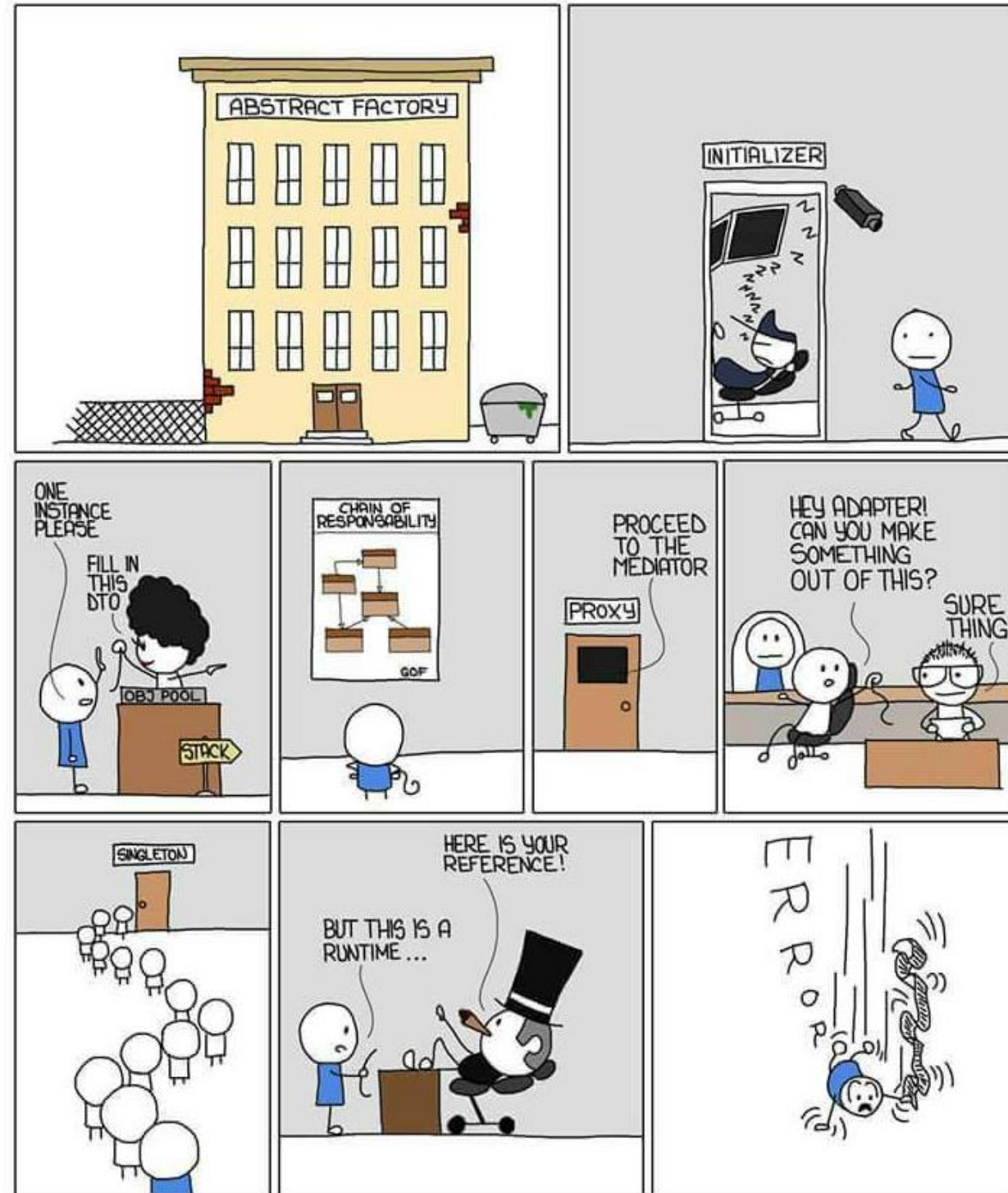


Code Inspection and the Brain



Review: Design Patterns

- Design patterns are reusable software that can apply to multiple problems
 - **Structural Design Patterns**
 - Hide implementation details
 - Create interfaces
 - *Adapter Design Pattern*
 - **Creation Design Patterns**
 - Enforce use of objects
 - *Factory Design Pattern*
 - **Behavioral Design Patterns**
 - Capture some relationship between objects or tasks on objects
 - **Publish-Subscribe Design Pattern**
 - **Template Design Pattern**



The Story So Far ...

- We want to deliver and support a quality software product
- We have covered many process and technical aspects (measurement, testing, static analysis, code inspection, design patterns, etc.)
- And we have focused on many human biases and weaknesses
- But we have not considered **human expertise and productivity**
 - SE is performed by humans: how do humans work?

One-Slide Summary

- We can investigate neural correlates of software engineering activities using **medical imaging**.
- **Top-down comprehension** based on semantic cues is more **efficient** (easier) than bottom-up comprehension.
- Neural representations of **programming** and **natural** languages are **distinct**. Classifiers can distinguish them based solely on brain activity. The same brain locations distinguish all three tasks. Greater **skill** accompanies a less-differentiated neural representation.

Code Review and Comprehension

- Developers spend more time understanding and **comprehending** code than any other activity
 - NASA: understanding > correctness for reuse
- **Code review** is a de facto standard
 - “Should we accept this commented patch?”
 - Mandated in Facebook, Google, etc.
 - One of the most effective techniques in software development

Previously: Readability Model

- We considered a descriptive readability metric
 - Blank lines are good?
 - Identifier length had no effect?
 - Identifier content entirely unexplored?
- Produced an award-winning metric
 - But still somewhat unsatisfactory
- Why not just **ask programmers** which features make code easy to read?

Why Use Medical Imaging?

- Unreliable Self-Reporting
- Inform Pedagogy
- Understand Expertise
- Retrain Aging Engineers
- Guide Technology Transfer
- Fundamental Understanding



Self-Reporting is Unreliable

- Economics: **revealed choice** (e.g., people say they would not support Wal-Mart but shop there anyway)
- Computer Science: 3 of top 4 features that people self-report (e.g., shorter functions) as making code maintainable are irrelevant
- Psychology: “A review of 55 studies in which self-evaluations of ability were compared with measures of performance showed a low mean validity coefficient (mean $r=.29$) with high variability ($SD=0.25$).”
- [Mabe and West. *Validity of self-evaluation of ability: a review and meta-analysis*. J. Applied Psych.]

Pedagogy and Expertise

- Informal model: your mind is a computer, with the brain and neurons as hardware and your memories as software
- When you learn something new, is that just new “software”, or does mental “hardware” change?
- Answer: both! We can observe “learning” by looking at physical brain structures.
- [Bassett et al. *Dynamic reconfiguration of human brain networks during learning*. PNAS.]
[Ritchey et al. *Functional connectivity relationships predict similarities in task activation and pattern information during associative memory encoding*. J. Cognitive Neuroscience.]

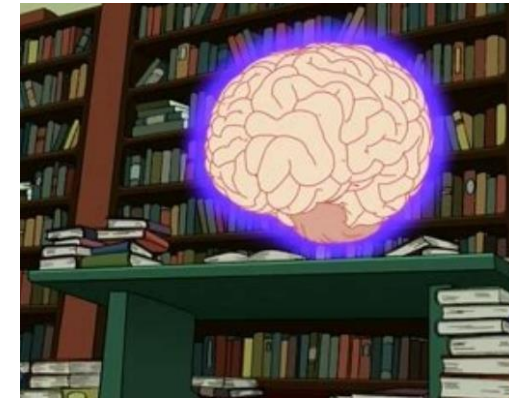
Retraining Aging Engineers

- **Older** workers are **retraining** into engineering and companies are **hiring** older engineers
- [J. Wright. *In-demand and aging: A look at engineers and engineering technicians in the workforce.*]
- Older humans show more **diffuse** patterns of neural activity, recruiting nearby parts of the brain to help
 - “The occipital reduction is consistent with the view that sensory processing decline is a common cause in cognitive aging, and the prefrontal increase may reflect **functional compensation.**”
- [Cabeza et al. *Task-independent and task-specific age effects on brain activity during working memory, visual attention and episodic retrieval.* Cereb. Cortex.]

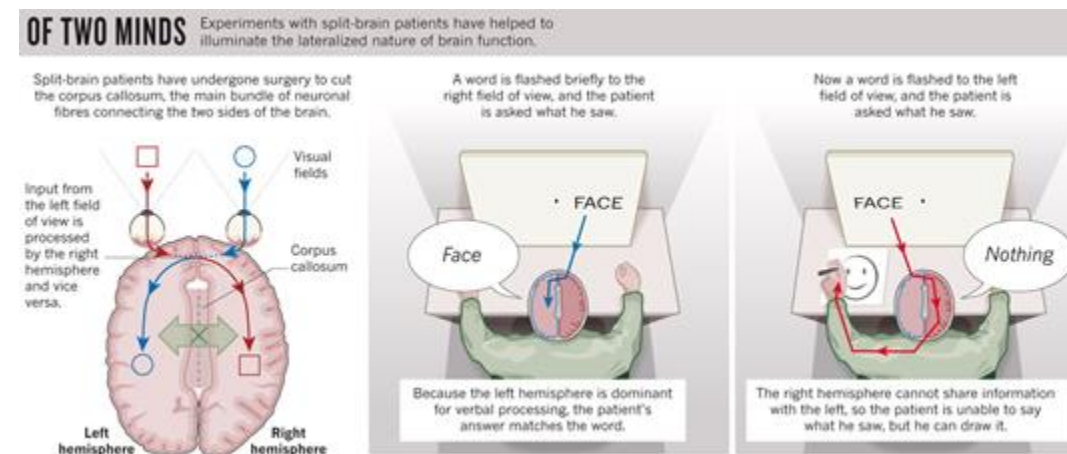
Guiding Technology Transfer

- **Technology transfer** involves turning a research idea (like a dynamic analysis) into an effective product that is actually used
- Fault localization can produce ranked lists and false positives that humans dislike (and don't use) [Parnin and Orso reading]
- Humans and tools can disagree on what is a false positive [Bessey et al. reading]
- We need better models so that tools can produce results that humans like, because human champions are essential
- “A key part of technology transfer between research and development organizations is to have champions ...” [Ball et al. *SLAM and static driver verifier: technology transfer of formal methods inside Microsoft*. Integrated Formal Methods.]

Examining the Brain

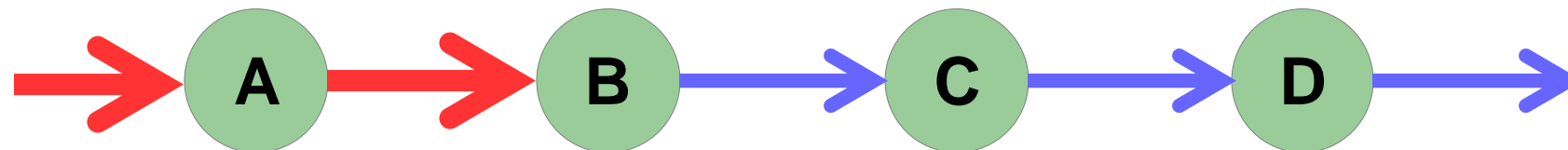


- In psychology, many neural studies involve patients with **epilepsy** who have the corpus callosum severed to treat seizures
 - “Since you've already cut the skull open, while you're in there, put in this electrode ...”
- We need a **non-invasive** way to observe what is happening inside the human brain



How brain works (a super simplified explanation)

- Your brain uses energy to carry out activities
- There is no fat stored in the brain
- So energy must be transported into the brain
- This is done via **oxygen** in the blood
- If we could only tell oxygen-rich blood from oxygen-poor blood, we could tell which part of the brain is using energy and is thus active!

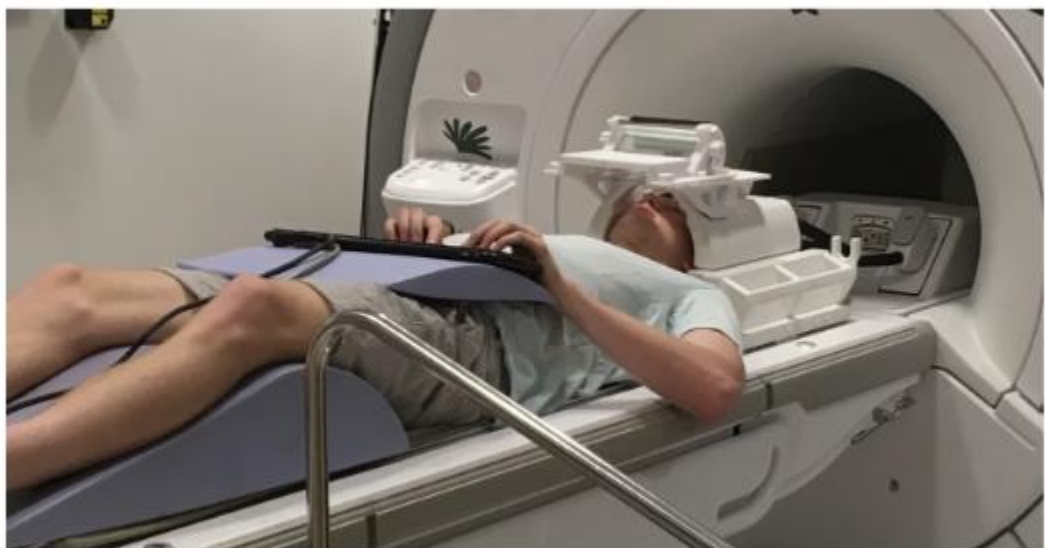


The BOLD and the Beautiful

- Oxygen-rich and oxygen-poor blood have **different** electromagnetic properties
- That is, they respond differently to certain powerful magnetic fields
 - Can be detected by a **magnetic resonance (MR)** scanner
 - Note: powerful magnets do not hurt humans
- The **blood-oxygen level dependent (BOLD)** response is the ratio of oxygenated to deoxygenated hemoglobin

fMRI

- **Functional magnetic resonance imaging (fMRI)** is a non-invasive technique for probing the neurobiological substrates of various cognitive functions *in vivo* by measuring the BOLD signal
 - Millimeter scale (>> EEG or PET, etc.)
 - It is a relatively recent technique; pioneering uses are associated with psychology
 - Especially in CS: first research using fMRI to investigate software engineering was in 2014



A Study in Contrasts

- A subject might be doing multiple things
 - Reading code, breathing, being nervous
- How can we tell if an observed pattern of activation corresponds to one action?
- Experimental design and control
 - A = “reading code + breathing + ...”
 - B = “writing code + breathing + ...”
- The **contrast** A-B shows patterns of brain activation that *vary* between the stimuli/tasks

High-Level Question #1

What actually makes it
easier to read code?

One Model of Code Comprehension

- **Top-down comprehension** refers to cognitive processes in which experience and expectation and semantic cues (**beacons**) guide the understanding of source code
- **Plans** are knowledge structures representing semantic and syntactic software patterns
 - Example: the identifier **bubbleSort** encourages (**primes**) programmers to expect elements of that algorithm, such as array element swaps

Another Model of Code Comprehension

- **Bottom-up comprehension** refers to cognitive processes in which meaning is obtained from every individual statement and then synthesized into a holistic understanding
- Programmers may hold these elements in **working memory** and then abstract those pieces of information into higher-order concepts: this is called **semantic chunking**

Which is Correct?

- Researchers have debated and theorized
- Perhaps ...
 - You mostly use bottom-up comprehension because meaning must be extracted from perceptual and syntactic information
 - You mostly avoid bottom-up comprehension because it is tedious
- Let's find out!
- [Siegmund et al. *Measuring Neural Efficiency of Program Comprehension*. ESEC/FSE.]

Neural Efficiency

- **Neural efficiency** is the phenomenon where lower brain activation indicates that a cognitive process is more efficient and thereby perceived as **easier**
- More details in upcoming lecture on expertise
- Informally, one difference between experts and novices is that experts use less energy than novices to solve the same task
 - The task feels easier to the expert
 - Experts literally do not have to work as hard

Trivia: Arts and Comics

- Identify the Renaissance artist associated with each work.



A



B



D



C

Trivia: Arts and Comics

- Identify the Renaissance artist associated with each work.

The Creation of Adam: Michelangelo

B



A

Mona Lisa: Leonardo da Vinci



D

The School of Athens: Raphael



C

David: Donatello

Trivia: Astronauts

- This physicist and astronaut became the first American woman in space in 1983 (after USSR cosmonauts Tereshkova '63 and Savitskaya '82) and the youngest American in space (aged 32). After flying twice on the Orbiter Challenger, she left NASA in 1987, working as a professor at UCSD before dying of pancreatic cancer in 2012.

Trivia: Astronauts

- This physicist and astronaut became the first American woman in space in 1983 (after USSR cosmonauts Tereshkova '63 and Savitskaya '82) and the youngest American in space (aged 32). After flying twice on the Orbiter Challenger, she left NASA in 1987, working as a professor at UCSD before dying of pancreatic cancer in 2012.

Sally Ride



Experimental Setup 1

- Start with code snippets
 - 8-19 lines, 20-30 seconds to read, etc.
- Participants look at code and determine if it implements the same thing as a snippet shown earlier during a pre-training session
 - No scrolling, no typing, etc.
- N=11 student participants

Experimental Setup 2

- Two-way controlled experiment
 - **Top-down** comprehension
 - **Bottom-up** comprehension
- Four-way controlled experiment
 - Code **with beacons** and **pretty-printing**
 - Code **with beacons** and **disrupted layout**
 - Code **without beacons** but with **pretty-printing**
 - Code **without beacons** but with **disrupted layout**

Example Stimuli

Listing 1: Code snippet with beacons and pretty-printed layout (*BY, LP*)

```
1 public float arrayAverage(int[] array) {
2     int counter = 0;
3     int sum = 0;
4
5     while (counter < array.length) {
6         sum = sum + array[counter];
7         counter = counter + 1;
8     }
9
10    float average = sum / (float) counter;
11    return average;
12 }
```

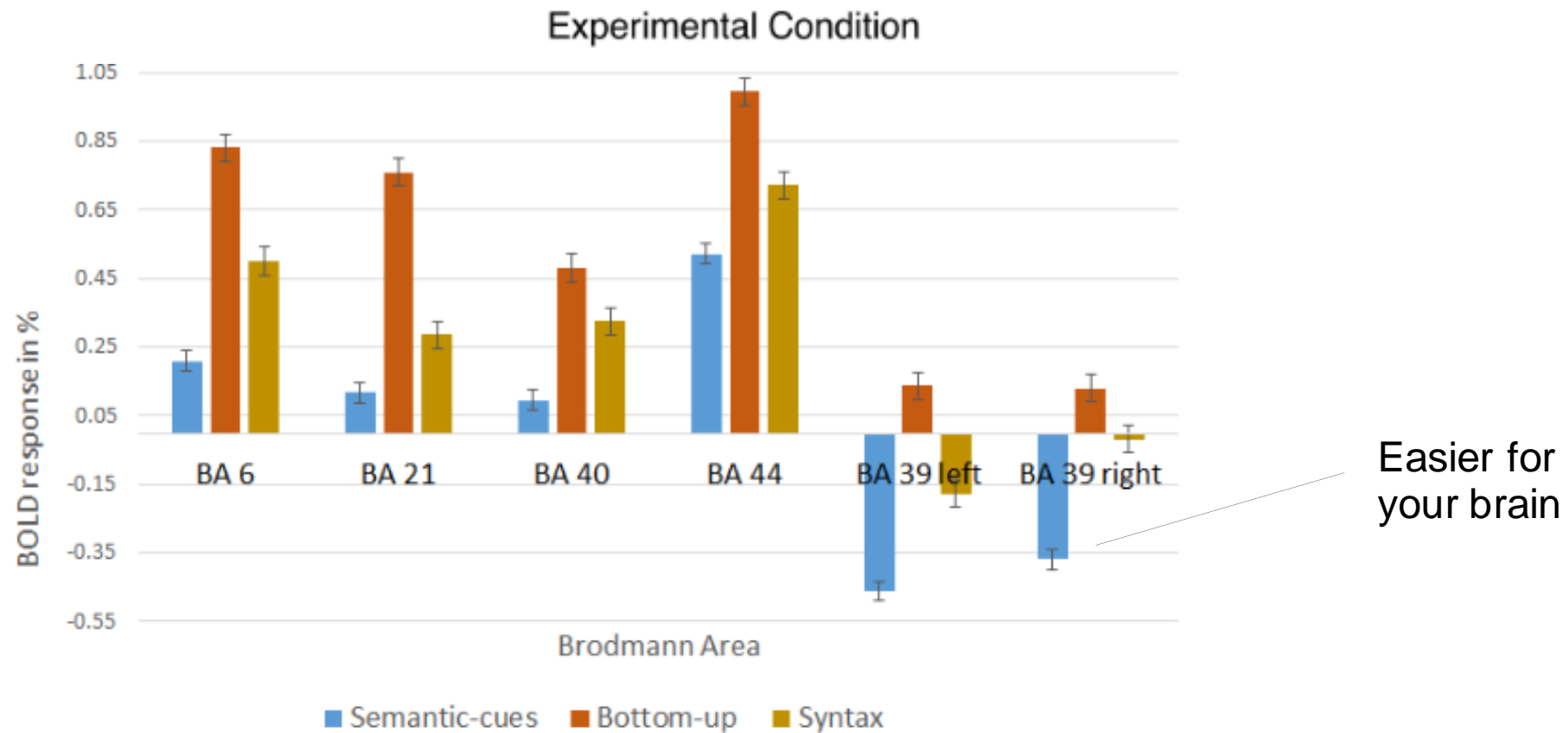
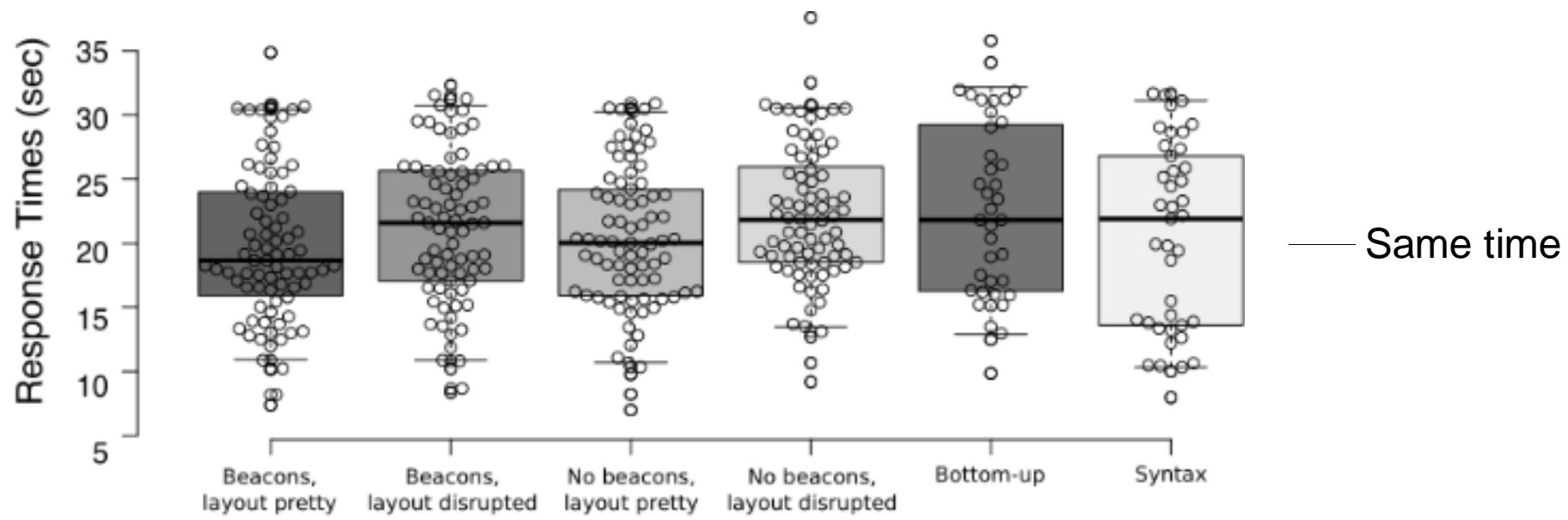
Listing 2: Code snippet with no beacons and disrupted layout (*BN, LD*)

```
1 public float ayyaoAwyaky(int[] array) {
2     int
3         mgqakyy
4     = 0;
5     int sum = 0;
6
7     while (mgqakyy
8         < array.length) {
9         sum =
10        sum + array[mgqakyy];
11        mgqakyy
12        = mgqakyy + 1;
13    }
14
15    float average
16        = sum /
17        (float) mgqakyy;
18    return
19        average;
20 }
```

Bottom-up: obfuscate identifier names so that they show usage but not meaning
Top-down: beacons like method names and layout

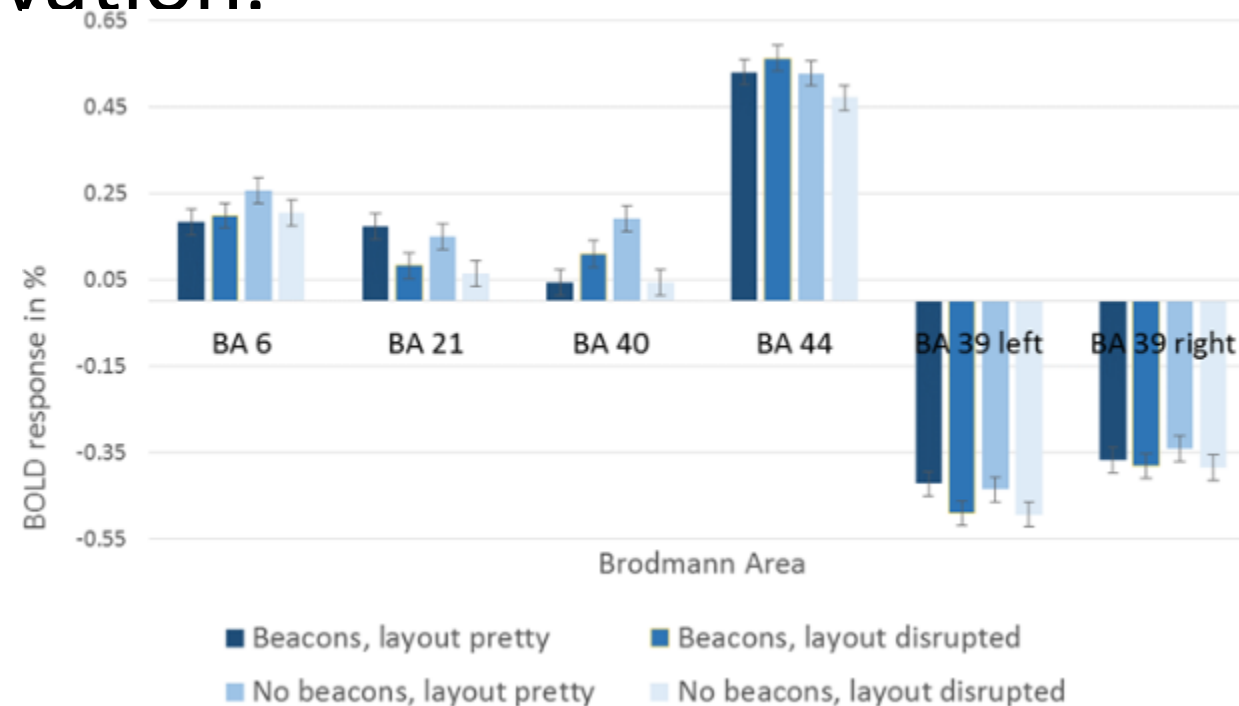
Result 1

- What is the difference between bottom-up program comprehension and comprehension via semantic cues?
- **Brain area 39** deactivates during comprehension based on semantic cues and activates during bottom-up comprehension
- Comprehension based on **semantic cues requires less cognitive effort** than bottom-up comprehension:
 - Response times are identical
 - Energy use “across the board” is lower



Result 2

- How do layout and beacons in source code influence program comprehension?
- As far as we can tell, they do *not*. No significant differences in brain activation.



Comprehension Take-Home #1

- Program comprehension based on **semantic cues** is a very **efficient** process for understanding source code compared with the tedious, statement-by-statement process employed during bottom-up comprehension.
- When you are choosing identifier names and perhaps writing comments and documentation, emphasize beacons, cues and semantic plans: hint at the program's **purpose** or idiom
 - cf. high-level “**why**” vs. low-level “what” documentation

High-Level Question #2

- Is reading code more like doing **math** or more like reading **prose**?

Experimental Design: 3 Tasks

- Code Comprehension
- Code Review (top 100 GitHub repos)
- Prose Review (College Board SAT, etc.)

```
if (l < 0) {
    l = (Z_STRLEN_P(orig_str) - f) + 1;
    if (l < 0) {
        l = 0;
    }
}
```

Given the following values for variables, the value of l after executing this code will be 0.

l = -2
Z_STRLEN_P(orig_str) = 10
f = 9

(a) Code Comprehension

```
skynet-src/skynet_socket.c
47 47 sm->ud = result->ud;
48 48 if (padding) {
49 49     sm->buffer = NULL;
50 50     strcpy((char*)(sm+1), result->data);
51 51     strcpy((char*)(sm+1), result->data, strlen(result->data));
52 52 } else {
53 53     sm->buffer = result->data;
54 54 }
```

avoid potential memory leakage.

If not packing '\0' into the message then maybe we should be careful when strcpy() between strings. Current implementation may cause memory leakage somehow?

(b) Code Review


The study revealed that the conditions of a cat's teeth, eyes, and fur are good indiees-indexes of the cat's health. Importantly, Note that the study only considered male cats, so these results are not necessarily generalizable. However, another independent study shows that females with these characteristics live longer like-than the males do.

(c) Prose Review

Experiment Setup and Data

- 29 grads and undergrads (38% women)
 - Right-handed, native English speakers, corrected-to-normal vision, etc.
- Placed in fMRI, computer projection displayed via mirror
- A single participant completing four 11-minute runs produces 399,344,400 floating point numbers of data (153,594 voxels × 650 volumes × 4 runs)

Dead Fish and Software Bugs



Neural correlates of interspecies perspective taking in the post-mortem Atlantic Salmon: An argument for multiple comparisons correction

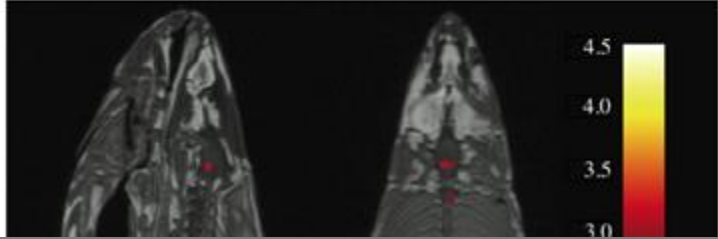
Craig M. Bennett¹, Abigail A. Baird², Michael B. Miller¹, and George L. Wolford³

¹ Psychology Department, University of California Santa Barbara, Santa Barbara, CA; ² Department of Psychology, Vassar College, Poughkeepsie, NY; ³ Department of Psychological & Brain Sciences, Dartmouth College, Hanover, NH

INTRODUCTION

With the extreme dimensionality of functional neuroimaging data comes extreme risk for false positives. Across the 130,000 voxels in a typical fMRI volume the probability of a false positive is almost certain. Correction for multiple comparisons should be completed with these datasets, but is often ignored by investigators. To illustrate the magnitude of the problem we carried out a real experiment that demonstrates the danger of not correcting for chance properly.

GLM RESULTS



Cluster failure: Why fMRI inferences for spatial extent have inflated false-positive rates

Anders Eklund^{a,b,c,1}, Thomas E. Nichols^{d,e}, and Hans Knutsson^{a,c}

^aDivision of Medical Informatics, Department of Biomedical Engineering, Linköping University, S-581 85 Linköping, Sweden; ^bDivision of Statistics and Machine Learning, Department of Computer and Information Science, Linköping University, S-581 83 Linköping, Sweden; ^cCenter for Medical Image Science and Visualization, Linköping University, S-581 83 Linköping, Sweden; ^dDepartment of Statistics, University of Warwick, Coventry CV4 7AL, United Kingdom; and ^eWMG, University of Warwick, Coventry CV4 7AL, United Kingdom

Edited by Emery N. Brown, Massachusetts General Hospital, Boston, MA, and approved May 17, 2016 (received for review February 12, 2016)

The most widely used task functional magnetic resonance imaging (fMRI) analyses use parametric statistical methods that depend on a variety of assumptions. In this work, we use real resting-state data

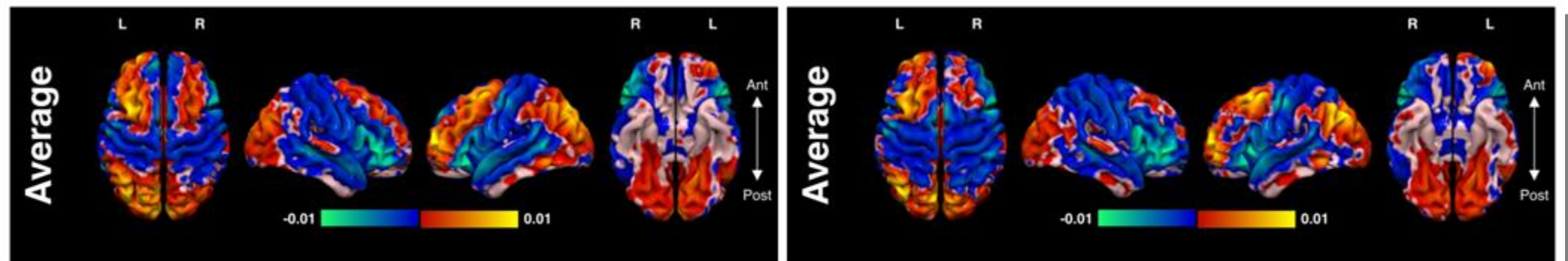
(FWE), the chance of one or more false positives, and empirically measure the FWE as the proportion of analyses that give rise to any significant results. Here, we consider both two-sample and

Preprocessing and Overfitting

- A significant challenge in fMRI analysis is **processing the data correctly**
- We cannot naively build a model from 150,000 features and 100 labeled instances
- Align and unwarp data, coregistered with a high-resolution anatomical scan, generalized linear models, high pass filters, robust weighted least squares, multivariate Gaussian process classification, feature selection via Automated Anatomical Labeling atlas, kernel function, expectation propagation ...

Results: Mind Reading

- We can classify which task a participant is undertaking based solely on brain activity
 - Balanced accuracy 79%, $p < .001$
- These results suggest that Code Review, Code Comprehension, and Prose Review all have largely **distinct neural representations**

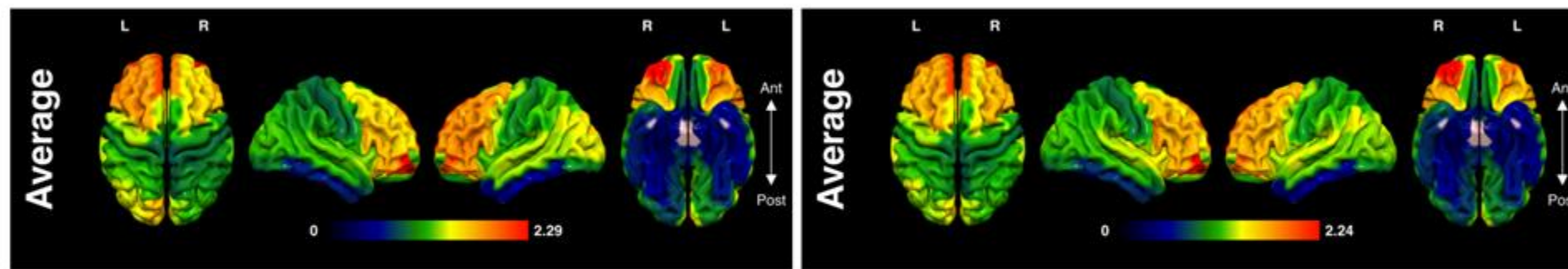


(a) Code Comprehension vs. Prose Review

(b) Code Review vs. Prose Review

Results: Can we relate tasks to brain regions?

- Near-perfect correspondence: $r=0.99$, $p<.001$



(a) Code Comprehension vs. Prose Review

(b) Code Review vs. Prose Review

- A wide swath of **prefrontal regions** known to be involved in higher-order cognition (executive control, decision-making, language, conflict monitoring, etc.) were highly weighted
- Activity in those areas strongly drove the distinction between code and prose processing

Results: Can we relate expertise to classification accuracy?

- “Expertise” = (CS GPA) * (CS Credits Taken)
- How accurately our model distinguishes between Code Comprehension and Prose **significantly predicted expertise** ($r = -0.44$, $p=0.016$)
- The inverse relationship between accuracy and expertise suggests that, as one develops more skill in coding, the neural representations of code and prose are less differentiable. **That is, programming languages are treated more like natural languages with greater expertise.**

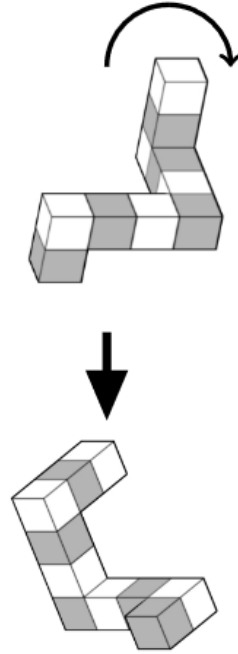
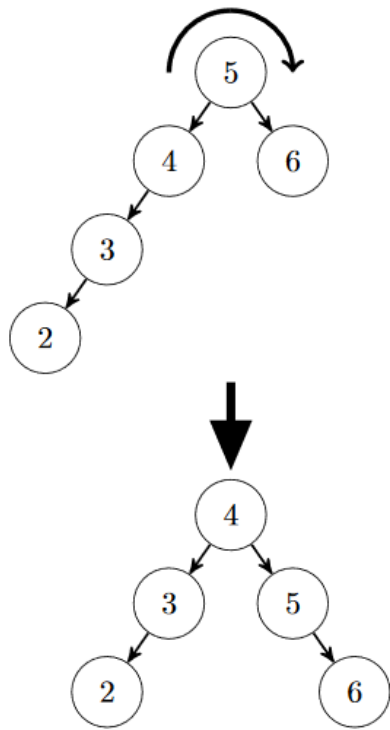
Comprehension Take-Home #2

- Neural representations of **programming** and **natural** languages are **distinct**. Classifiers can distinguish them based solely on brain activity. The same brain locations distinguish all three tasks. Greater **skill** accompanies a less-differentiated neural representation.
- These studies are still exploratory
- The area is wide open for future work
 - Social relationships, experts, writing code
- [Floyd et al. *Decoding the representation of code in the brain: An fMRI study of code review and expertise*. ICSE. Best paper award.]

Current and Future Studies

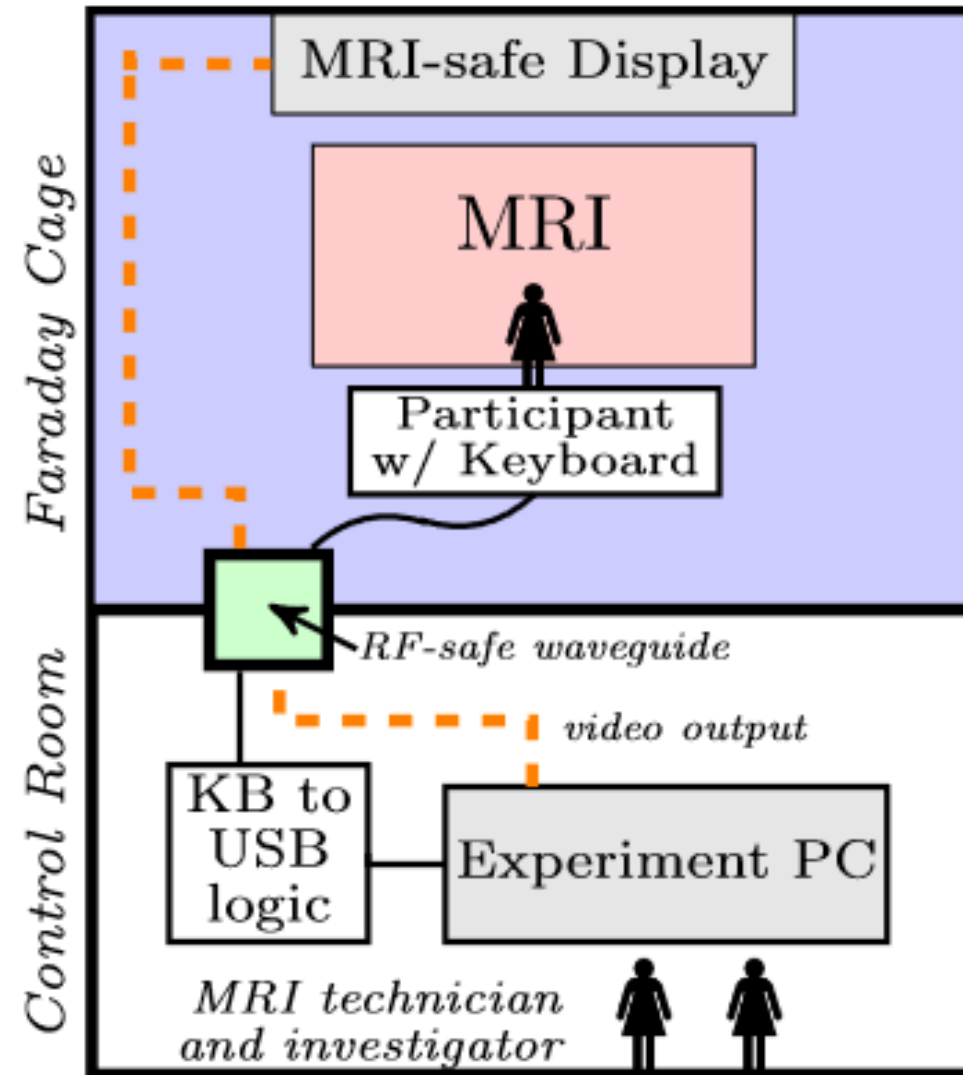
- Social relationships (boss over shoulder)
- **Spatial processing** (trees vs. tetris)
- **Patch provenance** (cheating)
- Industrial expertise (replicate protocol)
- **Writing code** (fMRI-safe keyboard)
- Transcranial magnetic stimulation (read-write)

- Does any of this sound interesting? ...

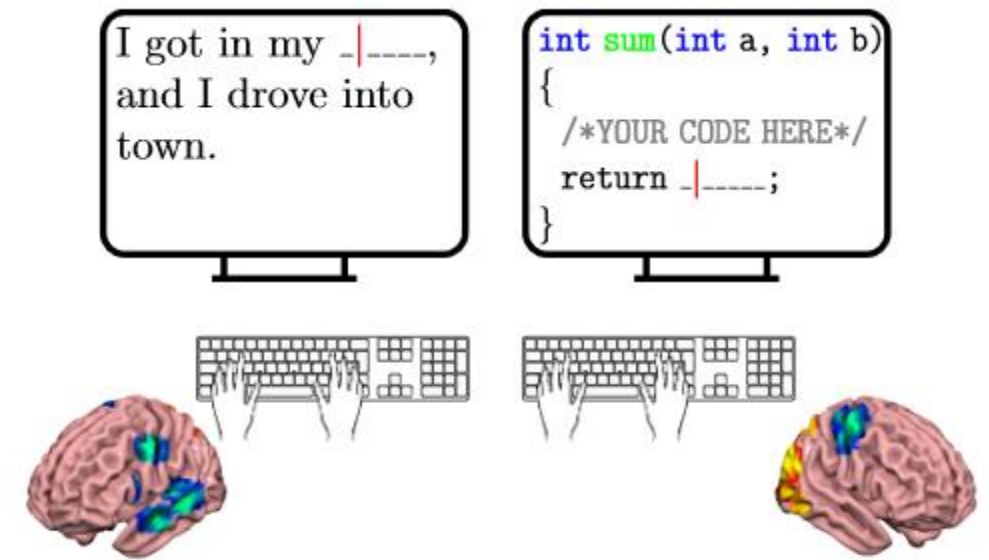


- fMRI and fNIRS show that mental rotation and data structure tasks use the same parts of the brain (e.g., 95% voxel similarity, $p < 0.01$)
- The brain works harder (cognitive load) to solve more difficult (Big-Oh) CS problems
- [Huang et al. *Distilling Neural Representations of Data Structure Manipulation using fMRI and fNIRS*. ICSE.]

Writing Code in an fMRI?



Code and Prose Writing



- **Code writing** requires significantly more activity in parts of the brain associated with **top-down control, planning, and categorization** than **prose writing**
- **Coding** involves right-lateralized brain regions associated with **attention, memory, planning, and spatial ability**
- [Krueger et al. *Neurological Divide: An fMRI Study of Prose and Code Writing*. ICSE.]