Week 3: Teaching Networks to Remember

Discovering RNNs, LSTMs, and the Memory Problem
Pre-Lab Exercise (No Programming Required)

NLP Course 2025 - Student Version

Time: 40 minutes
Objective: Discover why networks need memory and how gates solve the vanishing gradient problem.

Part 1: The Context Problem (10 minutes)

Why Order Matters

Task 1: Same words, different meanings
Rearrange these words to create two sentences with opposite meanings:

{dog, bites, man}

Sentence 1: _____
Sentence 2: _____

Task 2: Context tracking
Read this sentence word by word and track what you remember:
"The student who studied hard..."

1. After "The": I expect _____
2. After "student": I remember _____
3. After "who": I expect _____
4. After "studied hard": I'm waiting for _____

Task 3: What information are you maintaining?
List three things your brain tracks while reading:

- _____
- _____
- _____

Think About It

Word embeddings (Week 2) treat "dog bites man" and "man bites dog" as identical - they have the same word vectors. How can we teach networks that order matters?

Part 2: Building Memory - Discovering Hidden States (10 minutes)

Designing Memory for Networks

Scenario: You're teaching a computer to read "The cat sat on the mat" word by word.
Task 1: Design your memory system
The computer can only see one word at a time. How would you help it remember previous words?
Your design:

Task 2: Memory updates
Fill in how memory should update at each step:

Current Word	Previous Memory	New Memory Should Contain
"The"	(empty)	_____
"cat"	"The"	_____
"sat"	"The cat"	_____

Task 3: Mathematical pattern
The pattern you discovered can be written as:
New Memory = Function(Current Word, Previous Memory)
Or mathematically: $h_t = f(x_t, h_{t-1})$
What are the inputs to this function?

- Input 1: _____
- Input 2: _____

What is the output? _____

Discovery Moment

Congratulations! You've just invented the core idea of RNNs - using hidden states (h_t) to maintain memory of previous inputs!

Part 3: When Memory Fails - The Vanishing Gradient Problem (10 minutes)

Long-Distance Dependencies

Task 1: The forgetting problem

Try to complete this sentence: "The keys that I left on the table in the kitchen yesterday before going to work _____ missing."

What word did you need to remember? _____

How many words back was it? _____

Task 2: Memory decay simulation

Imagine your memory fades by 10% at each word. Starting with strength 1.0:

Step	Calculation	Memory Strength
Start	1.0	1.0
After 1 word	1.0×0.9	0.9
After 2 words	0.9×0.9	_____
After 5 words	0.9^5	_____
After 10 words	0.9^{10}	_____
After 20 words	0.9^{20}	_____

Task 3: The problem

At what point does memory become effectively zero (¡ 0.01)? _____

This is why RNNs can't handle long sequences - the gradient (learning signal) vanishes!

Think About It

If memory decays exponentially, how can we preserve important information for longer?

Part 4: The Gate Solution - Selective Memory (10 minutes)

Designing Gates

Scenario: You're managing your email inbox. You have three actions:

- Delete (forget) spam
- Save (input) important emails
- Reply to (output) urgent emails

Task 1: Gate design
For the sentence "The cat that was black sat on the mat", decide what to do with each word:

Word	**Forget?** (0=forget all, 1=keep all)	**Save?** (0=ignore, 1=save)	**Use Now?** (0=hide, 1=use)
"The"	0.5	0.3	0.2
"cat"	_____	_____	_____
"that"	_____	_____	_____
"was"	_____	_____	_____
"black"	_____	_____	_____
"sat"	_____	_____	_____

Task 2: Gate benefits
How do gates help with the vanishing gradient problem?
If forget gate = 1.0, then memory decay = 1.0^{20} = _____
This creates a "highway" for gradients!
Task 3: Memory update with gates
New formula with gates:

$$\text{New Memory} = (\text{Forget Gate} \times \text{Old Memory}) + (\text{Input Gate} \times \text{New Info})$$

Write this mathematically using your notation:
_____

Discovery Moment

You've just invented LSTM! The three gates you designed (Forget, Input/Save, Output/Use) are exactly how LSTMs work!

Part 5: Comparing Memory Systems (10 minutes)

RNN vs LSTM vs GRU

Task 1: Complete the comparison
Based on your discoveries, fill in this table:

Aspect	**Simple RNN** (No gates)	**LSTM** (3 gates)	**GRU** (2 gates)
How it handles memory	Overwrites completely		Selective update
Maximum sequence length	10-20 words		100 words
Gradient flow	Exponential decay		Good preservation
Best for		Long documents	Quick training

Task 2: Application matching
Which architecture would you choose for:

1. Predicting the next character in a name: _____
2. Summarizing a 10-page document: _____
3. Real-time speech recognition on phone: _____
4. Analyzing sentiment in tweets: _____

Task 3: Design your own architecture
If you could add a fourth gate to LSTM, what would it do?
Gate name: _____
Function: _____
When would it activate (0 or 1)? _____

Synthesis Questions

1. **Why can't we just make the memory infinitely large?**

2. **The forget gate seems counterintuitive - why would we want to forget?**

3. **How is an LSTM like a computer's RAM?**

4. **Could we have more than 3 gates? What would be the trade-off?**

Key Discoveries:

• Order matters in language → need memory

• Hidden states maintain context → RNN

• Gradients vanish over time → long sequences fail

• Gates control information flow → LSTM solves vanishing gradients