Week 5: Transformers - The Architecture Behind ChatGPT

Simplified BSc Handout with Visual Learning

Today's Journey

Imagine you're in a classroom where everyone can talk to everyone else at the same time, instead of passing notes one by one. That's the transformer revolution!

---

1 Part 1: Why Transformers? The Problem with Sequential Processing

1.1 The Telephone Game Problem

Activity

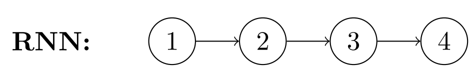
Play telephone with this sentence: "The quick brown fox jumps over the lazy dog"
Person 1 → Person 2 → Person 3 → ... → Person 10
What happens to the message?

This is how RNNs work - information degrades as it passes through the network sequentially.

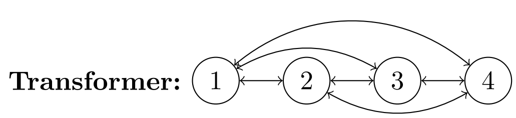
1.2 The Parallel Processing Solution

Real World Example

RNN (Sequential): Like reading a book one word at a time with your finger **Transformer (Parallel)**: Like seeing the whole page at once



Must wait for each step



All connections at once!

Checkpoint

Key Insight: Transformers process all words simultaneously, making them much faster on modern hardware!

2 Part 2: Self-Attention - The Core Innovation

2.1 The Attention Mechanism Explained Simply

Think of attention like a **spotlight in a theater**:

- Each word is an actor on stage
- Each actor has a spotlight they control
- They can shine their spotlight on other actors (or themselves)
- The brightness shows how much they're paying attention

2.2 The Three Roles: Query, Key, Value

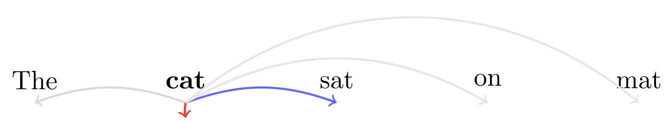
Real World Example

Imagine a library:

- **Query**: "I'm looking for books about cats"
- **Key**: Each book's catalog card
- **Value**: The actual book content

You compare your query to all keys, then take the values of matching books!

2.3 Visual Example: How "cat" Attends to Other Words



Thicker arrow = More attention

Activity

For the sentence "The student loves pizza", draw attention arrows from "loves" to each word. Which words should get the most attention?

3 Part 3: Multi-Head Attention - Different Perspectives

3.1 Why Multiple Heads?

Real World Example

Like having multiple cameras filming a scene:

- Camera 1: Focuses on the main actor (subject)
- Camera 2: Captures the action (verb)
- Camera 3: Shows the setting (context)
- Camera 4: Tracks emotions (sentiment)

Each "head" captures different relationships!

3.2 Visual: 4 Heads Looking at "bank"

Head	**Focus**	**Attends to**
Head 1	Syntax	"The" (determiner)
Head 2	Meaning	"river" (context)
Head 3	Position	nearby words
Head 4	Topic	"water", "flow"

Checkpoint

Each head learns to look for different patterns. Together, they understand language from multiple angles!

4 Part 4: Positional Encoding - Teaching Order

4.1 The Position Problem

Without position information:

- "Cat chased mouse" = same as "Mouse chased cat"
- "John loves Mary" = same as "Mary loves John"

Activity

Write two sentences using the same words but different order, where the meaning completely changes:
1. __________________________
2. __________________________

4.2 The Sine Wave Solution

Real World Example

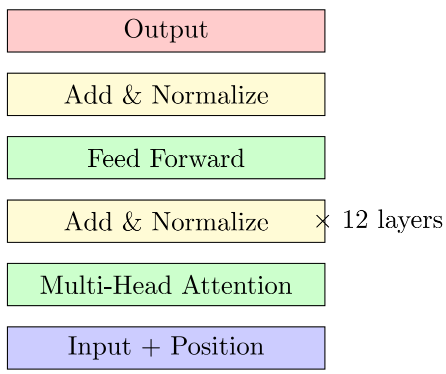
Like giving each word a unique "address":

- Position 1: [0.84, 0.54, 0.00, 1.00, ...]
- Position 2: [0.91, 0.42, 0.84, 0.54, ...]
- Position 3: [0.14, -0.99, 0.91, 0.42, ...]

Each position has a unique pattern, like a fingerprint!

5 Part 5: Building a Complete Transformer

5.1 The Layer Cake Architecture



Checkpoint

Like a layer cake, each layer adds more understanding. GPT-3 has 96 layers!

5.2 Residual Connections - Information Highways

Real World Example

Like having both stairs AND an elevator in a building:

- Stairs = Going through the transformation
- Elevator = Direct connection (residual)

Information can take either path!

6 Part 6: Hands-On Code Understanding

6.1 Simplified Attention in Python

```python
# Simplified self-attention (conceptual)
def attention(sentence):
    words = sentence.split()
    attention_scores = {}

```

```
    for word1 in words:
        scores_for_word1 = {}
        for word2 in words:
            # How relevant is word2 to word1?
            score = calculate_relevance(word1, word2)
            scores_for_word1[word2] = score

        # Normalize scores to sum to 1
        total = sum(scores_for_word1.values())
        for word2 in scores_for_word1:
            scores_for_word1[word2] /= total

        attention_scores[word1] = scores_for_word1

    return attention_scores

# Example usage
result = attention("The cat sat")
# result["cat"] might be: {"The": 0.2, "cat": 0.3, "sat": 0.5}
```

Activity

Trace through this code with "I love pizza". What would result["love"] contain?

7 Part 7: Real-World Applications

7.1 Transformers Everywhere!

Application	**Model**	**What it Does**
Chat	ChatGPT	Conversations
Translation	Google Translate	100+ languages
Code	GitHub Copilot	Writes code
Images	DALL-E	Creates pictures
Science	AlphaFold	Protein folding

Real World Example

When you use autocomplete on your phone, that's a tiny transformer running locally!

8 Part 8: Practice Problems

8.1 Problem 1: Attention Weights

Given the sentence "Dogs love treats", fill in reasonable attention weights:

From ↓ To →	Dogs	love	treats
Dogs	0.5	___	___
love	___	0.2	___
treats	___	___	0.4

Remember: Each row must sum to 1.0!

8.2 Problem 2: Parallelization

Calculate the speedup:

- Sentence length: 50 words

- RNN: Processes 1 word per time step
- Transformer: Processes all words at once

Time for RNN: ____ steps Time for Transformer: ____ step(s) Speedup: ____ times faster

8.3 Problem 3: Design Challenge

Design a 2-head attention system for understanding "Time flies like an arrow":

- Head 1 focuses on: _____________
- Head 2 focuses on: _____________

9 Key Takeaways

Checkpoint

Remember these 5 key points:

1. Transformers process all words **in parallel**
2. Self-attention creates **direct connections** between all words
3. Multiple heads capture **different relationships**
4. Position encoding tells the model about **word order**
5. This architecture powers **ChatGPT** and modern AI!

10 Bonus: Fun Facts

- The transformer paper has been cited over 100,000 times!
- GPT-3 would take 355 years to train on a single GPU
- Transformers can work with images, music, and even DNA sequences
- The name comes from "transforming" one sequence to another
- The original transformer was trained for translation in just 3.5 days

---

Next Steps

1. Try the Jupyter notebook to build your own transformer
2. Experiment with attention visualizations
3. Read "Attention Is All You Need" paper (challenge yourself!)

Real World Example

You now understand the architecture that powers ChatGPT, BERT, and almost every modern NLP system. That's huge!