Week 4: Teaching Machines to Translate

Discovery-Based Learning Exercises (Instructor Version with Solutions)

Learning Objectives

By the end of this session, students will:

- Discover why variable-length input/output is challenging
- Design their 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 s'est assis sur le tapis
How are you?	Comment ∅ es-tu?
I love natural language processing	Je aime naturel langue traitement

Solution

Word-by-word translation fails because:

- Different word orders: French uses "es-tu" (are-you) vs English "are you"
- Different number of words: "sat" becomes "s'est assis" (two words)
- Missing words: French often omits certain translations
- Grammar differences: "love" should be "aime" not "amour"

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

Solution

Students should identify:

- Word order differences between languages
- One word can map to multiple words (or vice versa)
- Grammatical structure differences
- Context is lost (e.g., ”bank” - financial vs river)
- Idiomatic expressions don't translate literally

1.2 The Length Mismatch Problem

Exercise: Count the words in these equivalent sentences:

- English: "I love you" = 3 words
- French: "Je t'aime" = 2 words
- German: "Ich liebe dich" = 3 words
- Japanese (romanized): "Aishiteru" = 1 word

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

Solution

This exercise demonstrates that:

- Same meaning requires different numbers of words across languages
- A fixed 1-to-1 mapping won't work
- We need a flexible architecture that can:
  - Read any number of input words
  - Generate any number of output words
  - The two numbers don't need to match!

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?

Solution

Guide students toward these insights:

- **Common student ideas:**
  - "Read all words first, then translate" - Good! This leads to encoder-decoder
  - "Use a buffer/memory" - Excellent intuition about context vectors
  - "Process chunks at a time" - This hints at attention windows
- **Key insight to draw out:** We need to separate reading (understanding) from writing (generation)
- **Diagram solution:**
  - Input: English words → Reader/Encoder
  - Middle: Fixed-size "understanding" vector
  - Output: Writer/Decoder → French words

Discovery

If students thought of processing the entire input first before generating output, they'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:

Solution

Example compressions (students' will vary):

1. "The cat sat" → [0.2, 0.8, -0.5] (animal, past action, simple)
2. "A dog ran quickly" → [0.3, 0.9, 0.7] (animal, past action, with modifier)
3. "The International Conference on Machine Learning accepted our paper about neural networks" → [0.9, 0.4, 0.6] (academic, past action, complex)

Key teaching points:

- Short sentences compress well
- Long sentences lose massive amounts of information
- The fixed size (3 numbers) is the bottleneck

Q: Which sentence was hardest to compress? Why?

Solution

Sentence 3 is hardest because:

- Much longer (13 words vs 3-4)
- More semantic content (conference name, paper topic, action)
- Specific details (which conference? what about neural networks?)
- 3 numbers can't capture all this information

Q: Which information did you lose in sentence 3?

Solution

Lost information includes:

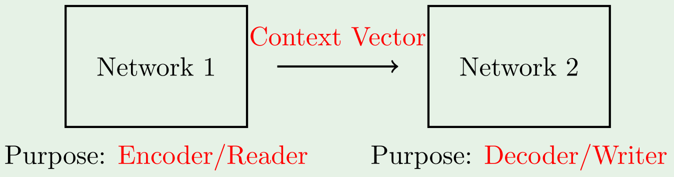
- Specific conference name (just becomes "academic")
- "our" - ownership information
- "neural networks" - specific topic
- Word order and relationships
- Basically everything except general gist!

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:

Solution



Network 1 (Encoder): Reads and understands the input sequence
Network 2 (Decoder): Generates the output sequence
Connection: Fixed-size context vector containing the "meaning"

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

Solution

The context vector should contain:

- Semantic meaning of the input
- Abstract representation (not word-specific)
- Fixed size (e.g., 256 or 512 dimensions)
- Think of it as the "understanding" or "gist"

Discovery

Congratulations! Students 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?

Solution

Information loss by position:

- Beginning: Conference name, "premier venue" detail
- Middle: "researchers around the world", "various aspects"
- End: "attention mechanisms", "translation quality" specifics
- Overall: Lose sequential structure, specific relationships between clauses

This demonstrates that a fixed-size bottleneck loses information proportional to input length!

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:

Solution

- Short sentence (5 words) compressed to 256 numbers: Minimal loss
  - Ratio: 256/5 = 51.2 numbers per word (plenty!)
- Long document (500 words) compressed to 256 numbers: Massive loss
  - Ratio: 256/500 = 0.512 numbers per word (not enough!)

Q: What's the fundamental problem here?

Solution

The fundamental problem is the **information bottleneck**:

- Fixed-size representation regardless of input length
- Information capacity doesn't scale with input size
- Longer sequences = more compression = more loss
- This is why early seq2seq models failed on long sentences!

Discovery

Students have 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"

Solution

Generating French word	**Looking at English words**
"Le"	The
"chat"	cat
"noir"	black
"s'est assis"	sat
"sur"	on
"le"	the (second one)
"tapis"	mat

Key insight: We don't look at all words equally - we focus on relevant parts for each output word!

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:

Solution

1. How would you decide which encoder states are relevant?
   - Compare current decoder state with each encoder state
   - Use similarity (dot product or learned function)
   - Higher similarity = more relevant
2. How would you combine multiple relevant states?
   - Weighted average based on relevance scores
   - More relevant states get higher weights
3. How would you turn this into weights that sum to 1?
   - Apply softmax to similarity scores
   - This gives probability distribution over encoder states

Congratulations! Students just invented the attention mechanism!

4.3 Computing Attention Scores

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

Solution

English word	**Relevance to "chat"**
The	0.05
black	0.10
cat	0.70
sat	0.05
on	0.03
the	0.02
mat	0.05
Total	1.00

Teaching points:

- "cat" gets highest weight (0.70) - direct translation
- "black" gets some weight (0.10) - modifier that might influence
- Other words get minimal weight - not directly relevant
- Weights sum to 1.0 (probability distribution)

Discovery

Students just invented attention! The 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:

Solution

Key components students should include:

1. **Encoder:**
   - Processes input words sequentially
   - Produces hidden state for each word
   - Keeps ALL hidden states (not just final)
2. **Attention Mechanism:**
   - Compares decoder state to all encoder states
   - Computes attention weights (softmax of similarities)
   - Creates context vector (weighted sum)
3. **Decoder:**
   - Generates output words one at a time
   - Uses both its own state AND attention context
   - Can "look back" at relevant input parts

5.2 Key Components Checklist

Check off each component you included:

Solution

All boxes should be checked:

✓ 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?

Solution

Attention solves the bottleneck problem by:

- No information compression - keeps all encoder states
- Dynamic information access - decoder chooses what's relevant
- Scales with input length - longer inputs have more states to attend to
- Preserves fine details - can look back at specific words
- Handles long-range dependencies - can attend to distant words

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

Solution

Students might suggest:

- Text summarization (attend to important parts)
- Question answering (attend to relevant context)
- Image captioning (attend to image regions)
- Speech recognition (attend to audio segments)
- Code generation (attend to specifications)
- Any task mapping sequences to sequences!

Q: What limitations might this approach still have?

Solution

Limitations to discuss:

- Still sequential processing (slow)
- Attention computation scales $O(n^2)$ with sequence length
- RNN-based encoders still have gradient problems
- Fixed attention mechanism (not learnable enough)
- These limitations lead to Transformers (Week 5)!

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:

Solution

Beam Search strategy:

- Keep top-k translation hypotheses at each step (e.g., k=3)
- For “bank”:
  - Path 1: “banque” (financial) with probability 0.6
  - Path 2: “rive” (river bank) with probability 0.3
  - Path 3: “bord” (edge) with probability 0.1
- Expand each path, keep top-k overall
- Final: Choose highest probability complete sentence
- This avoids getting stuck with early bad choices!

Teaching Notes

6.1 Timing Guidelines

- Part 1: 10 minutes - Focus on length mismatch problem
- Part 2: 15 minutes - Let students struggle with compression
- Part 3: 10 minutes - Make bottleneck problem visceral
- Part 4: 15 minutes - Guide toward attention naturally
- Part 5: 10 minutes - Consolidate understanding

6.2 Common Student Misconceptions

1. **“Just use a bigger vector”** - Explain that any fixed size eventually fails
2. **“Store all words separately”** - Good intuition! This leads toward attention
3. **“Use multiple networks”** - Interesting, but coordination becomes complex
4. **“Attention looks at future words”** - Clarify it only looks at input (encoder) states

6.3 Key Pedagogical Moments

- When students realize word-by-word fails - validate this discovery
- During compression exercise - let them experience information loss
- When inventing attention - connect to how humans translate
- Final assembly - celebrate that they invented a real system!

Summary

Today students discovered:

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

These concepts they “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!