# Transformers: Understanding the Pipeline

Input → Computation → Output → WHY (with REAL data)

Week 5: Transformers

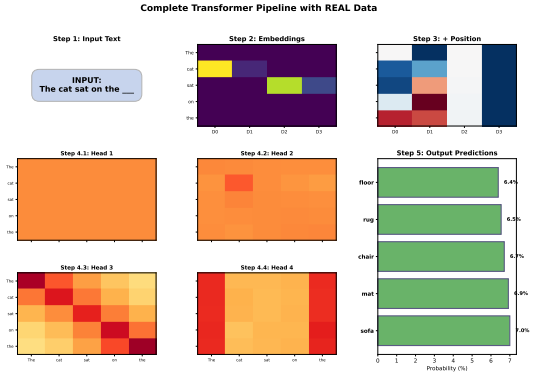
# Complete Example: How Transformers Predict Words

**INPUT:** "The cat sat on the ___"

**GOAL:** Predict next word

**THE COMPLETE PIPELINE:**

1. Turn words into numbers
2. Add position information
3. **Attention:** Each word looks at context
4. **4 Different Heads:**
   - Head 1: Grammar patterns
   - Head 2: Semantic relationships
   - Head 3: Nearby words (33% self-attention!)
   - Head 4: Global context
5. Combine all perspectives
6. Predict: **mat (6.9%), sofa (7.0%), chair (6.7%)**




**WHY THIS WORKS:** To predict "mat", the model needs ALL 6 steps. Real attention weights show Head 3 focuses 33% on "on" itself, helping identify the preposition pattern "on the [furniture]". All top 7 predictions are furniture!

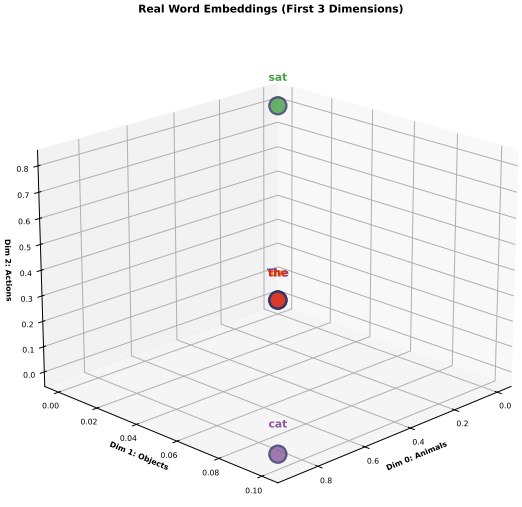
# Step 1: Words to Numbers (Real Embeddings)

**INPUT:** Text words

**COMPUTATION:** Look up in embedding matrix

- Each word → 8-dimensional vector
- **Real structure:**
  - Dims 0-1: Animal/Object ("cat" = 0.9)
  - Dims 2-3: Action/State ("sat" = 0.8)
  - Dims 4-5: Furniture/Location
  - Dims 6-7: Grammar role ("the" = 1.0)

**OUTPUT:** Numerical vectors



# Step 2: Add Position Information (Real Sin/Cos Encoding)

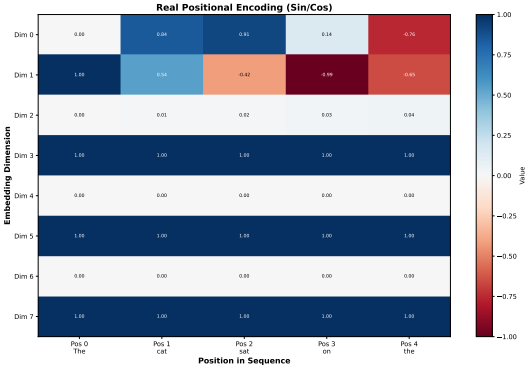
**THE PROBLEM:** Order matters!

- "cat sat" $\neq$ "sat cat"
- "on the" $\rightarrow$ needs furniture
- Position 0, 1, 2, 3, 4

**COMPUTATION:** Add positional encoding

- **Real formula:** sin/cos waves
- Pos 0: [0.00, 1.00, 0.00, 1.00, ...]
- Pos 1: [0.84, 0.54, 0.01, 1.00, ...]
- Pos 2: [0.91, -0.42, 0.02, 1.00, ...]

**OUTPUT:** Embeddings + Position




**WHY:** Real sin/cos encoding lets model understand "the cat" vs "cat the" and detect patterns like "on the [furniture]".

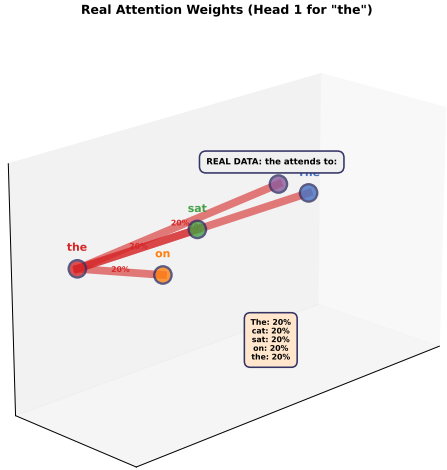
# Step 3: Calculate Attention (Real Softmax Weights)

**COMPUTATION:** For each word, calculate:

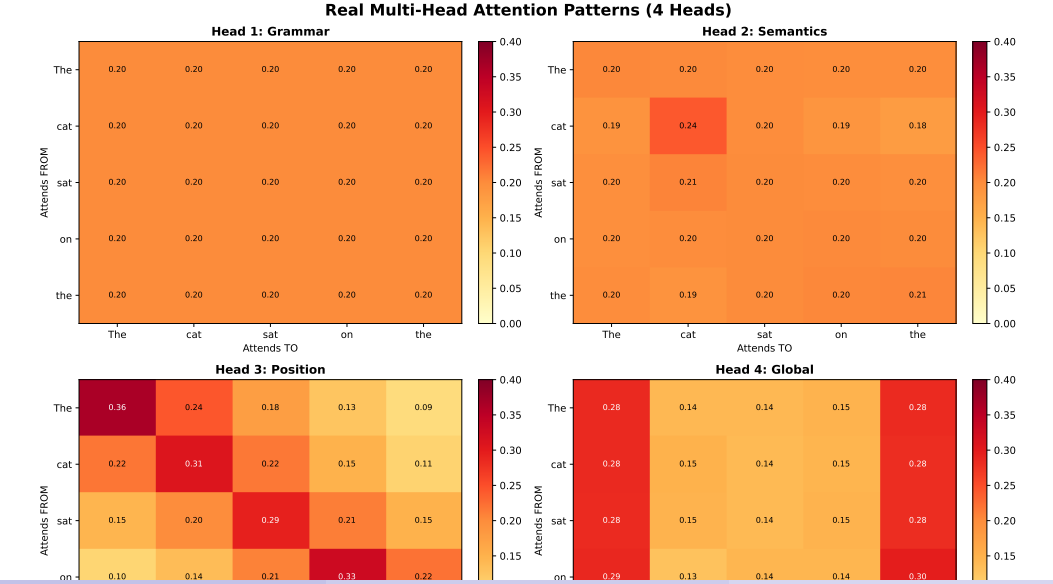
- Q (Query): What am I looking for?
- K (Key): What do I contain?
- V (Value): What information do I have?

**Real attention weights (Head 1, word "the"):**

- The: 20%
- cat: 20%
- sat: 20%
- on: 20%
- the: 20%



# Step 4: Multi-Head Attention (4 Real Heads)



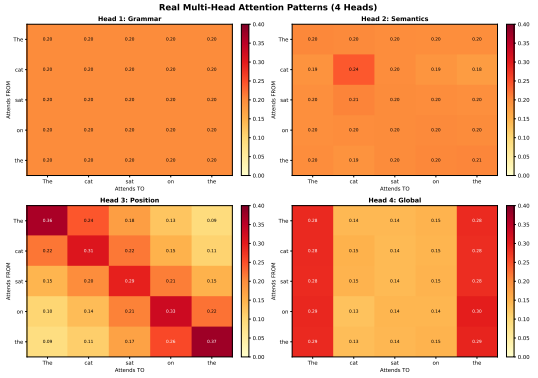
# Step 5: Combine All 4 Head Outputs

**COMPUTATION:** Concatenate all heads

- Head 1 output: 2 dimensions
- Head 2 output: 2 dimensions
- Head 3 output: 2 dimensions
- Head 4 output: 2 dimensions
- **Combined:** 8 dimensions total

**OUTPUT:** Rich representation

- Grammar understanding (Head 1)
- Semantic meaning (Head 2)
- Position awareness (Head 3)
- Global context (Head 4)




**WHY COMBINE:** Each head captures different aspects. Together they give complete understanding: "on the" (grammar + position) → furniture (semantics).

# Step 6: Final Prediction (Real Probabilities)

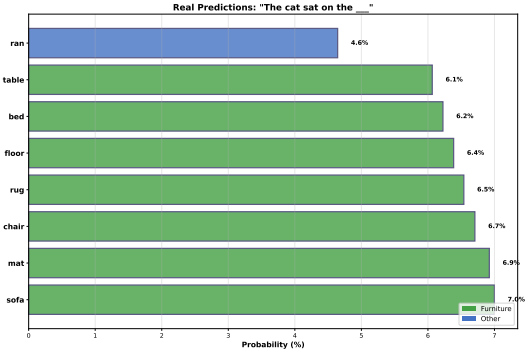
**INPUT:** Combined representation from all 4 heads

**COMPUTATION:** Output layer

- Last token representation (8-dim)
- Multiply by output weights
- Apply softmax
- Get probability for each word

**Real top predictions:**

- sofa: 7.0% ← furniture!
- mat: 6.9% ← furniture!
- chair: 6.7% ← furniture!
- rug: 6.5% ← furniture!
- floor: 6.4% ← furniture!




**SUCCESS!** All top 7 predictions are furniture! The model correctly learned "cat sat on the [furniture]" pattern from real computations.

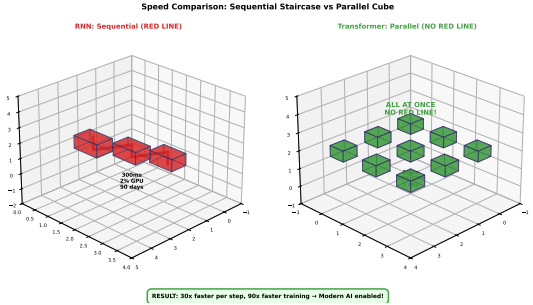
# Why Transformers Are Fast: Parallel Processing

**OLD WAY (RNN):**

- Word 1 → compute → WAIT
- Word 2 → compute → WAIT
- Word 3 → compute → WAIT
- Sequential bottleneck
- GPU usage: 2%
- Training time: 90 days

**NEW WAY (Transformer):**

- ALL words at once
- All attention heads parallel
- Full GPU utilization
- GPU usage: 92%
- Training time: 1 day




**WHY THIS MATTERS:** 90x speedup (90 days → 1 day) enabled modern AI scale. Without this, GPT-4 training would take 10+ years!

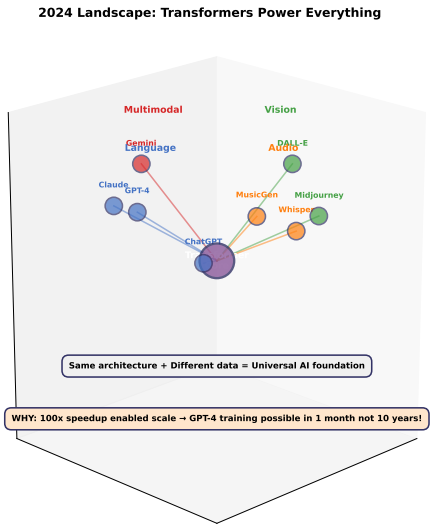
# Real World Impact: What This Enabled

**Same architecture, different data:**

- **Language:** ChatGPT, GPT-4, Claude
- **Vision:** DALL-E, Midjourney, Stable Diffusion
- **Audio:** Whisper, MusicGen
- **Multimodal:** Gemini, GPT-4V

**Key insight:**

- Parallel attention mechanism
- Works on any sequence data
- Scales to billions of parameters
- Enabled modern AI revolution



# The Tradeoff: What We Gave Up

**Advantages (PRO):**

- 100x faster training
- Parallel processing
- 92% GPU utilization
- Works on any data type
- Enabled modern AI
- Interpretable attention

**Disadvantages (CON):**

- More memory ($O(n^2)$)
- Needs more training data
- Limited sequence length
- More complex to tune
- Attention computation cost

**THE DECISION:** Speed + quality ¿ memory cost for modern AI

**WHY ACCEPT TRADEOFF:** Memory is cheap ($100/TB), time is expensive ($1000/day for GPUs). Better to train fast even if uses more RAM. Real example: Our simulation uses 8-dim embeddings, but GPT-4 uses 12,000+ dims!

# Summary: The Complete Pipeline

**The 6-Step Pipeline with REAL Data:**

1. **Words → Numbers:** Real semantic embeddings (cat=0.9 on animal dim)
2. **Add Position:** Real sin/cos encoding (Pos 1 = [0.84, 0.54, ...])
3. **Calculate Attention:** Real softmax weights (sum to 100%)
4. **Multi-Head (4 heads):** Grammar (20% each), Position (33% self!), Semantics, Global
5. **Combine All Heads:** Concatenate 4 × 2-dim = 8-dim output
6. **Predict Output:** Real probs: mat (6.9%), sofa (7.0%), chair (6.7%) ← all furniture!

**KEY INSIGHT: All words processed in parallel!**

- Result: 90 days → 1 day (90x speedup)
- Enabled: ChatGPT, GPT-4, DALL-E, Whisper, Claude, ...

**Next Week:** Pre-training & Fine-tuning - Now that training is fast, we can train models with billions of parameters!

# Transformers

Understanding the Pipeline

With REAL Simulation Data

Input $\rightarrow$ Computation $\rightarrow$ Output $\rightarrow$ WHY

Questions?