GPT Memorization Capacity Scaling Experiment

Model Architecture Sequence

Base Configuration

- **Architecture**: GPT-2 style decoder-only transformer
- **Sequence Length**: 64 tokens (for consistent comparison)
- **Vocabulary Size**: 2 (binary: 0, 1)
- **Position Embeddings**: Learned
- **Activation**: GELU
- **Normalization**: Layer norm

Model Scale Progression

Model	Layers	d_model	Heads	Head_dim	Parameters	Scale Factor
Nano	2	32	2	16	~8K	1x
Micro	4	64	4	16	~67K	8x
Mini	6	128	8	16	~370K	46x
Small	8	256	16	16	~2.1M	260x
Base	12	512	32	16	~15M	1875x
Large	16	768	48	16	~46M	5750x

Experimental Protocol

Data Generation

```python
def generate_random_binary_dataset(n_sequences, seq_length=64):
    """Generate truly random binary sequences"""
    return np.random.randint(0, 2, size=(n_sequences, seq_length))
```

Simplified Validation Experiment

Objective

Confirm Morris et al.'s linear scaling relationship (≈3.6 bits-per-parameter) holds across our model range using random binary sequences.

Model Architecture Sequence

Model	Layers	d_model	Heads	Parameters
Nano	2	32	2	~8K
Micro	4	64	4	~67K
Mini	6	128	8	~370K
Small	8	256	16	~2.1M

Experimental Protocol

Data: Random binary sequences (64 tokens, vocab=2) **Dataset sizes**: Exponential progression [1, 2, 4, 8, 16, 32, 64, 128, 256, 512] **Seeds**: 3 per model size **Measurement**: Morris memorization = $H(X) - H(X|\theta)$

Expected Results

- **Linear relationship**: Capacity ≈ 3.6 × Parameters
- **Plateau behavior**: Memorization plateaus at capacity limit
- **Cross-validation**: BPP approximately constant across model sizes

Success Criteria

1. **R^2 > 0.95** for linear fit between capacity and parameters
2. **BPP coefficient** within 20% of Morris et al.'s 3.6 value
3. **Consistent plateaus** observable for each model size

This streamlined approach validates the core scaling relationship without unnecessary precision in boundary detection.

Training Configuration

Optimization Settings

```python
```

```
training_config = {
    'optimizer': 'AdamW',
    'learning_rate': 1e-4,
    'weight_decay': 0.01,
    'gradient_clipping': 1.0,
    'batch_size': min(32, dataset_size),  # Adaptive batch size
    'warmup_steps': 100,
    'scheduler': 'cosine_with_restarts'
}
```

Convergence Criteria

- **Loss threshold**: 0.01 (following Morris protocol)
- **Morris threshold**: 95% efficiency for "successful" memorization
- **Patience**: 10,000 steps without improvement
- **Maximum steps**: 500,000 (generous upper bound)
- **Early stopping**: When both loss and Morris criteria met

Memorization Metrics

1. Traditional Loss-Based

python

```python
def memorization_achieved(loss, threshold=0.01):
    return loss < threshold
```

2. Morris Framework

python

```python
def morris_memorization(model, dataset):
    """H(X) - H(X | θ)"""
    theoretical_entropy = len(dataset) * seq_length * 1.0  # 1 bit per token
    model_entropy = calculate_cross_entropy_bits(model, dataset)
    return theoretical_entropy - model_entropy

def morris_efficiency(morris_bits, theoretical_max):
    return morris_bits / theoretical_max
```

3. Bits-Per-Parameter

```python
def bits_per_parameter(morris_memorization, model_params):
    return morris_memorization / model_params
```

Expected Scaling Relationships

Primary Hypothesis: Linear Scaling (Morris et al.)

Based on Morris et al. findings, GPT-family models have approximately constant bits-per-parameter capacity:

```
Capacity = k × Parameters + b
BPP ≈ constant ≈ k across model sizes
```

Key Questions to Investigate

1. **Coefficient Validation**: What is the empirical value of k (bits-per-parameter)?
2. **Architecture Sensitivity**: Does k vary with depth/width trade-offs?
3. **Scale Invariance**: Does linear relationship hold from 8K to 46M parameters?
4. **Offset Effects**: Is there a meaningful intercept b (fixed overhead)?

Data Collection

Primary Measurements

- **Memorization Capacity**: Maximum sequences memorized
- **Morris BPP**: Bits per parameter at capacity
- **Training Efficiency**: Steps to convergence
- **Scaling Exponent**: Power law fit across sizes

Secondary Measurements

- **Loss curves**: Full training dynamics
- **Parameter utilization**: Weight magnitude distributions
- **Interpolation capacity**: Performance between train sequences

Statistical Analysis

Linear Scaling Validation

```python
# Validate: Capacity = k * Parameters + b
# Focus on estimating k (bits-per-parameter coefficient)

def fit_linear_scaling(model_params, max_memorized_sequences):
    # Linear fit with confidence intervals
    slope, intercept, r_value, p_value, std_err = stats.linregress(
        model_params, max_memorized_sequences
    )
    return slope, intercept, r_value**2  # slope = k (BPP)
```

Key Validation Questions

- **Linearity**: How well does linear fit explain variance (R^2)?
- **BPP Constancy**: Is k consistent across the 8K→46M parameter range?
- **Architectural Effects**: Do different layer/width ratios affect k?
- **Intercept Significance**: Is there meaningful fixed capacity overhead b?

Experimental Controls

Randomization Controls

- **Fixed seeds** for reproducibility
- **Independent datasets** for each measurement
- **Randomized model initialization** across trials

Training Controls

- **Consistent optimization** settings across sizes
- **Proportional training budgets** based on model size
- **Hardware normalization** (same GPU type/memory)

Measurement Controls

- **Identical evaluation** protocols
- **Consistent thresholds** across all models
- **Multiple validation** datasets at capacity boundary

Expected Deliverables

1. **Linear Scaling Validation**: Confirm Morris et al. linear relationship holds across our model range
2. **Empirical BPP Coefficient**: Precise measurement of bits-per-parameter constant for binary sequences
3. **Architectural Sensitivity**: Whether depth vs. width affects memorization efficiency within linear scaling
4. **Scale Range Validation**: Confirm linearity holds from small (8K) to medium (46M) parameter models
5. **Baseline for Comparison**: Reference values for comparing with other data types (text, structured data)

The goal is validating and precisely measuring the linear scaling relationship rather than discovering unknown scaling behavior.