Week 4: Teaching Machines to Translate

Discovery-Based Learning Exercises (Student Version)

Learning Objectives

By the end of this session, you will:

- Discover why variable-length input/output is challenging
- Design your own encoder-decoder architecture
- Identify the information bottleneck problem
- Invent the attention mechanism

---

1 Part 1: The Translation Challenge (10 minutes)

1.1 Warm-up: Word-by-Word Translation

Q: Let's try translating English to French word-by-word:

English	French (word-by-word)
The cat sat on the mat	Le chat ___ sur le tapis
How are you?	Comment ___ ___?
I love natural language processing	Je ___ naturel langue traitement

Q: What problems do you notice with word-by-word translation?

1.2 The Length Mismatch Problem

Exercise: Count the words in these equivalent sentences:

- English: "I love you" = ___ words
- French: "Je t'aime" = ___ words
- German: "Ich liebe dich" = ___ words
- Japanese (romanized): "Aishiteru" = ___ word(s)

Think: If a neural network produces one output per input, how can it handle these different lengths?

1.3 Design Challenge

Q: You're building a translation system. Your input is a sequence of English words, your output needs to be French words. The lengths don't match. How would YOU solve this?

Draw your solution here:

Discovery

If you thought of processing the entire input first before generating output, you're thinking like a sequence-to-sequence model designer!

2 Part 2: Building the Bridge (15 minutes)

2.1 The Compression Exercise

Exercise: Compress these sentences into exactly 3 numbers, then try to reconstruct them:

1. "The cat sat" → [___, ___, ___]
2. "A dog ran quickly" → [___, ___, ___]
3. "The International Conference on Machine Learning accepted our paper about neural networks" → [___, ___, ___]

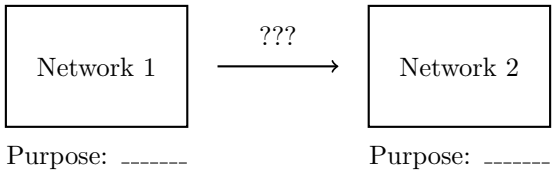
Q: Which sentence was hardest to compress? Why?

Q: Which information did you lose in sentence 3?

2.2 The Two-Network Solution

Think: What if we had TWO separate networks - one to read/understand, one to write/generate?

Fill in this diagram with what each network should do:



Q: What should pass between the two networks (what goes in the ??? box)?

Discovery

Congratulations! You just invented the encoder-decoder architecture! Network 1 = Encoder (compresses input to understanding) Network 2 = Decoder (generates output from understanding)

3 Part 3: The Bottleneck Discovery (10 minutes)

3.1 Long Sentence Challenge

Exercise: Try to translate this paragraph using your two-network system:

"The International Conference on Machine Learning, which is one of the premier venues for presenting research in machine learning and attracts submissions from researchers around the world working on various aspects of learning algorithms, accepted our paper about using neural networks for natural language understanding, specifically focusing on how attention mechanisms can improve translation quality."

Q: If you compress this entire paragraph into a fixed-size vector (say, 256 numbers), what information might you lose?

- Beginning: _____
- Middle: _____
- End: _____

Think: This is like trying to remember an entire book by storing it as a single "feeling" - you'll forget the details!

3.2 Information Theory

Q: Calculate the information loss:

- Short sentence (5 words) compressed to 256 numbers: Minimal loss
- Long document (500 words) compressed to 256 numbers: _____ loss

Q: What's the fundamental problem here?

Discovery

You've identified the information bottleneck! Fixed-size representations can't capture all the information from arbitrarily long sequences.

4 Part 4: Inventing Attention (15 minutes)

4.1 Human Translation Process

Exercise: Translate this sentence step by step, marking which English words you look at for each French word:

English: "The black cat sat on the mat"

Generating French word	Looking at English words
"Le"	_____
"chat"	_____
"noir"	_____
"s'est assis"	_____
"sur"	_____
"le"	_____
"tapis"	_____

Think: Notice how you don't look at ALL words equally - you focus on relevant parts!

4.2 Designing the Looking-Back Mechanism

Q: Instead of compressing everything into one vector, what if the decoder could "look back" at all encoder states? Design a mechanism:

1. How would you decide which encoder states are relevant?

2. How would you combine multiple relevant states?

3. How would you turn this into weights that sum to 1?

4.3 Computing Attention Scores

Exercise: For the word "chat" (cat), assign relevance scores (0-1) to each English word:

English word	Relevance to "chat"
The	___
black	___
cat	___
sat	___
on	___
the	___
mat	___
Total	Should sum to 1.0

Discovery

You just invented attention! Your relevance scores are attention weights, and looking back at all encoder states based on relevance is exactly how attention works!

5 Part 5: Putting It All Together (10 minutes)

5.1 Complete Architecture

Draw the complete sequence-to-sequence model with attention:

5.2 Key Components Checklist

Check off each component you included:

- ☐ Encoder network (processes input)
- ☐ Multiple encoder hidden states (not just final)
- ☐ Decoder network (generates output)
- ☐ Attention mechanism (looks back at encoder states)
- ☐ Attention weights (relevance scores)
- ☐ Context vector (weighted sum)

5.3 Reflection Questions

Q: Why is attention better than a fixed-size bottleneck?

Q: What tasks besides translation could benefit from this architecture?

Q: What limitations might this approach still have?

6 Bonus Challenge: Beam Search

Think: When generating translations, should we always pick the most likely next word?

Exercise: Consider translating "bank" - it could mean financial institution or river bank. Design a strategy to explore multiple translation paths:

Summary

Today you discovered:

1. The variable-length challenge in translation
2. The encoder-decoder architecture
3. The information bottleneck problem
4. The attention mechanism

These concepts you "invented" are the foundation of modern machine translation systems like Google Translate!

Next Week: We'll see how taking attention to the extreme (attention is ALL you need) leads to Transformers!