboarding

the first step in design (after need-finding)

not art, communication (e.g., use star people)

convey:

- setting: people, environment, and task being accomplished
- sequence: steps involved, how someone starts using the system, and what task is supported by your system
- user satisfaction: what motivates usage, what is accomplished, what need is filled

at end, show satisfactory outcome

Rapid prototyping: paper and digital prototypes

Paper prototyping

- sketch interface on paper
- act out the interface for the user
- can change interface on the fly, letting users help

Digital prototypes

- take more time to create but higher fidelity

Rapid prototyping: Wizard of Oz (aka Faking it)

simulating system behavior with human puppeteers

user believes they are interacting with the real system

Rapid prototyping: Wizard of Oz (aka Faking it)

Advantages:

- good when it's faster/cheaper/easier than the real thing
- more "real" than paper prototyping
- identifies bugs and problems with current design (fail sooner!)
- can envision challenging-to-build applications (like robots…)
- designers learn by playing wizard
- scales with system's increasing functionality through shared autonomy

Rapid prototyping: Wizard of Oz (aka Faking it)

Disadvantages:

- might misrepresent otherwise imperfect tech
  - e.g., speech recognition
- may simulate technologies that don't and will never exist
- wizards require training and can be inconsistent
- playing the wizard can be exhausting
  - all interaction hours are human experimenter hours (i.e., not automated)
- some features and limitations are difficult/impossible to simulate

Rapid prototyping: Wizard of Oz (aka Faking it)

Design:

- map out scenarios and application flow (what should happen in response to user behavior)
- create interface "skeletons" and "hooks" for wizard input (connection between interface and wizard's interface)
- decide where and how wizard will provide input
- keep in mind that typically you've got to eventually replace human with computer
- rehearse wizard role with colleague

Rapid prototyping: Wizard of Oz (aka Faking it)

Running:

- practice with colleague first
- recruit subjects when comfortable
- two people needed: study facilitator and wizard
- might ask users to give feedback by
  - thinking aloud during task,
  - retrospectively evaluating (maybe with video), or
  - heuristic evaluation (beforehand ask them to pay attention to certain aspects that you'll want feedback on)
- debrief users (with full honesty)

Mental Models

mental model - effects that user expects from actions

Mental Models

mental model - effects that user expects from actions

- develops through interaction
- mental model is often very different than that of designers!
- mismatch btwn designer's and user's mental models leads to slow performance, errors, frustration, …

sources of mental models: experience, metaphor, and analogy to familiar interfaces

Mental Models

categories of errors

- slips - correct mental model but error from motor error, lazy thinking, etc.
- mistakes - come from incorrect mental models

world-in-miniature strategy - make the control interface look like a smaller version of the thing being controlled

Designing studies

Designing studies

base rates - how often does y occur

correlations - do x and y co-vary?

causes - does x cause y

- requires manipulation of x and way of controlling for confounding variables (e.g. randomization)

independent variables and dependent variables

Validity

Internal validity - would repeating the study get the same results?

External/ecological validity - generalizability from test population/conditions to population/ conditions of interest

- tradeoff with how controlled experiment is

Comparing approaches

Factors: approach (core idea), fidelity of implementation, user expertise

Strategies for fairer comparisons

- put both approaches in the same setting
  - possibly in the production setting (e.g. on the actual web)
- when expertise is relevant, train people up

Comparing approaches

If someone says “interface X is better than interface Y,”

- better for what??
- depends on what??
  - setting, person, back end, evaluation metric, ...?

Don't say this!!!!!

Another version: “X needs Y.”

- for what end?

Experimentation: assignment of subjects to conditions

Between-subjects – each subject experiences one condition

- variance added because of individual differences

Within-subjects

- ordering effects
  - counterbalance order
- even with counterbalancing order, variance is added
- data from a counterbalanced within-subjects experiment contains a between-subjects experiment

Counterbalancing with >2 conditions – Latin square (i.e. round robin)

Experimentation: assignment of subjects to conditions

Random assignment – spreads confounding factors equally amongst conditions

Experimentation: assignment

You can counterbalance assignment to reduce impact of confounding factors.

- e.g. typing speed on a keyboard interface experiment
- in effect, helping the law of large numbers work faster by reducing variance in legitimate ways

Experimentation: assignment

offline counterbalancing:

1. order all participants based on confounding variable,
2. pair them along that ordering, and
3. randomly assign each pair

To counterbalance offline, you must know and pretest all participants before any experimentation.

Experimentation: assignment

online counterbalancing:

1. pick a threshold for the confounding variable that's around your expected median.
2. when, say, a "low" person comes in and the number of "low" participants in each condition is even, randomly assign the person; if uneven, assign the person to the condition with one less participant

can also compare across threshold for interaction with independent variable(s)

online counterbalancing is applied when participants are only known serially as they come in

Experimentation: assignment

danger of "regression to the mean"

occurs when:

- divide subjects into groups with pretest to divide into conditions
- if grouping comes largely from randomness, the posttest group means will usually move from the pretest group means towards the overall mean (since some of the group assignment will be from noise)
  - e.g., grouping quarters as "heady" or "taily", manipulating in a meaningless way (e.g., tapping bagel on them to give them a snack), and then testing their headiness

not an issue with using pretest for counterbalanced assignment, where both pretest-defined groups are assigned to both conditions

- e.g., the heady and taily quarters would be divided among snack and no snack

Experimentation: design

make clear goals:

- limit scope
- create hypothesis ahead of time if possible
  - or alternatively, a clear yes/no question that your experiment will answer

plan it out:

- research questions
- data to be collected
- set-up for experiment
- roles of researchers

try to have two people present: one to facilitate, one to gather data (e.g., taking notes)

Experimentation: design

create concrete tasks

experimental details:

- order of tasks
- training level of participants
- what occurs when someone doesn't finish
- decide beforehand whether and how to intervene to help participant move along

Experimentation: design

ethical considerations

- voluntary consent
- avoid pressure to participate
- can stop any time
- remind: testing the system, not them
  - to help them avoid getting upset at mistakes/failure
  - but in certain situations this might harm motivation to push through challenges rather than blaming the system and giving up

Experimentation: design

pilot experiments

Klemmer recommends two:

- with a colleague to get materials ready
- with one real user who doesn't know the design

Often a number of subjects are needed to choose the dependent variable(s) for the real experiment

Experimentation: collecting data

methods for collecting process data (i.e., details about the interaction):

- notebook for general notes (a ha! moments that lead to design changes, what worked well, stories, or problems)
- video and/or screen recording
- ask users to think aloud through experiment - thoughts, goals, questions that arise, what they read
- ask questions
  - vague questions can be better to get a relevant response
  - can't assume that people's answers are true reflections of their mental processes
    - e.g., reasons for why they do something or anything with an subconscious, embarrassing, etc. component

Experimentation: collecting data

methods for collecting process data (i.e., details about the interaction):

- interaction record – with IML, you can freeze learner at any point and test it offline
  - before experiments, important to test that you can recreate an agent with an interaction record

Experimentation: collecting data

process data - observations of details of interaction; usually more qualitative

bottom-line data - summary of what happened, including evaluation metrics

- don't mix bottom-line data and think aloud (or any unnatural intervention)

debriefing - share your goals for their education and then get their thoughts after having the full picture of the study

Experimentation: collecting data

Metrics:

- Task completion time
- Task performance / errors
- Base rates (# of occurrences)
- Computational demands (time and memory)
- Human demands (time, cognitive load, # of input instances)
- Constraints on human input
- Likert-scale self report
  - e.g., NASA TLX - rates perceived workload on six different subscales: Mental Demand, Physical Demand, Temporal Demand, Performance, Effort, and Frustration

Experimentation: analysis

Analyzing your data in 3 questions:

1) What does my data look like?
   - explore your data graphically
   - plot all of your data
   - plot several different summaries

2) What are the overall numbers?
   - aggregate statistics for each conditions (mean and sd, usually)

3) Are the differences generalizable?
   - compute significance (p-values)
   - likelihood that results are due to chance

Pitfalls

Causal misattribution

- Correlation does not imply causation (by one def. of "imply")
- Casual causal claims – where claim isn't a contribution of the paper
  - e.g., "Because negative emotions degrade performance, we …"

Pitfalls

Lack of blindness

- Subject blindness
  - Please the experimenter bias
- Experimenter blindness
  - Experimenter can unintentionally influence the subject
  - Any coder should also be blind if possible

Run double-blind experiments when feasible

Resources

- Scott Klemmer's HCI course on Coursera
- Norman's *The Design of Everyday Things*
- Schneiderman and Plaisant's *Designing the User Interface*
- Dix et al.'s *Human-Computer Interaction*