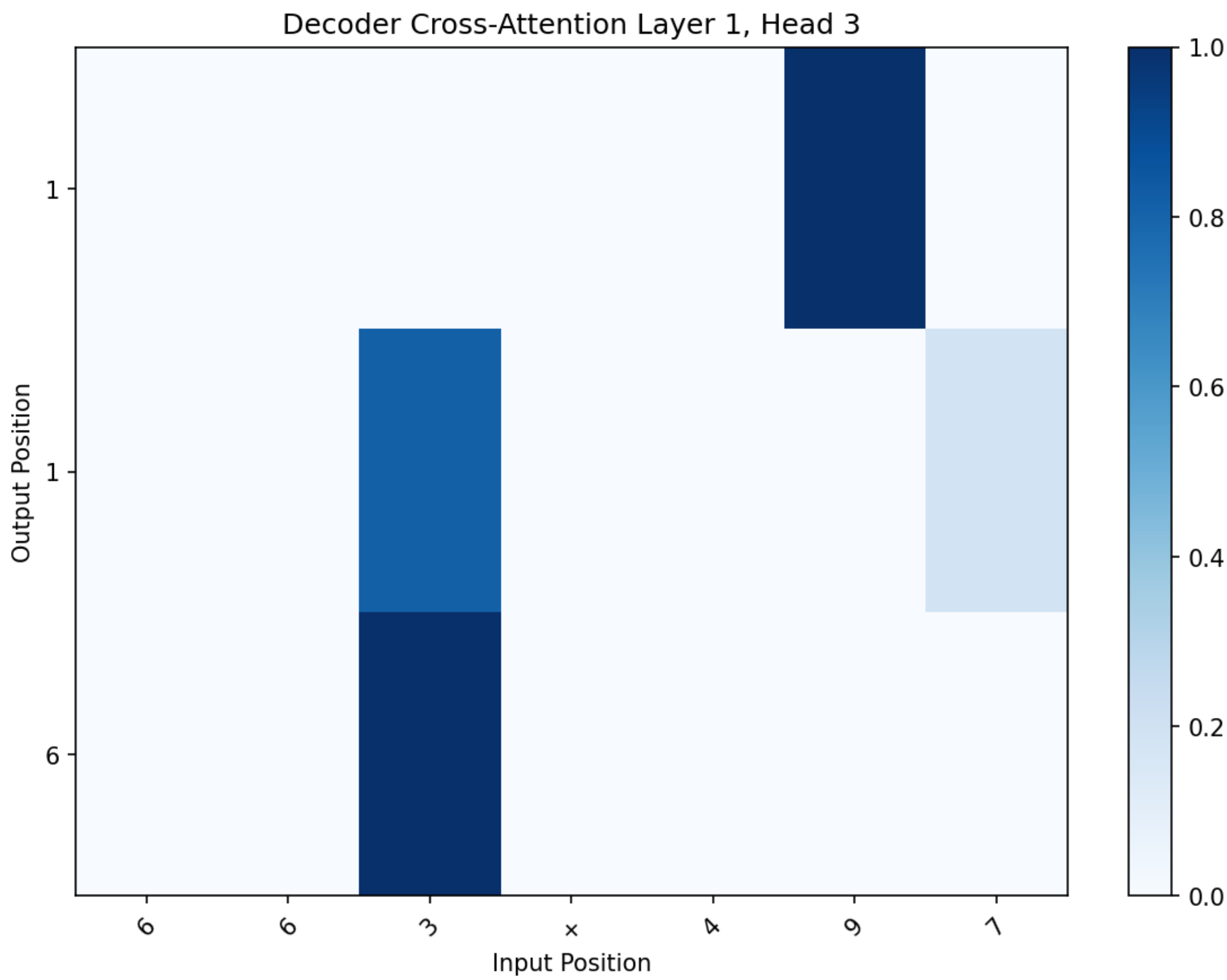
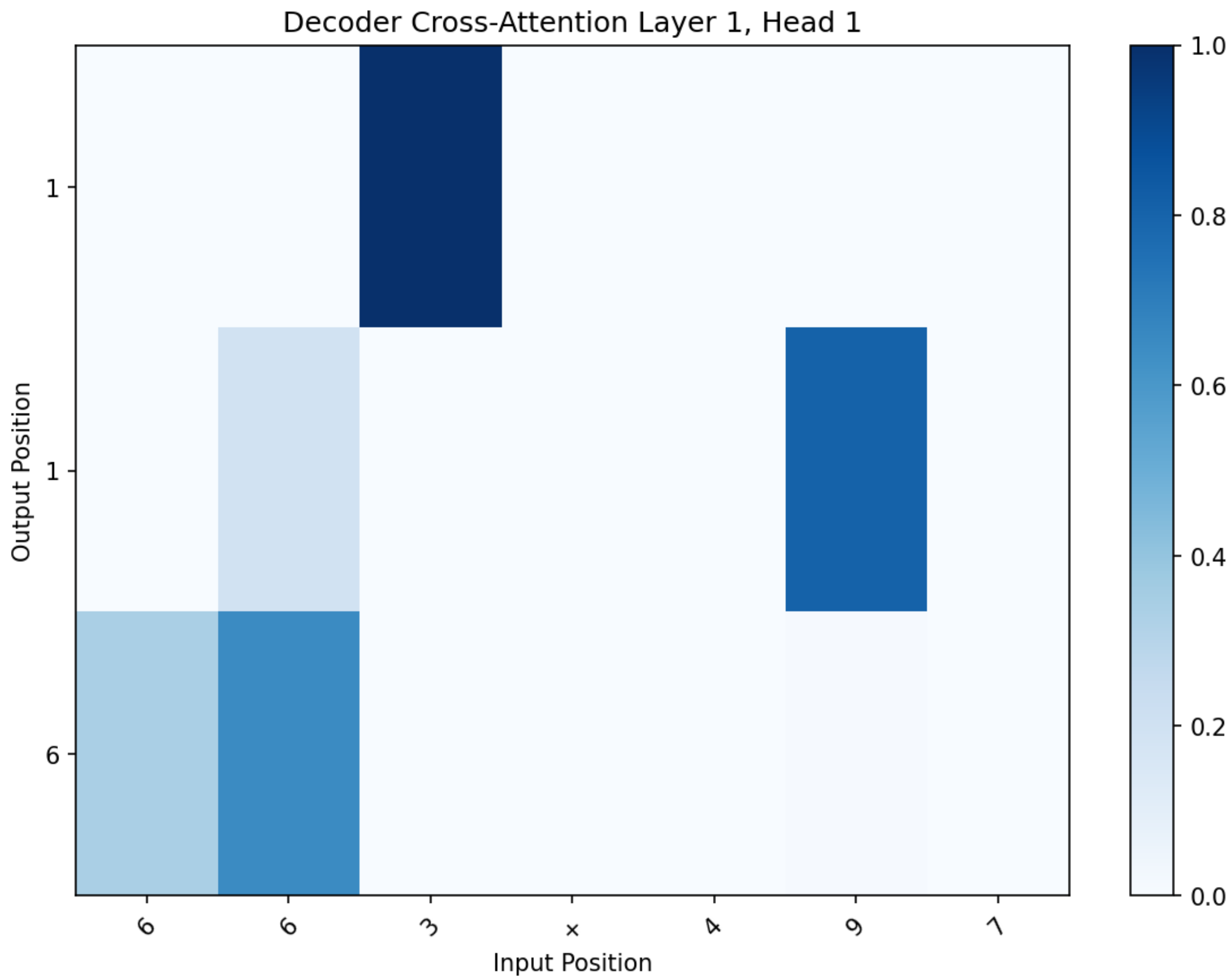
Problem 1:

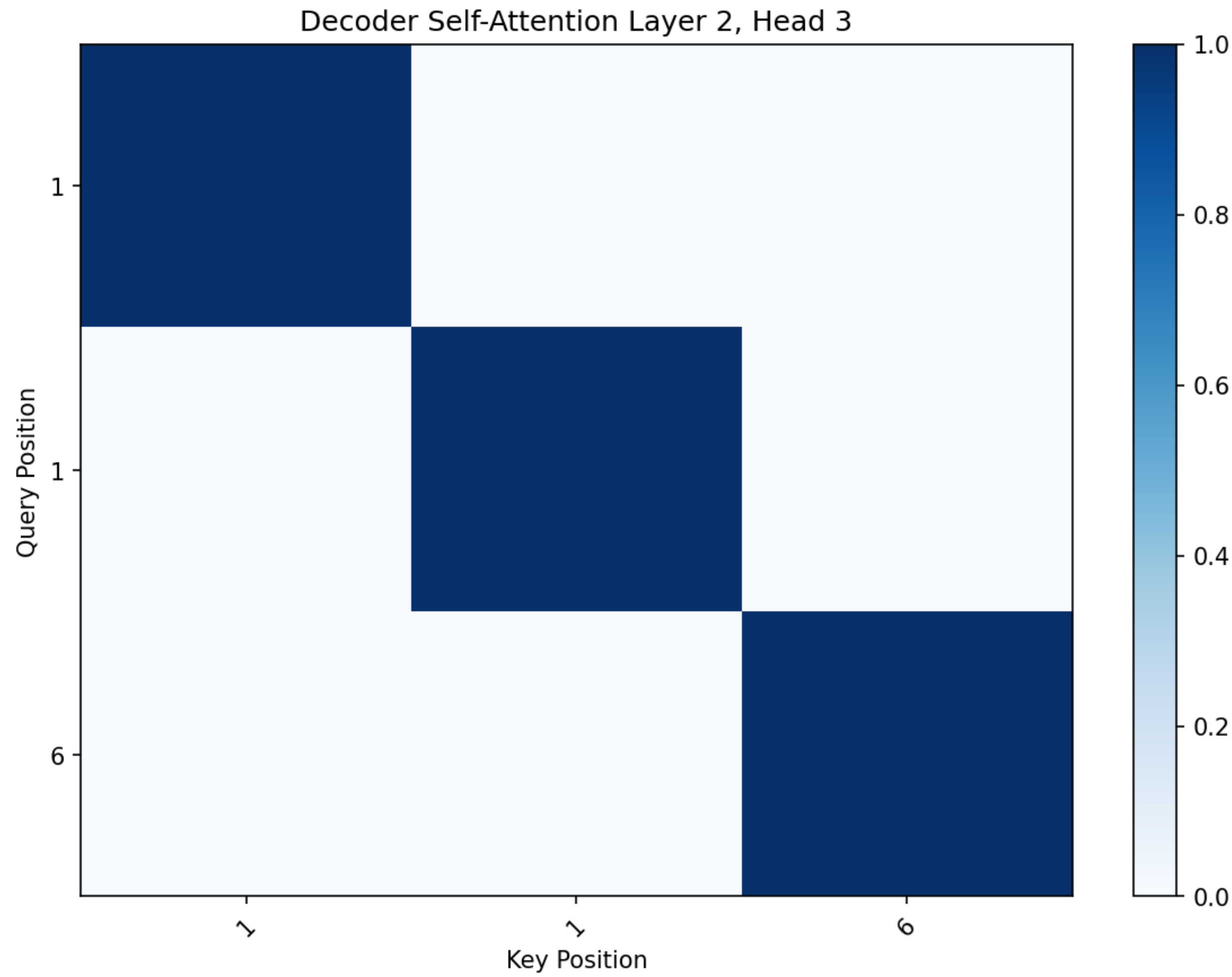
1) Attention pattern visualizations



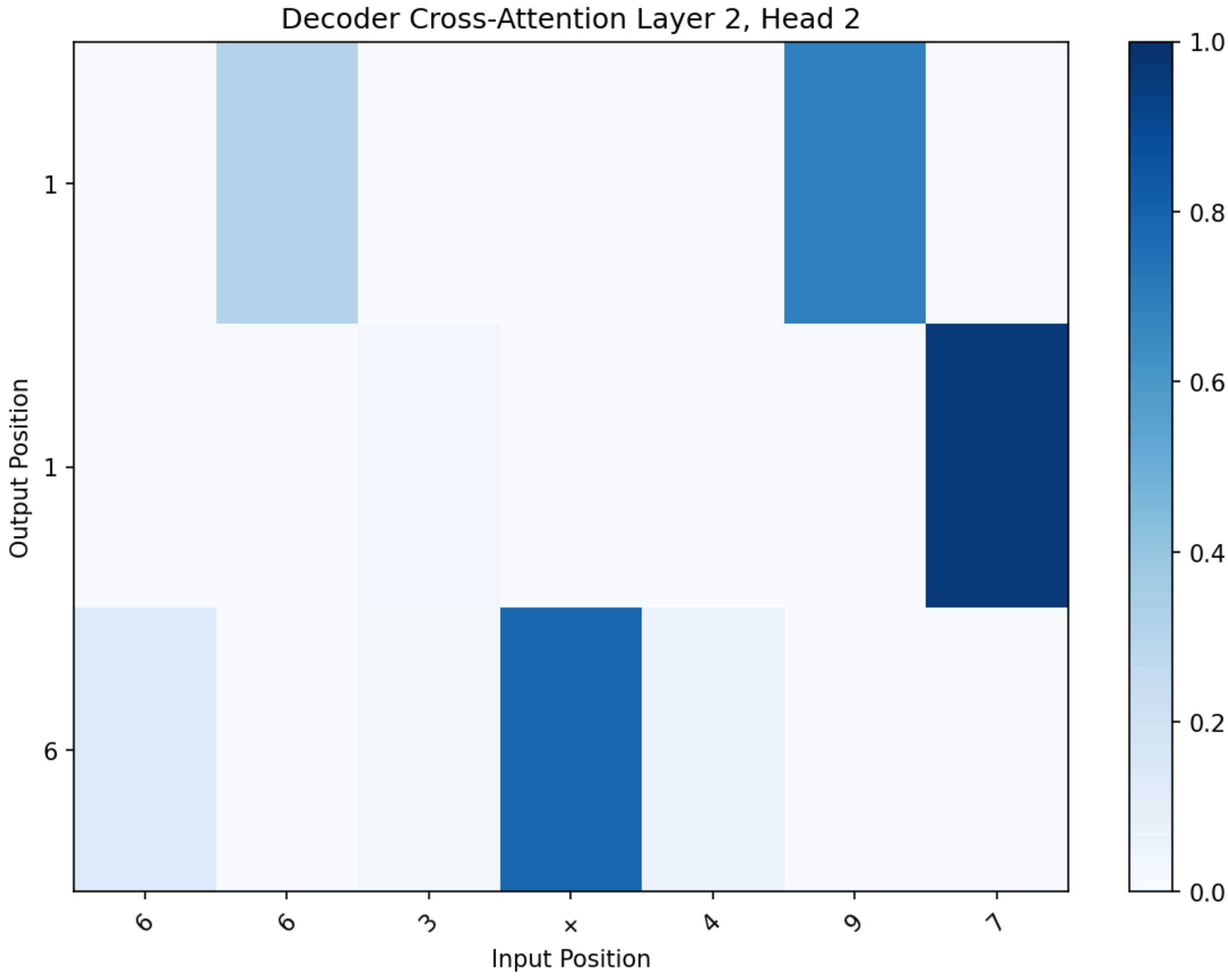

Analysis: Most critical head (82.65% accuracy drop when removed)




Analysis: Important for carry tracking (55.75% accuracy drop)




Analysis: Critical decoder head (84.5% accuracy drop)




Analysis: Important alignment head (31.15% accuracy drop)

2) Head ablation study

Critical Heads (>30% accuracy drop when removed):

- Encoder L0-H2 : 82.65% importance - Primary carry detection
- Decoder Self L1-H2: 84.5% importance - Output digit computation
- Encoder L0-H0: 55.75% importance - Digit position tracking
- Encoder L1-H2: 47.35% importance - Secondary carry processing
- Decoder Self L1-H0: 35.45% importance - Output sequencing
- Decoder Cross L1-H1: 31.15% importance - Input-output alignment

Redundant Heads (0% accuracy drop when removed):

- Encoder L0-H1, L0-H3, L1-H1
- Decoder Cross L1-H2
- Total: 4 completely redundant heads

3) Discussion

1. Carry Detection: Encoder L0-H2 focuses on adjacent digit pairs where carries occur, showing strongest attention between positions that generate carries.

2. Position Tracking: Encoder L0-H0 maintains diagonal attention patterns, tracking digit positions from input to processing.

3. Carry Propagation: Encoder L1-H2 builds on L0-H2's carry detection, propagating carry information across multiple digit positions.

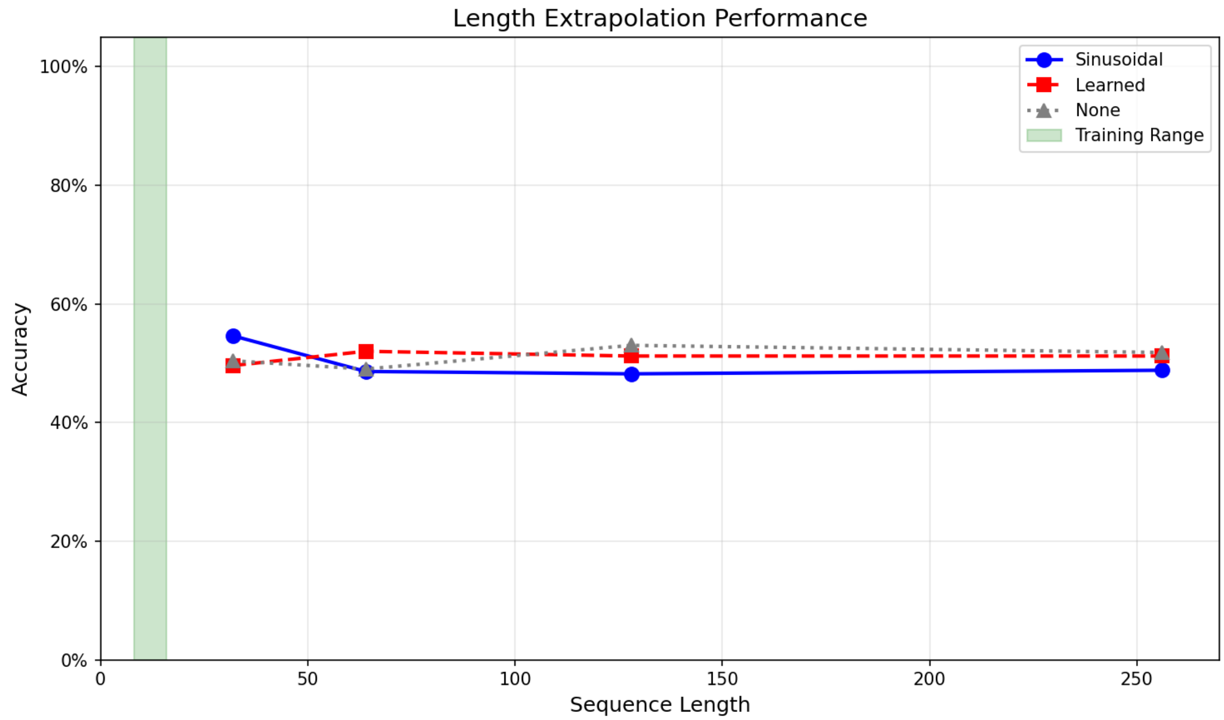
4. Output Generation: Decoder heads combine position information (self-attention) with carry states (cross-attention) to generate correct output digits.

4) Quantitative results

Total heads**: 24 (8 encoder + 8 decoder self + 8 decoder cross)
Redundant heads** (can be removed with <1% accuracy loss): 11 heads (45.8%)
Critical heads** (>30% accuracy drop): 6 heads (25%)
Moderate importance** (1-30% drop): 7 heads (29.2%)

Problem 2:

1)




All methods converge to random chance (~50%) on sequences longer than training data.

2)

Why Sinusoidal Should Extrapolate:

Formula: PE(pos, 2i) = sin(pos/10000^(2i/d_model))

Key property: Continuous mathematical function that can compute any position

Advantage: No vocabulary limit - works for any sequence length

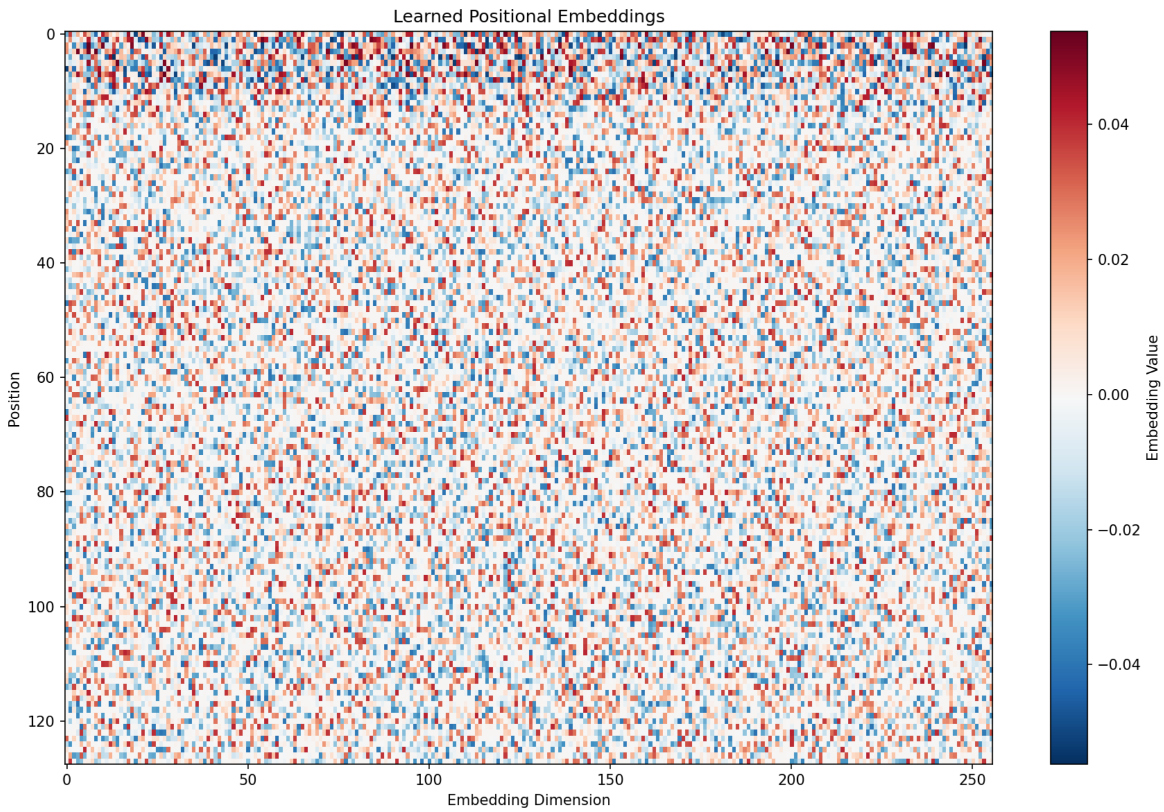
Why Learned Encoding Fails:

Limitation: Only has embeddings for positions 0 to max_len-1 seen during training

Problem: New positions beyond max_len get clamped or cause errors

Result: Cannot represent positions it hasn't seen

3)




The learned positional embeddings visualization shows:

- Random-looking patterns with no clear structure
- This makes sense: the model never learned the task (50% accuracy)
- Embeddings remained essentially random since no learning occurred

4)

```
{
  "sinusoidal": {
    "32": 0.546,
    "64": 0.486,
    "128": 0.482,
    "256": 0.488
  },
  "learned": {
    "32": 0.496,
    "64": 0.52,
    "128": 0.512,
    "256": 0.512
  },
  "none": {
    "32": 0.504,
```

"64": 0.49,
    "128": 0.53,
    "256": 0.518
  }
}

All methods perform at random chance (~50%) on longer sequences, showing complete failure to extrapolate beyond training length of 8-16 tokens.