Teacher Guide: Chain Rule and Backpropagation Activity

For AP Calculus BC Students (Juniors/Seniors)

Overview

This inquiry-based activity helps students discover how the chain rule—a concept they're learning in calculus—powers the training of neural networks and all of modern AI. Students work through hands-on calculations, make predictions, and use Python simulations to see backpropagation in action.

Learning Objectives

By the end of this activity, students will be able to:

1. Apply the chain rule to calculate gradients in a multi-layer network
2. Explain how neural networks use backpropagation to learn from data
3. Connect abstract calculus concepts to real-world AI applications
4. Understand why longer networks require longer chains of derivatives

Materials Needed

Required:

- Student worksheets (1 per student or 1 per group)
- Calculators
- Large poster paper or whiteboards for group work
- Markers

Optional but Recommended:

- Computer lab access for Python simulation
- Projector for demonstrations
- Printed answer key for teacher reference

Time Requirements

Flexible Format Options:

Option 1: Single 50-minute Period

- Brief intro (5 min)
- Guided discovery (20 min)
- Whole class discussion (10 min)
- Python demo by teacher (10 min)

- Wrap-up (5 min)

Option 2: Two 45-minute Periods

- **Day 1:** Complete discovery activity and discussion
- **Day 2:** Students run Python simulations in computer lab

Option 3: Extended 90-minute Block

- Complete all components with deeper exploration
- Time for extension challenges

Lesson Plan: Detailed Timeline

Phase 1: The Challenge (5 minutes)

Teacher Script: "Today you're going to discover something amazing: the chain rule you've been learning isn't just abstract math—it's literally how AI learns. You're going to figure out how neural networks adjust their parameters, using only the calculus you already know. I won't tell you the answer—you'll discover it yourself."

Setup:

1. Display the simple network diagram on board:

   Input (x) → [×3, +w_1] → Hidden (h) → [×2, +w_2] → Output (y)

2. Present the scenario:

   - "This network predicted 17, but the answer should be 20"
   - "How should it adjust w_1 and w_2 to do better?"

3. Distribute worksheets and organize students into groups of 3-4

Key Point: Don't mention "chain rule" or "backpropagation" yet. Let them discover it!

Phase 2: Guided Exploration (15-20 minutes)

Circulate and facilitate as groups work through Tasks 1-3 on the worksheet.

Common Student Approaches:

Task 1 (Forward Pass):

- Most groups will correctly calculate h = 7, y = 17, E = 4.5
- If struggling: "What does the equation $h = 3x + w_1$ mean? Can you plug in the numbers?"

Task 2 (Exploring w_2):

- Students should discover that increasing w_2 increases y and changes error
- Look for: Are they noticing the pattern in how error changes?
- Guiding question: "When w_2 changes by 0.1, how much does y change?"

Task 3 (Exploring w_1):

- This is harder because w_1's effect is indirect (through h)
- Students should notice w_1 has a bigger effect than w_2
- Key insight: "Why does w_1 affect the error differently than w_2?"

Expected Student Observations:

✓ "When we increase w_2 by 0.1, error decreases by about 0.3" ✓ "When we increase w_1 by 0.1, error decreases by about 0.6" ✓ "w_1 seems to have a bigger effect" ✓ "The change in error seems proportional to the change in weight"

If Groups Finish Early:

- "Can you predict what happens if we change both weights at once?"
- "Why do you think w_1 has double the effect of w_2?"

Phase 3: The Calculus Connection (10-15 minutes)

This is where the "aha!" moment happens. Guide groups through Tasks 4-6.

Task 4 (Direct Derivatives):

Common Issues:

- Students may forget that h is constant with respect to w_2
- Remind them: "When we take dy/dw_2, treat everything except w_2 as a constant"

Expected Answers:

- $dy/dw_2 = 1$ (because $y = 2h + w_2$)
- $dE/dy = (y - 20) = -3$ (at y = 17)

Task 5 (Chain Rule for w_2):

The Key Question: "How do we find dE/dw_2 when E depends on y, and y depends on w_2?"

Let them struggle briefly, then guide:

- "What calculus rule do we use when functions are composed?"

- Wait for "chain rule!"
- "Exactly! Write it out."

Expected Answer:

$dE/dw_2 = (dE/dy) \times (dy/dw_2)$

$= (-3) \times (1)$

$= -3$

Verification: "Does this match your experimental value from Task 2? Yes!"

Task 6 (Chain Rule for w_1):

This requires a **longer chain**: $w_1 \rightarrow h \rightarrow y \rightarrow E$

Scaffolding Questions:

- "What does w_1 directly affect?" (Answer: h)
- "What does h affect?" (Answer: y)
- "What does y affect?" (Answer: E)
- "So we have a chain: $w_1 \rightarrow h \rightarrow y \rightarrow E$. How do we handle that?"

Expected Answer:

$dE/dw_1 = (dE/dy) \times (dy/dh) \times (dh/dw_1)$

$= (-3) \times (2) \times (1)$

$= -6$

Critical Discussion Point: "Why did we need three derivatives for w_1 but only two for w_2?"

- Answer: w_1's effect goes through more layers

Phase 4: The Big Picture (10 minutes)

Gallery Walk (Optional):

- Groups post their poster boards
- 3-minute rotation to see other solutions
- Discussion of different approaches

Whole Class Discussion:

Guiding Questions:

1. "What mathematical tool did everyone need to use?"
    - **Answer:** Chain rule
2. "Why was w_1 harder than w_2?"
    - **Answer:** Longer chain of dependencies
3. "Our network had 2 weights. What if it had 1000 weights?"
    - Let them think...
    - **Answer:** Same process, just more chains
4. "What if we had 10 layers instead of 2?"
    - **Answer:** Chains get longer (10 derivatives instead of 2-3)

The Reveal: "What you discovered is called **backpropagation**—the fundamental algorithm that trains every neural network. It's literally just the chain rule, applied systematically through all the layers.

- ChatGPT: ~100 layers, billions of weights
- All use the same chain rule you practiced today
- Your calculus homework is literally the foundation of AI!"

Phase 5: Python Simulation (10-20 minutes)

Option A: Teacher Demonstration If no computer lab access, run the Python simulation and project results.

What to Show:

1. Run backpropagation_simulator.py
2. Pause at Part 1 to verify their worksheet calculations
3. Show Part 2 animation of weights converging
4. Display the error reduction graph
5. Show Part 5 (learning from real data) to connect to practical applications

Key Talking Points:

- "See how the weights change in exactly the direction we calculated?"
- "The error decreases smoothly because we're following the gradient"
- "This same process works on real data with thousands of examples"

Option B: Student Exploration If in computer lab, have students:

1. Open the Jupyter notebook
2. Work through cells in order
3. Try changing weights in Part 3 (interactive exploration)
4. Observe how gradients change with different weight values

Observation Questions:

- "What happens when you set w_2 to exactly the right value?"
- "Can you find weights where the gradient is zero?"
- "What happens if the learning rate is too large?"

Assessment Options

Formative Assessment (During Activity):

Circulate and listen for:

- ✓ Correct application of chain rule
- ✓ Understanding of gradient direction
- ✓ Connection between derivatives and weight changes
- ✓ Explanation of why longer chains are needed for earlier layers

Check-in Questions:

- "Explain why we multiply these derivatives together"
- "How would you find the gradient if we added another layer?"
- "Why does the network need to know these gradients?"

Summative Assessment Options:

Option 1: Written Reflection (10 minutes) Prompt: "Explain how the chain rule you learned in calculus is used to train neural networks. Use the specific example from today to illustrate your explanation."

Option 2: Extension Problem Give students a 3-layer network and ask them to:

1. Calculate the forward pass
2. Derive the gradient formulas using the chain rule
3. Explain why the first weight requires the longest chain

Option 3: Real-World Connection "Research one application of neural networks (face recognition, language translation, etc.) and write a paragraph explaining how the chain rule is essential to making it work."

Common Student Difficulties & Solutions

Difficulty 1: "I don't see why we multiply the derivatives"

Solution: Use the rate-of-change interpretation:

- "If y changes by 2 when h changes by 1 (dy/dh = 2)"
- "And h changes by 1 when w_1 changes by 1 ($dh/dw_1 = 1$)"
- "Then y changes by 2 when w_1 changes by 1"
- "That's multiplication: $dy/dw_1 = (dy/dh) \times (dh/dw_1)$"

Difficulty 2: "Why is it called 'backward' propagation?"

Solution:

- "Forward pass: Input → Hidden → Output → Error"
- "Backward pass: Error → Output → Hidden → Input"
- "We calculate gradients in reverse order"
- Draw arrows on the board showing both directions

Difficulty 3: "This seems too simple to power real AI"

Solution:

- "You're right that real networks are more complex"
- "But the fundamental principle is identical"
- "GPT-4: 100+ layers, but still just chaining derivatives"
- "The math scales up beautifully"

Difficulty 4: Computational mistakes in Task 2-3

Solution:

- Provide a table format
- Encourage calculator use
- Pair-check answers
- Focus on patterns, not precise numbers

Difficulty 5: "What's a learning rate?"

Solution:

- "It controls how big our steps are"
- "Too large: we overshoot the minimum"
- "Too small: training takes forever"
- "It's like walking downhill—do you take tiny steps or big strides?"

Extensions & Modifications

For Advanced Students:

1. **Nonlinear Activation Functions:**

    - Introduce sigmoid: $\sigma(z) = 1/(1 + e^{-z})$
    - Derive: $\sigma'(z) = \sigma(z)(1 - \sigma(z))$
    - Calculate gradients with this extra step

2. **Matrix Formulation:**

    - Express network in matrix form
    - Show how matrix chain rule works
    - Connect to linear algebra

3. **Vanishing Gradients:**

    - "What happens in a 100-layer network?"
    - Explore why gradients get very small
    - Discuss modern solutions (ResNets, etc.)

4. **Programming Challenge:**

    - Implement backpropagation from scratch
    - Train on real dataset (MNIST digits)
    - Visualize learned weights

For Students Needing Support:

1. **Pre-calculate some values**

    - Provide completed Task 1
    - Focus on Tasks 2-5

2. **Reduce numerical complexity**

    - Use simpler numbers ($w_1 = 1$, $w_2 = 2$)
    - Use target = 10 instead of 20

3. **Visual aids**

    - Color-code the chain: $w_1 \rightarrow h \rightarrow y \rightarrow E$
    - Use physical cards to represent functions
    - Draw derivative arrows on network diagram

4. **Scaffolded worksheet**

- Fill-in-the-blank for derivative calculations
- More explicit hints
- Worked example side-by-side

Answer Key

Task 1: Forward Pass

- $h = 3(2) + 1 = 7$
- $y = 2(7) + 3 = 17$
- $E = 0.5(17 - 20)^2 = 4.5$

Task 2: Exploring w_2

w_2 Value	**h**	**New y**	**New Error**	**Change in E**
3.0	7	17.0	4.50	baseline
3.1	7	17.1	4.21	-0.29
2.9	7	16.9	4.81	+0.31

Should **increase** w_2 to reduce error.

Task 3: Exploring w_1

w_1 Value	**New h**	**New y**	**New Error**	**Change in E**
1.0	7.0	17.0	4.50	baseline
1.1	7.1	17.2	3.92	-0.58
0.9	6.9	16.8	5.12	+0.62

w_1 has a **bigger effect** (≈2× the effect of w_2). Should **increase** w_1.

Task 4: Direct Derivatives

- $dy/dw_2 = 1$
- $dE/dy = (y - 20) = 17 - 20 = -3$

Task 5: Chain Rule for w_2

$dE/dw_2 = (dE/dy) \times (dy/dw_2)$

$= (-3) \times (1)$

= -3

Task 6: Chain Rule for w_1

$dh/dw_1 = 1$

$dy/dh = 2$

$dE/dy = -3$

$dE/dw_1 = (dE/dy) \times (dy/dh) \times (dh/dw_1)$

$= (-3) \times (2) \times (1)$

$= -6$

Task 7: Why harder?

w_1 requires 3 derivatives (longer chain) because its effect must propagate through the hidden layer first.

Task 8: Weight Updates

$w_2\_new = 3 - 0.1(-3) = 3.3$

$w_1\_new = 1 - 0.1(-6) = 1.6$

$h\_new = 3(2) + 1.6 = 7.6$

$y\_new = 2(7.6) + 3.3 = 18.5$

$E\_new = 0.5(18.5 - 20)^2 \approx 1.125$

Error decreased from 4.5 to 1.125 ✓

Connection to Broader Curriculum

Mathematics:

- **Calculus BC:** Chain rule, derivatives, optimization
- **Linear Algebra:** Matrix multiplication, transformations
- **Statistics:** Mean squared error, loss functions

Computer Science:

- **Algorithms:** Gradient descent, iterative improvement
- **Data Structures:** Networks, graphs

- **Programming:** Implementing mathematical algorithms

Cross-Disciplinary:

- **Physics:** Optimization, energy minimization
- **Economics:** Marginal analysis (derivatives in context)
- **Biology:** Neural networks (biological inspiration)

Additional Resources

For Teachers:

Background Reading:

- "Deep Learning" by Goodfellow, Bengio, and Courville (Chapter 6)
- 3Blue1Brown YouTube series on neural networks
- "But what is a neural network?" video specifically

Python Resources:

- backpropagation_simulator.py (included)
- backpropagation_simulator.ipynb (Jupyter notebook version)
- Generated visualizations (PNG files)

For Students:

Videos:

- 3Blue1Brown: "But what is a neural network?"
- 3Blue1Brown: "Gradient descent, how neural networks learn"
- Khan Academy: Chain rule review

Interactive:

- TensorFlow Playground (playground.tensorflow.org)
- Neural Network Simulator (this activity's Python code)

Reading:

- "The Master Algorithm" by Pedro Domingos (accessible overview)
- "AI Superpowers" by Kai-Fu Lee (applications)

Troubleshooting

"The simulation won't run"

Common Issues:

1. Python not installed → Install Anaconda
2. Missing packages → Run: pip install numpy matplotlib
3. Jupyter not opening → Use .py script instead

"Students are lost on Task 5"

Intervention:

1. Pause the class
2. Do a mini-lesson on chain rule
3. Work through one example together
4. Release them to finish

"We're running out of time"

Priority Order:

1. Tasks 1-5 (essential)
2. Big picture discussion (essential)
3. Task 6 (important)
4. Python demo (valuable but optional)
5. Extension (skip if needed)

"Class finished early"

Extension Activities:

1. Network design: "Design a network for classifying images"
2. Research task: "Find 3 real applications of neural networks"
3. Math exploration: "What if we had 3 hidden layers?"
4. Programming: Try the Jupyter notebook

Reflection Questions (For Teachers)

After teaching this lesson, consider:

1. Did students make the connection between chain rule and AI unprompted?
2. Which tasks took longer than expected?
3. What additional scaffolding would help?

4. Were the Python visualizations effective?
5. How could this activity be improved?

Final Notes

Key Message to Emphasize:

The mathematics students learn in calculus class isn't just for tests—it's powering the AI revolution happening right now. Every time they practice the chain rule, they're building skills that are literally training ChatGPT, self-driving cars, and medical AI systems.