Week 4: Sequence-to-Sequence Models

Post-Class Learning Verification

From Fixed-Length Prison to Variable-Length Freedom

NLP Course 2025 - Assessment

Time Required: 45-60 minutes
Purpose: Verify and deepen your understanding of seq2seq models after completing the lab
Format: Conceptual problems, implementation questions, and real-world applications

Checkpoint

Before Starting: You should have completed the Week 4 lab and understood:

- Why variable-length sequences need special handling
- The encoder-decoder architecture
- The information bottleneck problem
- How attention mechanism works
- Beam search for sequence generation

Part A: Conceptual Understanding (25 minutes)

A1: The Variable-Length Problem (5 minutes)

Understanding the Core Challenge

Question 1: Explain why traditional RNNs cannot handle translation tasks effectively.

Question 2: Consider these translation pairs. Circle the fundamental problem:

- English: "How are you?" (3 words) → Spanish: "¿Cómo estás?" (2 words)
- English: "Thank you" (2 words) → German: "Vielen Dank" (2 words)
- English: "Good morning" (2 words) → Japanese: "Ohayou gozaimasu" (2 words)

☐ Different word counts between languages

☐ RNNs can only produce one output per input

☐ No shared vocabulary between languages

☐ Grammar structures are too different

Question 3: List three real-world applications where variable-length input/output is essential:
1. ____________________________________
2. ____________________________________
3. ____________________________________

A2: Encoder-Decoder Architecture (8 minutes)

Understanding the Solution

Question 1: Draw and label the encoder-decoder architecture for translating "The cat sleeps" → "Le chat dort":

Draw your architecture here

Question 2: Complete the mathematical description:

- Encoder equations: $h_t =$ ____________________________
- Context vector: $c =$ ____________________________
- Decoder equations: $s_t =$ ____________________________
- Output probability: $P(y_t|\ldots) =$ ____________________________

Question 3: Why is the context vector a "bottleneck"?

Question 4: If your encoder has 128 hidden units and processes a 20-word sentence, what is the size of:

- Context vector: ________________ dimensions
- Total encoder information: ________________ dimensions
- Information compression ratio: ________________

A3: Attention Mechanism Deep Dive (8 minutes)

Attention Mathematics and Intuition

Question 1: Complete the attention mechanism steps:
Step 1: Compute alignment scores: $e_{t,i} =$ ______________________________
Step 2: Normalize with softmax: $\alpha_{t,i} =$ ______________________________
Step 3: Compute context vector: $c_t =$ ______________________________
Question 2: Given these simplified encoder states and decoder state:

- $h_1 = [0.5, 0.2]$ (word: "The")
- $h_2 = [0.8, 0.1]$ (word: "cat")
- $h_3 = [0.3, 0.7]$ (word: "sleeps")
- $s_t = [0.7, 0.3]$ (generating French word)

Calculate dot-product attention weights:

- $e_{t,1} = h_1 \cdot s_t =$ _______________
- $e_{t,2} = h_2 \cdot s_t =$ _______________
- $e_{t,3} = h_3 \cdot s_t =$ _______________

Which word gets highest attention? _______________
Question 3: Attention visualization interpretation. If you see this attention pattern:

French	**The**	**black**	**cat**	**sleeps**
Le	0.8	0.1	0.1	0.0
chat	0.1	0.1	0.8	0.0
noir	0.0	0.9	0.1	0.0
dort	0.0	0.0	0.1	0.9

What pattern do you observe? ____
Why is this good for translation? ____

A4: Beam Search Strategy (4 minutes)

Search and Generation

Question 1: Why can't we just use greedy search (always pick highest probability word)?

Question 2: Complete this beam search example (beam size = 2):
Starting with "START", expand to first word:

- P("Le" — START) = 0.6
- P("Un" — START) = 0.3
- P("La" — START) = 0.1

Keep top 2: ____________________ and ____________________
Expand "Le" to second word:

- P("chat" — Le) = 0.8 → Total: _______________
- P("chien" — Le) = 0.2 → Total: _______________

Expand "Un" to second word:

- P("chat" — Un) = 0.7 → Total: _______________
- P("chien" — Un) = 0.3 → Total: _______________

Final top 2 sequences: ______________________________ and ______________________________

Question 3: What happens to beam search quality vs. beam size?
Beam size 1: _________________________
Beam size 100: _________________________
Optimal beam size (typical): _______________

Part B: Implementation and Code Understanding (15 minutes)

B1: Code Analysis (8 minutes)

PyTorch Implementation

Question 1: Analyze this encoder code. Fill in the missing dimensions:

```
class Seq2SeqEncoder(nn.Module):
    def __init__(self, vocab_size=5000, embed_size=128, hidden_size=256):
        self.embedding = nn.Embedding(vocab_size, embed_size)
        self.lstm = nn.LSTM(embed_size, hidden_size, batch_first=True)

    def forward(self, x):  # x shape: [batch_size, seq_len]
        embedded = self.embedding(x)  # shape: [______, ______, ______]
        output, (h_n, c_n) = self.lstm(embedded)
        return h_n, c_n  # shapes: [______, ______], [______, ______]
```

Fill in the shapes:

- embedded shape: [__________, __________, __________]
- h_n shape: [__________, __________]
- c_n shape: [__________, __________]

Question 2: What's wrong with this attention implementation?

```
def attention(decoder_hidden, encoder_outputs):
    scores = torch.matmul(decoder_hidden, encoder_outputs.T)
    weights = F.softmax(scores, dim=1)
    context = torch.sum(weights * encoder_outputs, dim=1)
    return context
```

Problem: ____

Fix: ____

Question 3: Complete this beam search pseudocode:

```
def beam_search(model, input_seq, beam_size=4, max_length=20):
    beams = [{"sequence": [START_TOKEN], "score": 0.0}]

    for step in range(max_length):
        candidates = []
        for beam in beams:
            if beam["sequence"][-1] == END_TOKEN:
                candidates.append(beam)
                continue

            # Get next word probabilities
            probs = model.predict_next(beam["sequence"])

            # Expand beam
            for word, prob in probs.top_k(beam_size):
                new_score = ___________________
                new_sequence = ______________
                candidates.append({"sequence": new_sequence, "score": new_score})

        # Keep top beam_size candidates
        beams = sorted(candidates, key=lambda x: x["score"])[:______]
```

B2: Training Insights (7 minutes)

Training Process

Question 1: What is "teacher forcing" and why do we use it during training?
Definition: ____
Why use it: ____
What problem does it cause: ____
Question 2: Loss function analysis. Given these target and predicted sequences:
Target: ["Le", "chat", "dort", "¡EOS¿"] Predicted logits for each position:

- Position 1: P("Le")=0.8, P("Un")=0.15, P("La")=0.05
- Position 2: P("chat")=0.9, P("chien")=0.07, P("oiseau")=0.03
- Position 3: P("dort")=0.6, P("mange")=0.3, P("court")=0.1
- Position 4: P("¡EOS¿")=0.95, P("bien")=0.03, P("mal")=0.02

Calculate the cross-entropy loss:

- Position 1 loss: $-\log(0.8) =$ _______________
- Position 2 loss: $-\log(0.9) =$ _______________
- Position 3 loss: $-\log(0.6) =$ _______________
- Position 4 loss: $-\log(0.95) =$ _______________
- Total loss: _______________

Question 3: Why might attention weights look random at the start of training?

What should happen to attention weights as training progresses?

Part C: Real-World Applications (12 minutes)

C1: Modern Applications Analysis (6 minutes)

2024 Industry Applications

Question 1: Map these modern systems to seq2seq components:

Application	Input	Output
Google Translate		
GitHub Copilot		
Email Summarization		
Text-to-SQL		
Chatbot Response		

Question 2: Which of these would benefit most from attention mechanism?

- ☐ Translating short phrases (3-5 words)
- ☐ Translating technical documents (500+ words)
- ☐ Generating code from one-line comments
- ☐ Summarizing research papers
- ☐ Converting speech to text

Explain your top choice: ____

Question 3: Compare 2014 vs 2024 seq2seq capabilities:

Aspect	2014 (Original)	2024 (Modern)
Max input length		
Translation quality		
Inference speed		
Model size		
Applications		

C2: System Design Challenge (6 minutes)

Building Real Systems

Question 1: Design a meeting summarization system:
Input: 2-hour meeting transcript (5000 words) **Output:** Key points and action items (200 words)
Your architecture:

- Preprocessing: ____________________________________
- Encoder design: ____________________________________
- Attention strategy: ____________________________________
- Decoder design: ____________________________________
- Post-processing: ____________________________________

Question 2: Code comment to function generator:
Input: "Function to calculate fibonacci numbers up to n" **Output:** Complete Python function
What challenges would this face?

1. ____________________________________
2. ____________________________________
3. ____________________________________

How would attention help? ____________________________________
Question 3: Multilingual customer support system:
Requirements:

- Input: Customer query in any of 10 languages
- Output: Response in same language as input
- Must handle domain-specific terminology

Design approach:

- Language detection: ____________________________
- Translation pipeline: ____________________________
- Response generation: ____________________________
- Quality assurance: ____________________________

Part D: Critical Thinking and Extensions (8 minutes)

D1: Problem Solving (4 minutes)

Common Pitfall

Real challenges you might encounter in production:

Debugging and Optimization

Scenario 1: Your seq2seq translator produces repetitive output: "The cat the cat the cat sleeps"
Possible causes:

- ☐ Attention weights are too uniform
- ☐ Beam search beam size too small
- ☐ Training data has repetitive patterns
- ☐ Learning rate too high
- ☐ Decoder getting stuck in local optima

Best solution: ____
Scenario 2: Translation quality drops drastically for sentences longer than 20 words.
Root cause: ____
Solution strategy: ____
Scenario 3: Your model works great on formal text but fails on casual social media language.
Why this happens: ____
How to fix: ____

D2: Future Connections (4 minutes)

Think About It

Connecting to Advanced Topics:
Question 1: How do Transformers (next week) improve on seq2seq?
Key improvements:

1. ______________________________________
2. ______________________________________
3. ______________________________________

Question 2: Modern large language models (ChatGPT, GPT-4) use modified seq2seq principles. What's different?
Architectural changes: ______________________________________
Scale differences: ______________________________________
Training approach: ______________________________________
Question 3: Beyond text, what other sequence-to-sequence problems exist?

- Audio domain: ______________________________
- Video domain: ______________________________
- Scientific domain: ______________________________
- Creative domain: ______________________________

Self-Assessment and Next Steps

Checkpoint

Check your understanding level:

- ☐ I can explain why seq2seq was needed (vs fixed-length RNNs)
- ☐ I understand the encoder-decoder architecture completely
- ☐ I can describe the information bottleneck problem
- ☐ I can explain how attention solves the bottleneck
- ☐ I can implement attention mechanism from scratch
- ☐ I understand beam search vs greedy search trade-offs
- ☐ I can design seq2seq systems for real applications
- ☐ I see the connection to modern transformer models

Real World Application

Industry Relevance Check: Rate your confidence (1-5) for these career-relevant skills:

- Building translation systems: __________
- Designing text summarization: __________
- Code generation tools: __________
- Conversational AI systems: __________
- Understanding modern LLM architecture: __________

Areas needing review: ____
Most interesting discovery: ____
Connection to your projects/interests: ____

Next Steps

Immediate review: Focus on areas where you scored below 3/5

Hands-on practice:

- Implement seq2seq from scratch in your preferred framework
- Try the model on a different language pair
- Experiment with different attention mechanisms
- Build a simple summarization system

Preparation for Week 5:

- Review attention mechanism thoroughly
- Understand the limitations of RNN-based seq2seq
- Think about parallelization challenges
- Read "Attention Is All You Need" paper introduction

— End of Assessment —

You're now ready to understand how Transformers revolutionized this field!