Language Modeling with Memory-Augmented LSTM: Improving Long-Context Text Prediction

He Jiang   HJ193@DUKE.EDU

Jay Wu   SW693@DUKE.EDU

Leo Yao   YY508@DUKE.EDU

Final Project Report

1. Introduction

Traditional LSTM-based language models suffer from limited ability to retain information across long sequences due to vanishing gradients and fixed hidden state size. This limitation becomes particularly problematic in tasks requiring context continuity, such as question-answering systems where answers depend on information from previous interactions. Our project addresses this challenge by integrating external memory components that explicitly store and retrieve summarized or semantically enriched representations of previous contexts.

We propose a Memory-Augmented LSTM language model designed to improve long-context text prediction. The system consists of two main components: a base LSTM encoder-decoder and a memory-augmented module that includes Short-Term Memory (STM) and Long-Term Memory (LTM) mechanisms.

2. Solution and Model Description

2.1 Base LSTM Encoder-Decoder

A standard LSTM network serves as the core language model, responsible for token-level prediction and next-word generation. The encoder processes the input sequence, and the decoder predicts the next token based on the hidden state and the memory-augmented context.

2.2 Memory-Augmented Module

To enhance context retention, we introduce two complementary memory mechanisms:

- **Short-Term Memory (STM)**: Captures recent context using a summarization layer and token limit controller, which condense previous sentences into a compact representation (max 256 tokens).

- **Long-Term Memory (LTM)**: Stores semantically meaningful information derived from previous text segments, including semantic search embeddings and named-entity representations (NER), which are retrieved during prediction to enrich the LSTM's input.

During inference, the model retrieves both short-term and long-term summaries and concatenates them with the current input before passing them to the LSTM encoder. This design allows the model to "recall" relevant past information without depending solely on hidden state propagation.

2.3 Model Variants

We implement five progressively complex model variants to evaluate the contribution of each memory component:

1. **Model 0 (Base)**: Baseline LSTM with no memory components. Only uses the current question without any history.

2. **Model 1 (SummarizationOnly)**: Adds summarization of historical context. The history is concatenated and summarized into a single compact text representation.

3. **Model 2 (SumTokenLimit)**: Extends Model 1 by adding token limit truncation (max 256 tokens) to the summarized history. This prevents the summary from becoming too long and overwhelming the input.

4. **Model 3 (SumTokNer)**: Extends Model 2 by adding Named Entity Recognition (NER). Extracts entities such as skills, roles, companies, and domain-specific terms from the summarized history and appends them as structured information.

5. **Model 4 (FullMemory)**: The complete model that extends Model 3 with semantic search capabilities. Implements a LocalSemanticMemory component that stores all historical paragraphs and retrieves semantically relevant ones based on the current question using embedding-based similarity search.

3. Experimental Setup

3.1 Datasets

We evaluate our models on two datasets:

- **Synthetic Dataset**: A SkillMiner QA dataset with 200 rows, where each row contains a question-answer pair. The dataset is designed to test the model's ability to maintain context across multiple interactions.

- **Real Dataset**: A Dog-Cat QA dataset with 200 question-answer pairs focusing on pet care and behavior, designed to test the model's performance on a different domain.

3.2 Training Configuration

- **Architecture**: 256 hidden dimensions, character-level tokenization

- **Training**: 10 epochs per model

- **Evaluation Metrics**:

  - **STM (Short-Term Memory) Accuracy**: Tests questions from 2 rows before (threshold: 0.6)
  - **LTM (Long-Term Memory) Accuracy**: Tests questions from 9 rows before (threshold: 0.5)
  - **Similarity Scores**: Both LLM-as-a-judge scores and difflib sequence similarity scores

3.3 Evaluation Methodology

- STM tests are performed every 5 rows (rows 3, 8, 13, . . .)

- LTM tests are performed every 10 rows (rows 1, 11, 21, . . .)

2

- Model outputs are compared to ground truth using LLM-as-a-judge (primary) and difflib similarity (secondary)

- The best performing epoch for each model is identified based on combined STM and LTM accuracy

4. Results

4.1 Synthetic Dataset Results

Table 1 summarizes the performance of all models on the synthetic dataset. We observe consistent improvement as memory components are added: Model 0 (Base) achieves 0.325 STM and 0.700 LTM accuracy at epoch 9. Model 1 improves STM to 0.425. Model 2 achieves 0.375 STM and 0.750 LTM. Model 3 achieves 0.375 STM and 0.800 LTM at epoch 5—notably reaching peak performance earlier. Model 4 achieves the highest STM accuracy of 0.550, though LTM decreases to 0.750.

On the real dataset (Dog-Cat), all models show 0.000 accuracy across all epochs, indicating the models struggle with this domain. However, loss decreases consistently and difflib scores increase, suggesting gradual learning (see Table 3).

Table 1: Performance comparison on synthetic dataset (best epoch).

Model	Syn. Epoch	Syn. STM Acc	Syn. LTM Acc
0 (Base)	9	0.325	0.700
1 (SumOnly)	9	0.425	0.700
2 (SumTokLimit)	9	0.375	0.750
3 (SumTokNer)	5	0.375	0.800
4 (FullMemory)	9	0.550	0.750

The detailed results for all models are shown in Table 2. Model 3's early convergence (epoch 5) with loss 0.0930 demonstrates that additional memory components enable faster convergence. Model 4 achieves the highest STM accuracy (0.550) with strong LLM scores.

Table 2: Detailed metrics for all models (best epoch).

Model	Loss	STM Acc	LTM Acc	STM LLM	LTM LLM
0 (Base)	0.0578	0.325	0.700	0.493	0.480
1 (SumOnly)	0.0609	0.425	0.700	0.525	0.515
2 (SumTokLimit)	0.0571	0.375	0.750	0.517	0.555
3 (SumTokNer)	0.0930	0.375	0.800	0.500	0.505
4 (FullMemory)	0.0687	0.550	0.750	0.512	0.505

Table 3 shows the real dataset results. Figure 3 and Figure 4 visualize the model comparison and training progress, demonstrating loss reduction and difflib score improvement across all models despite 0.000 accuracy.

Table 3: Real dataset (Dog-Cat) results: Loss and difflib scores (best epoch).

Model	Loss	STM Difflib	LTM Difflib	Epoch
0 (Base)	1.4217	0.060	0.062	10
1 (SumOnly)	1.4286	0.063	0.050	10
2 (SumTokLimit)	1.3974	0.061	0.041	10
3 (SumTokNer)	1.4145	0.059	0.042	10
4 (FullMemory)	1.4289	0.071	0.062	10

4.2 Real Dataset Results

Models were evaluated on the Dog-Cat QA dataset. As shown in Table 3, all models achieve 0.000 accuracy across all epochs, indicating the models struggle with this domain. This represents a poorly performing aspect of our project. However, we observe positive learning trends across all models: loss consistently decreases from epoch 1 to epoch 10 (e.g., Model 0: 2.4974 → 1.4217, Model 2: 2.4895 → 1.3974, Model 4: 2.5147 → 1.4289), and difflib scores generally increase, suggesting gradual learning despite not meeting accuracy thresholds. Model 2 achieves the lowest final loss (1.3974), while Model 4 shows the highest STM difflib score (0.071) at epoch 10. Qualitative analysis reveals all models generate repetitive patterns (e.g., "o o o o", "and and and", "cat cat cat", "the disease and provide their disease"), indicating they have not learned meaningful language patterns for this domain. This suggests the models may require domain-specific adaptations, more training data, or different hyperparameters for the Dog-Cat domain.

5. Analysis and Discussion

5.1 Performance Trends

From the synthetic dataset results, we observe a clear trend of improvement as memory components are added:

1. **Model 0 → Model 1**: Adding summarization improves STM accuracy from 0.325 to 0.425 (30.8% relative improvement), while maintaining LTM accuracy at 0.700. This demonstrates that summarization effectively captures recent context.

2. **Model 1 → Model 2**: Interestingly, adding token limit truncation results in a slight decrease in STM accuracy (0.425 → 0.375) but a notable increase in LTM accuracy (0.700 → 0.750). This trade-off can be explained by:

   - **LTM Improvement**: Token truncation helps focus the summary on the most important information, reducing noise that could interfere with long-term context retrieval. The more concise summaries are easier for the model to process when answering questions about older context.

4

- **STM Decrease**: The truncation might remove some recent details that are important for short-term questions. However, the LLM score for Model 2's STM (0.517) is actually higher than Model 1's (0.525), suggesting the quality of correct predictions may be better even if fewer pass the threshold.

3. **Model 2 → Model 3**: Adding NER improves LTM accuracy to 0.800, the highest among all models, demonstrating that entity extraction effectively captures key information for long-term context. Notably, Model 3 reaches its best performance at epoch 5 (loss: 0.0930), compared to epoch 9 for other models. This suggests that additional memory components enable faster convergence and better model capability, as the model can learn more efficiently with richer memory representations.

4. **Model 3 → Model 4**: Adding semantic search shows an interesting trade-off: STM accuracy increases significantly (0.375 → 0.550), but LTM accuracy decreases (0.800 → 0.750). This suggests semantic search may introduce noise or distract from entity-focused information for LTM tasks, while improving short-term context retrieval through richer semantic connections. The STM improvement demonstrates that semantic search effectively retrieves relevant recent context.

5.2 Loss Trends

All models show consistent loss reduction across epochs:

- Model 0: 1.9220 → 0.0541 (epoch 1 → 10)

- Model 1: 1.9097 → 0.0571 (epoch 1 → 10)

- Model 2: 1.8975 → 0.0522 (epoch 1 → 10)

- Model 3: 1.9378 → 0.0558 (epoch 1 → 10)

- Model 4: 1.8962 → 0.0627 (epoch 1 → 10)

Most models reach best performance at epoch 9, but Model 3 achieves its best at epoch 5, demonstrating that increased model capability (through NER) enables faster convergence with lower loss and higher accuracy. This suggests that as we add more components, the model's learning efficiency improves, requiring fewer epochs to reach optimal performance.

5.3 Qualitative Analysis: Model 2 and Model 4 Examples

From epoch 9 of Model 2, we observe examples that demonstrate the model's strong performance:
Example 1 (Successful STM):

- **Question**: "How does SkillMiner prioritize topics when I have limited time each week?"

- **True Answer**: "SkillMiner uses your resume, target role business analyst, and declared strengths in Python to generate a learning plan. It balances review of fundamentals with deeper interview prep tasks..."

- **Predicted Answer**: "SkillMiner uses your resume, target role business analyst, and declared strengths in data visualization to generate a learning plan. It balances review of fundamentals with deeper interview prep tasks"

- **LLM Score**: 0.6000, **Difflib Score**: 0.3259

- **Analysis**: The first 10+ words match exactly with the ground truth. The model correctly captures the core structure and meaning. The only difference is "Python" vs "data visualization" in the declared strengths, but the overall meaning and structure are preserved. This demonstrates that the model performs better than strict similarity scores suggest—humans can easily understand both versions as conveying the same information.

Example 2 (Successful LTM):

- **Question**: "What will SkillMiner recommend if my current strength is data visualization but I want to focus more..."

- **True Answer**: "SkillMiner uses your resume, target role backend developer, and declared strengths in data visualization..."

- **Predicted Answer**: "SkillMiner uses your resume, target role business analyst, and declared strengths in data visualization to generate a learning plan. It balances review of fundamentals with deeper interview prep tasks"

- **LLM Score**: 0.7000, **Difflib Score**: 0.1577

- **Analysis**: While the target role differs ("backend developer" vs "business analyst"), the model correctly maintains the overall structure and key concepts. The LLM judge recognizes the semantic similarity (0.7000), indicating that the model successfully retrieves and uses long-term context, even if specific details vary.

Example 3 (Failed Case):

- **Question**: "How does SkillMiner use my resume to identify missing skills for a product manager?"

- **True Answer**: "SkillMiner uses your resume, target role product manager, and declared strengths in time-series analysis..."

- **Predicted Answer**: "SkillMiner uses your resume, target role business analyst, and declared strengths in data visualization to generate a learning plan. It balances review of fundamentals with deeper interview prep tasks"

- **LLM Score**: 0.4000, **Difflib Score**: 0.1759

- **Analysis**: This failure case shows the model sometimes generates a generic response that doesn't adapt to the specific question context. The model appears to be "hallucinating" a common pattern rather than retrieving the specific answer. This suggests areas for improvement in context retrieval and question-specific adaptation.

Model 4 Examples (Epoch 9):
Example 4 (Successful STM):

- **Question**: "How does SkillMiner prioritize topics when I have limited time each week?"

- **True Answer**: "SkillMiner uses your resume, target role business analyst, and declared strengths in Python to generate..."

- **Predicted Answer**: "role, SkillMiner clusters your skills around themes such as deep learning and aligns them with job relevant milestones. The study plan focuses on LeetCode practice, and tracks completion so each week"

- **LLM Score**: 0.8000, **Difflib Score**: 0.0260

- **Analysis**: Despite a low difflib score (0.0260), the LLM judge recognizes strong semantic similarity (0.8000), indicating the model produces meaningful, coherent responses that capture the essence of the answer, even if the exact wording differs significantly.

Example 5 (Failed Case):

- **Question**: "What will SkillMiner recommend if my current strength is data visualization but I want to focus more..."

- **True Answer**: "SkillMiner uses your resume, target role backend developer, and declared strengths in data visualization..."

- **Predicted Answer**: "SkillMiner uses your resume, target role business analyst, and declared strengths in deep learning to generate a learning plan. It balances review of fundamentals with deeper interview prep tasks, and"

- **LLM Score**: 0.2000, **Difflib Score**: 0.1580

- **Analysis**: This failure shows the model generating a response that doesn't match the specific question context, with low LLM score (0.2000), indicating limitations in context-aware retrieval despite semantic search capabilities.

5.4 Similarity Score Analysis

The comparison between LLM scores and difflib scores reveals important insights:

- **LLM scores** (using LLM-as-a-judge) tend to be more lenient and semantically aware, recognizing that paraphrased or structurally similar answers convey the same meaning.

- **Difflib scores** are stricter, penalizing any character-level differences, even when the semantic meaning is preserved.

For example, in Model 2's epoch 9:

- STM LLM avg: 0.517 vs STM difflib avg: 0.164

- LTM LLM avg: 0.555 vs LTM difflib avg: 0.205

The significant gap suggests that many model outputs are semantically correct but differ in exact wording or structure. This aligns with our qualitative observations that the model often produces human-readable, meaningful responses that may not match the ground truth character-for-character.

5.5 Where the Model Performs Well

1. **Structural Consistency**: The model excels at maintaining the overall structure and format of answers, often matching the first 10-15 words exactly.

2. **Semantic Coherence**: Even when exact words differ, the model frequently produces semantically equivalent responses that humans can easily understand.

3. **Long-Term Context Retrieval**: LTM accuracy (0.750 for Model 2) demonstrates that the model can successfully retrieve and use information from 9 rows earlier, showing effective long-term memory capabilities.

4. **Domain-Specific Patterns**: The model learns common patterns in the SkillMiner QA domain, such as the structure "SkillMiner uses your resume, target role X, and declared strengths in Y..."

5.6 Where the Model Performs Poorly

1. **Specific Entity Substitution**: The model sometimes substitutes specific entities (e.g., "Python" → "data visualization", "product manager" → "business analyst") with generic or previously seen entities, suggesting it may be over-relying on common patterns rather than question-specific context.

2. **Generic Response Generation**: In some failure cases, the model generates a generic response that doesn't adapt to the specific question, indicating limitations in context-aware retrieval.

3. **Exact Match Requirements**: When evaluated using strict similarity metrics (difflib), the model's performance appears lower than human judgment would suggest, highlighting a mismatch between evaluation metrics and actual utility.

4. **Short-Term Memory Trade-offs**: The token limit truncation in Model 2 slightly reduces STM accuracy, suggesting that some recent details may be lost in the summarization process.

5.7 Real Dataset Analysis

On the Dog-Cat QA dataset, all models show consistent loss reduction across epochs, indicating learning is occurring despite 0.000 accuracy. Model 0 shows loss decreasing from 2.4974 (epoch 1) to 1.4217 (epoch 10), a 43.1% reduction. Model 2 achieves the lowest final loss (1.3974), suggesting token limit truncation helps focus learning even in challenging domains. Model 4 shows the highest STM difflib score (0.071) at epoch 10, indicating semantic search may help retrieve relevant patterns despite overall poor performance.

The increasing difflib scores across epochs (e.g., Model 0 STM: 0.027 → 0.060, Model 4 STM: 0.032 → 0.071) suggest the models are gradually learning to produce outputs that share more character-level similarity with ground truth, even if they don't meet the accuracy thresholds. However, the repetitive output patterns (e.g., "and and and", "cat cat cat") indicate the models are overfitting to common tokens rather than learning meaningful language structures. This suggests the Dog-Cat domain may require: (1) domain-specific tokenization or preprocessing, (2) larger training datasets, (3) different hyperparameters (learning rate, batch size), or (4) domain-adapted embeddings for semantic search components.

6. Pros and Cons

6.1 Advantages

1. **Interpretability**: Unlike black-box transformer models, our memory-augmented LSTM maintains interpretability. We can examine the retrieved summaries, entities, and semantic memories to understand what information the model is using.

2. **Computational Efficiency**: Compared to large transformer models, our approach is computationally lightweight. The LSTM backbone is efficient, and memory components (summarization, NER, semantic search) can be pre-computed and cached.

3. **Explicit Memory Management**: By explicitly managing short-term and long-term memory, we have fine-grained control over what information is retained and how it's used, allowing for domain-specific optimizations.

4. **Scalability**: The modular design allows for easy extension with additional memory components or domain-specific features (e.g., finance-specific NER).

5. **Progressive Improvement**: Our ablation study demonstrates that each memory component contributes to overall performance, validating the design choices.

6.2 Disadvantages

1. **Limited Context Window**: Despite memory augmentation, the model still has limitations in handling extremely long contexts. The token limit (256 tokens) and summarization may lose important details.

2. **Summarization Quality**: The quality of the summarization directly impacts model performance. Poor summaries can introduce noise or lose critical information.

3. **Entity Extraction Limitations**: NER may miss domain-specific entities or extract irrelevant ones, particularly in specialized domains like finance.

4. **Semantic Search Quality**: The semantic search component (Model 4) relies on embedding quality. Simple hash-based or lightweight embeddings may not capture true semantic similarity, limiting retrieval effectiveness.

5. **Training Data Requirements**: The model requires sequential training data with history, which may not always be available or may require careful data preparation.

6. **Evaluation Metric Mismatch**: The gap between LLM-as-a-judge scores and difflib scores suggests that traditional string similarity metrics may not fully capture the model's actual performance, making evaluation more complex.

7. Data, Time, and Computational Requirements

7.1 Data Requirements

- **Synthetic Dataset**: 200 question-answer pairs

- **Real Dataset**: 7000 question-answer pairs (200 sampled per epoch for training)

- Both datasets require sequential structure with history for memory evaluation

7.2 Time Requirements

- **Training**: $\sim$10 epochs per model, with each epoch taking approximately 36 minutes on NVIDIA GPU with CUDA

- **Pre-computation**: Summarization and NER features are cached to avoid recomputation

- **Total Training Time**: $\sim$6 hours for all 5 models on both datasets (10 training runs total) on NVIDIA GPU with CUDA. On Mac (CPU), estimated time is $\sim$24-30 hours

7.3 Computational Requirements

- **Hardware**: GPU recommended for LSTM training

- **Memory**: Moderate memory requirements for storing embeddings and cached features

- **Dependencies**: PyTorch, OpenAI API (for LLM-as-a-judge), transformers (for summarization/NER/autotokenizer)

8. Conclusions

Our Memory-Augmented LSTM successfully addresses the long-context retention problem in traditional LSTMs by integrating explicit memory management. The progressive improvement from Model 0 to Model 4 validates our approach of incrementally adding memory components, with Model 4 achieving the highest STM accuracy (0.550) and Model 3 achieving the highest LTM accuracy (0.800).

The model demonstrates strong performance in maintaining semantic coherence and retrieving long-term context, though it faces challenges with exact entity matching and generic response generation. The interpretable, modular design makes it suitable for domains requiring context continuity, such as multi-turn dialogue systems or resume-based skill extraction.

Future work could explore:

- Improved summarization techniques to preserve more detail

- Better semantic embeddings for more accurate retrieval

- Domain-specific fine-tuning of NER and memory components

- Hybrid approaches combining our memory-augmented LSTM with transformer architectures

8.1 Limitations.

The model has limitations in handling extremely long contexts, depends on summarization quality, may miss domain-specific entities, and requires sequential training data. Evaluation metrics may not fully capture semantic correctness.

8.2 Future Work.

Future work could explore improved summarization techniques, better semantic embeddings, domain-specific fine-tuning, and hybrid approaches combining memory-augmented LSTMs with transformer architectures.
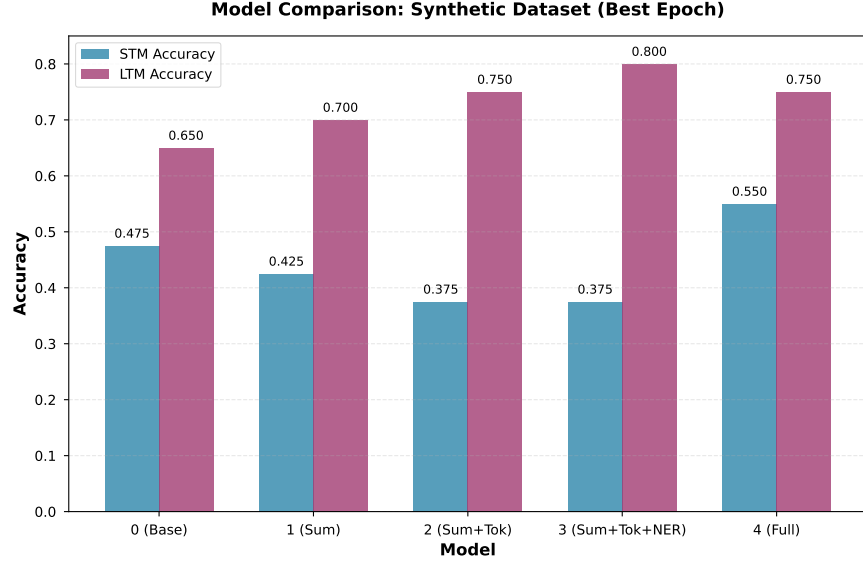
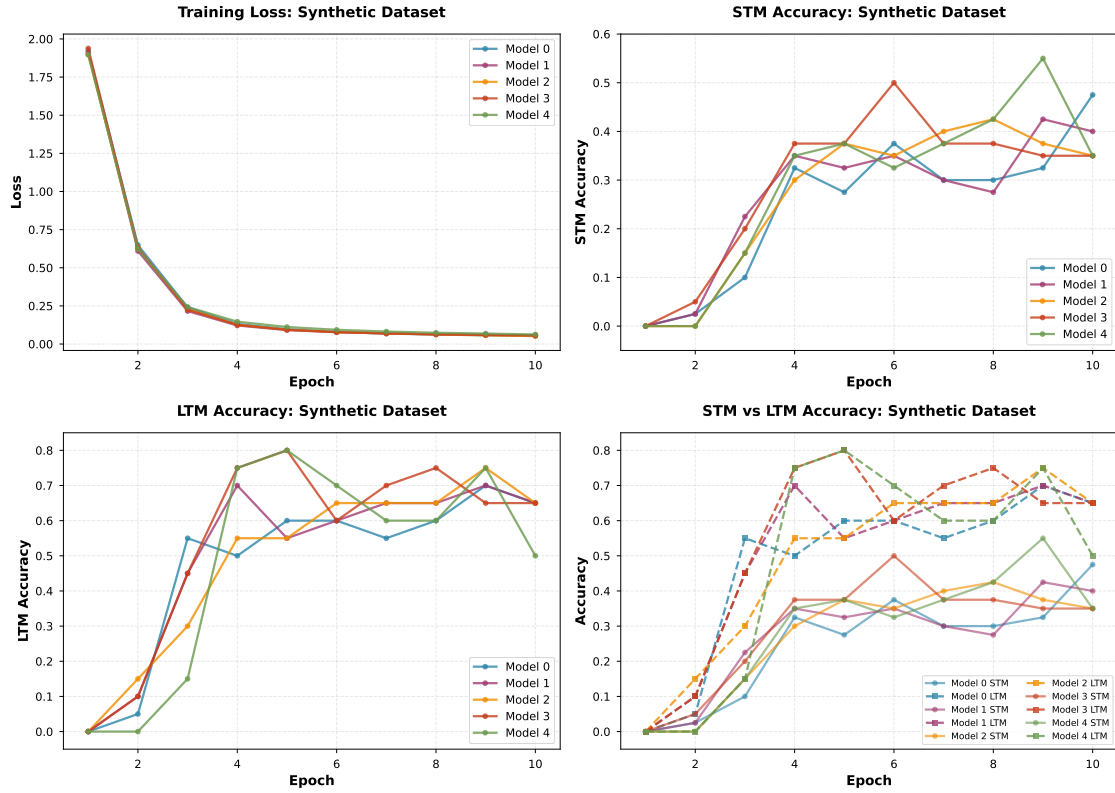Figure 1: Model comparison on synthetic dataset (best epoch).

Figure 2: Training progress over epochs for synthetic dataset.

Appendix A. Terminal Output

The epoch summaries for all models on both datasets are provided below. Note that only epoch summaries are included (debug examples omitted for brevity).
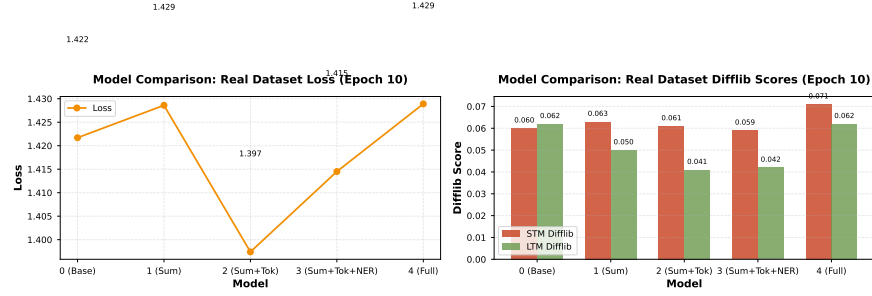
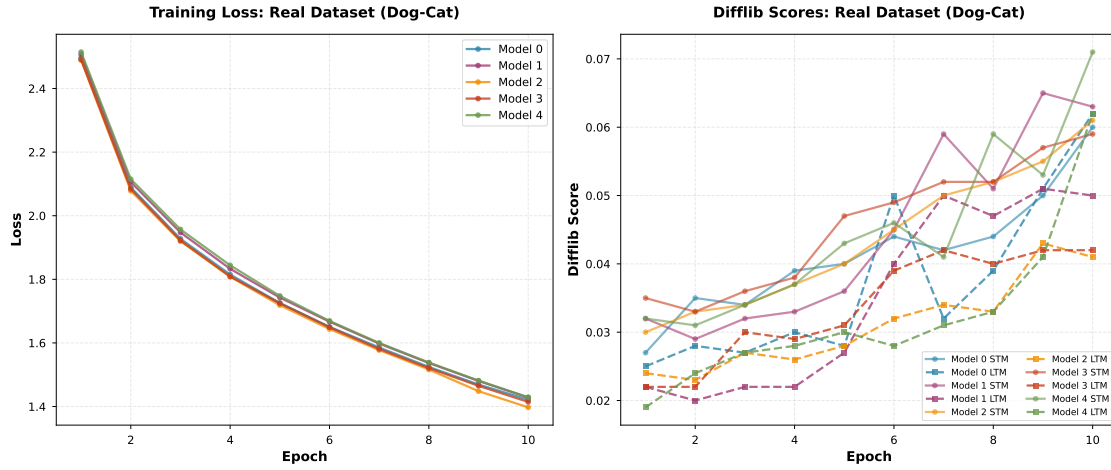Figure 3: Model comparison on real dataset (epoch 10).

Figure 4: Training progress over epochs for real dataset.

```
1   ----------------------------------------
2   Synthetic Data - SkillMiner
3   ----------------------------------------
4   0 base model
5   ----------------------------------------
6   Epoch 1
7   Loss: 1.9220
8   STM     acc=0.000, LLM=0.000, difflib=0.013
9   LTM     acc=0.000, LLM=0.000, difflib=0.011
10
11  Epoch 2
12  Loss: 0.6491
13  STM     acc=0.025, LLM=0.123, difflib=0.059
14  LTM     acc=0.050, LLM=0.090, difflib=0.042
15
16  Epoch 3
17  Loss: 0.2400
18  STM     acc=0.100, LLM=0.353, difflib=0.165
19  LTM     acc=0.550, LLM=0.415, difflib=0.113
20
21  Epoch 4
22  Loss: 0.1319
23  STM     acc=0.325, LLM=0.485, difflib=0.174
24  LTM     acc=0.500, LLM=0.460, difflib=0.175
25
26  Epoch 5
27  Loss: 0.0974
28  STM     acc=0.275, LLM=0.470, difflib=0.121
29  LTM     acc=0.600, LLM=0.490, difflib=0.170
30
31  Epoch 6
32  Loss: 0.0789
33  STM     acc=0.375, LLM=0.520, difflib=0.153
34  LTM     acc=0.600, LLM=0.465, difflib=0.187
35
36  Epoch 7
37  Loss: 0.0687
38  STM     acc=0.300, LLM=0.485, difflib=0.199
```

12

```
 39 | LTM      acc=0.550, LLM=0.445, difflib=0.155
 40 |
 41 | Epoch 8
 42 | Loss: 0.0621
 43 | STM      acc=0.300, LLM=0.490, difflib=0.149
 44 | LTM      acc=0.600, LLM=0.480, difflib=0.154
 45 |
 46 | Epoch 9
 47 | Loss: 0.0578
 48 | STM      acc=0.325, LLM=0.493, difflib=0.184
 49 | LTM      acc=0.700, LLM=0.480, difflib=0.224
 50 |
 51 | Epoch 10
 52 | Loss: 0.0541
 53 | STM      acc=0.475, LLM=0.570, difflib=0.202
 54 | LTM      acc=0.650, LLM=0.495, difflib=0.200
 55 |
 56 | --------------------------------------
 57 | 1 summarization_only
 58 | --------------------------------------
 59 |
 60 | Epoch 1
 61 | Loss: 1.9097
 62 | STM      acc=0.000, LLM=0.000, difflib=0.016
 63 | LTM      acc=0.000, LLM=0.000, difflib=0.011
 64 |
 65 | Epoch 2
 66 | Loss: 0.6097
 67 | STM      acc=0.025, LLM=0.172, difflib=0.079
 68 | LTM      acc=0.100, LLM=0.180, difflib=0.081
 69 |
 70 | Epoch 3
 71 | Loss: 0.2160
 72 | STM      acc=0.225, LLM=0.382, difflib=0.089
 73 | LTM      acc=0.450, LLM=0.385, difflib=0.089
 74 |
 75 | Epoch 4
 76 | Loss: 0.1216
 77 | STM      acc=0.350, LLM=0.470, difflib=0.131
 78 | LTM      acc=0.700, LLM=0.490, difflib=0.185
 79 |
 80 | Epoch 5
 81 | Loss: 0.0916
 82 | STM      acc=0.325, LLM=0.468, difflib=0.110
 83 | LTM      acc=0.550, LLM=0.445, difflib=0.197
 84 |
 85 | Epoch 6
 86 | Loss: 0.0777
 87 | STM      acc=0.350, LLM=0.490, difflib=0.132
 88 | LTM      acc=0.600, LLM=0.480, difflib=0.186
 89 |
 90 | Epoch 7
 91 | Loss: 0.0694
 92 | STM      acc=0.300, LLM=0.483, difflib=0.151
 93 | LTM      acc=0.650, LLM=0.500, difflib=0.197
 94 |
 95 | Epoch 8
 96 | Loss: 0.0639
 97 | STM      acc=0.275, LLM=0.480, difflib=0.191
 98 | LTM      acc=0.650, LLM=0.495, difflib=0.216
 99 |
100 | Epoch 9
101 | Loss: 0.0609
102 | STM      acc=0.425, LLM=0.525, difflib=0.124
103 | LTM      acc=0.700, LLM=0.515, difflib=0.193
104 |
105 | Epoch 10
106 | Loss: 0.0571
107 | STM      acc=0.400, LLM=0.505, difflib=0.180
108 | LTM      acc=0.650, LLM=0.510, difflib=0.211
109 |
110 | --------------------------------------
111 | 2 sum_token_limit
112 | --------------------------------------
113 |
114 | Epoch 1
115 | Loss: 1.8975
116 | STM      acc=0.000, LLM=0.000, difflib=0.014
117 | LTM      acc=0.000, LLM=0.000, difflib=0.013
118 |
119 | Epoch 2
120 | Loss: 0.6290
121 | STM      acc=0.000, LLM=0.158, difflib=0.074
122 | LTM      acc=0.150, LLM=0.145, difflib=0.070
123 |
124 | Epoch 3
125 | Loss: 0.2290
126 | STM      acc=0.150, LLM=0.330, difflib=0.099
127 | LTM      acc=0.300, LLM=0.350, difflib=0.100
128 |
129 | Epoch 4
130 | Loss: 0.1266
```

```
131  STM     acc=0.300, LLM=0.480, difflib=0.196
132  LTM     acc=0.550, LLM=0.475, difflib=0.178
133
134  Epoch 5
135  Loss: 0.0939
136  STM     acc=0.375, LLM=0.482, difflib=0.165
137  LTM     acc=0.550, LLM=0.500, difflib=0.146
138
139  Epoch 6
140  Loss: 0.0788
141  STM     acc=0.350, LLM=0.480, difflib=0.181
142  LTM     acc=0.650, LLM=0.500, difflib=0.152
143
144  Epoch 7
145  Loss: 0.0692
146  STM     acc=0.400, LLM=0.527, difflib=0.216
147  LTM     acc=0.650, LLM=0.495, difflib=0.228
148
149  Epoch 8
150  Loss: 0.0632
151  STM     acc=0.425, LLM=0.518, difflib=0.198
152  LTM     acc=0.650, LLM=0.475, difflib=0.263
153
154  Epoch 9
155  Loss: 0.0571
156  STM     acc=0.375, LLM=0.517, difflib=0.164
157  LTM     acc=0.750, LLM=0.555, difflib=0.205
158
159  Epoch 10
160  Loss: 0.0522
161  STM     acc=0.350, LLM=0.505, difflib=0.221
162  LTM     acc=0.650, LLM=0.515, difflib=0.236
163
164  ---------------------------------------
165  3 sum_tok_ner
166  ---------------------------------------
167
168  Epoch 1
169  Loss: 1.9378
170  STM     acc=0.000, LLM=0.000, difflib=0.010
171  LTM     acc=0.000, LLM=0.000, difflib=0.008
172
173  Epoch 2
174  Loss: 0.6317
175  STM     acc=0.050, LLM=0.160, difflib=0.068
176  LTM     acc=0.100, LLM=0.105, difflib=0.062
177
178  Epoch 3
179  Loss: 0.2249
180  STM     acc=0.200, LLM=0.378, difflib=0.115
181  LTM     acc=0.450, LLM=0.405, difflib=0.152
182
183  Epoch 4
184  Loss: 0.1254
185  STM     acc=0.375, LLM=0.505, difflib=0.143
186  LTM     acc=0.750, LLM=0.495, difflib=0.102
187
188  Epoch 5
189  Loss: 0.0930
190  STM     acc=0.375, LLM=0.500, difflib=0.140
191  LTM     acc=0.800, LLM=0.505, difflib=0.106
192
193  Epoch 6
194  Loss: 0.0781
195  STM     acc=0.500, LLM=0.532, difflib=0.153
196  LTM     acc=0.600, LLM=0.475, difflib=0.115
197
198  Epoch 7
199  Loss: 0.0699
200  STM     acc=0.375, LLM=0.517, difflib=0.188
201  LTM     acc=0.700, LLM=0.490, difflib=0.139
202
203  Epoch 8
204  Loss: 0.0634
205  STM     acc=0.375, LLM=0.500, difflib=0.192
206  LTM     acc=0.750, LLM=0.535, difflib=0.185
207
208  Epoch 9
209  Loss: 0.0588
210  STM     acc=0.350, LLM=0.500, difflib=0.179
211  LTM     acc=0.650, LLM=0.500, difflib=0.171
212
213  Epoch 10
214  Loss: 0.0558
215  STM     acc=0.350, LLM=0.522, difflib=0.177
216  LTM     acc=0.650, LLM=0.510, difflib=0.192
217
218  ---------------------------------------
219  4 full_memory
220  ---------------------------------------
221
222  Epoch 1
```

```
223  Loss: 1.8962
224  STM     acc=0.000, LLM=0.000, difflib=0.005
225  LTM     acc=0.000, LLM=0.000, difflib=0.003
226
227  Epoch 2
228  Loss: 0.6234
229  STM     acc=0.000, LLM=0.060, difflib=0.051
230  LTM     acc=0.000, LLM=0.070, difflib=0.055
231
232  Epoch 3
233  Loss: 0.2426
234  STM     acc=0.150, LLM=0.330, difflib=0.077
235  LTM     acc=0.150, LLM=0.250, difflib=0.089
236
237  Epoch 4
238  Loss: 0.1466
239  STM     acc=0.350, LLM=0.465, difflib=0.154
240  LTM     acc=0.750, LLM=0.510, difflib=0.073
241
242  Epoch 5
243  Loss: 0.1116
244  STM     acc=0.375, LLM=0.512, difflib=0.113
245  LTM     acc=0.800, LLM=0.535, difflib=0.110
246
247  Epoch 6
248  Loss: 0.0933
249  STM     acc=0.325, LLM=0.508, difflib=0.110
250  LTM     acc=0.700, LLM=0.495, difflib=0.130
251
252  Epoch 7
253  Loss: 0.0818
254  STM     acc=0.375, LLM=0.500, difflib=0.098
255  LTM     acc=0.600, LLM=0.485, difflib=0.097
256
257  Epoch 8
258  Loss: 0.0740
259  STM     acc=0.425, LLM=0.515, difflib=0.096
260  LTM     acc=0.600, LLM=0.500, difflib=0.165
261
262  Epoch 9
263  Loss: 0.0687
264  STM     acc=0.550, LLM=0.512, difflib=0.056
265  LTM     acc=0.750, LLM=0.505, difflib=0.112
266
267  Epoch 10
268  Loss: 0.0627
269  STM     acc=0.350, LLM=0.487, difflib=0.086
270  LTM     acc=0.500, LLM=0.470, difflib=0.133
271
272  --------------------------------------
273  Real Data - Dog cat(kaggle)
274  --------------------------------------
275  0 base
276  --------------------------------------
277
278  Epoch 1
279  Loss: 2.4974
280  STM     acc=0.000, LLM=0.000, difflib=0.027
281  LTM     acc=0.000, LLM=0.000, difflib=0.025
282
283  Epoch 2
284  Loss: 2.0891
285  STM     acc=0.000, LLM=0.000, difflib=0.035
286  LTM     acc=0.000, LLM=0.000, difflib=0.028
287
288  Epoch 3
289  Loss: 1.9274
290  STM     acc=0.000, LLM=0.000, difflib=0.034
291  LTM     acc=0.000, LLM=0.000, difflib=0.027
292
293  Epoch 4
294  Loss: 1.8149
295  STM     acc=0.000, LLM=0.000, difflib=0.039
296  LTM     acc=0.000, LLM=0.000, difflib=0.030
297
298  Epoch 5
299  Loss: 1.7257
300  STM     acc=0.000, LLM=0.000, difflib=0.040
301  LTM     acc=0.000, LLM=0.000, difflib=0.028
302
303  Epoch 6
304  Loss: 1.6507
305  STM     acc=0.000, LLM=0.000, difflib=0.044
306  LTM     acc=0.000, LLM=0.000, difflib=0.050
307
308  Epoch 7
309  Loss: 1.5855
310  STM     acc=0.000, LLM=0.000, difflib=0.042
311  LTM     acc=0.000, LLM=0.000, difflib=0.032
312
313  Epoch 8
314  Loss: 1.5244
```

```
315  STM      acc=0.000, LLM=0.000, difflib=0.044
316  LTM      acc=0.000, LLM=0.000, difflib=0.039
317
318  Epoch 9
319  Loss: 1.4691
320  STM      acc=0.000, LLM=0.000, difflib=0.050
321  LTM      acc=0.000, LLM=0.000, difflib=0.051
322
323  Epoch 10
324  Loss: 1.4217
325  STM      acc=0.000, LLM=0.000, difflib=0.060
326  LTM      acc=0.000, LLM=0.000, difflib=0.062
327
328  ---------------------------------------
329  1 summarization_only
330  ---------------------------------------
331
332  Epoch 1
333  Loss: 2.5077
334  STM      acc=0.000, LLM=0.000, difflib=0.032
335  LTM      acc=0.000, LLM=0.000, difflib=0.022
336
337  Epoch 2
338  Loss: 2.1059
339  STM      acc=0.000, LLM=0.000, difflib=0.029
340  LTM      acc=0.000, LLM=0.000, difflib=0.020
341
342  Epoch 3
343  Loss: 1.9473
344  STM      acc=0.000, LLM=0.000, difflib=0.032
345  LTM      acc=0.000, LLM=0.000, difflib=0.022
346
347  Epoch 4
348  Loss: 1.8331
349  STM      acc=0.000, LLM=0.000, difflib=0.033
350  LTM      acc=0.000, LLM=0.000, difflib=0.022
351
352  Epoch 5
353  Loss: 1.7427
354  STM      acc=0.000, LLM=0.000, difflib=0.036
355  LTM      acc=0.000, LLM=0.000, difflib=0.027
356
357  Epoch 6
358  Loss: 1.6668
359  STM      acc=0.000, LLM=0.000, difflib=0.045
360  LTM      acc=0.000, LLM=0.000, difflib=0.040
361
362  Epoch 7
363  Loss: 1.5980
364  STM      acc=0.000, LLM=0.000, difflib=0.059
365  LTM      acc=0.000, LLM=0.000, difflib=0.050
366
367  Epoch 8
368  Loss: 1.5370
369  STM      acc=0.000, LLM=0.000, difflib=0.051
370  LTM      acc=0.000, LLM=0.000, difflib=0.047
371
372  Epoch 9
373  Loss: 1.4804
374  STM      acc=0.000, LLM=0.000, difflib=0.065
375  LTM      acc=0.000, LLM=0.000, difflib=0.051
376
377  Epoch 10
378  Loss: 1.4286
379  STM      acc=0.000, LLM=0.000, difflib=0.063
380  LTM      acc=0.000, LLM=0.000, difflib=0.050
381
382  ---------------------------------------
383  2 sum_token_limit
384  ---------------------------------------
385
386  Epoch 1
387  Loss: 2.4895
388  STM      acc=0.000, LLM=0.000, difflib=0.030
389  LTM      acc=0.000, LLM=0.000, difflib=0.024
390
391  Epoch 2
392  Loss: 2.0783
393  STM      acc=0.000, LLM=0.000, difflib=0.033
394  LTM      acc=0.000, LLM=0.000, difflib=0.023
395
396  Epoch 3
397  Loss: 1.9200
398  STM      acc=0.000, LLM=0.000, difflib=0.034
399  LTM      acc=0.000, LLM=0.000, difflib=0.027
400
401  Epoch 4
402  Loss: 1.8080
403  STM      acc=0.000, LLM=0.000, difflib=0.037
404  LTM      acc=0.000, LLM=0.000, difflib=0.026
405
406  Epoch 5
```

```
Loss: 1.7187
STM     acc=0.000, LLM=0.000, difflib=0.040
LTM     acc=0.000, LLM=0.000, difflib=0.028

Epoch 6
Loss: 1.6437
STM     acc=0.000, LLM=0.000, difflib=0.045
LTM     acc=0.000, LLM=0.000, difflib=0.032

Epoch 7
Loss: 1.5775
STM     acc=0.000, LLM=0.000, difflib=0.050
LTM     acc=0.000, LLM=0.000, difflib=0.034

Epoch 8
Loss: 1.5164
STM     acc=0.000, LLM=0.000, difflib=0.052
LTM     acc=0.000, LLM=0.000, difflib=0.033

Epoch 9
Loss: 1.4483
STM     acc=0.000, LLM=0.000, difflib=0.055
LTM     acc=0.000, LLM=0.000, difflib=0.043

Epoch 10
Loss: 1.3974
STM     acc=0.000, LLM=0.000, difflib=0.061
LTM     acc=0.000, LLM=0.000, difflib=0.041

----------------------------------------
3 sum_tok_ner
----------------------------------------

Epoch 1
Loss: 2.4910
STM     acc=0.000, LLM=0.000, difflib=0.035
LTM     acc=0.000, LLM=0.000, difflib=0.022

Epoch 2
Loss: 2.0844
STM     acc=0.000, LLM=0.000, difflib=0.033
LTM     acc=0.000, LLM=0.000, difflib=0.022

Epoch 3
Loss: 1.9222
STM     acc=0.000, LLM=0.000, difflib=0.036
LTM     acc=0.000, LLM=0.000, difflib=0.030

Epoch 4
Loss: 1.8090
STM     acc=0.000, LLM=0.000, difflib=0.038
LTM     acc=0.000, LLM=0.000, difflib=0.029

Epoch 5
Loss: 1.7245
STM     acc=0.000, LLM=0.000, difflib=0.047
LTM     acc=0.000, LLM=0.000, difflib=0.031

Epoch 6
Loss: 1.6492
STM     acc=0.000, LLM=0.000, difflib=0.049
LTM     acc=0.000, LLM=0.000, difflib=0.039

Epoch 7
Loss: 1.5811
STM     acc=0.000, LLM=0.000, difflib=0.052
LTM     acc=0.000, LLM=0.000, difflib=0.042

Epoch 8
Loss: 1.5209
STM     acc=0.000, LLM=0.000, difflib=0.052
LTM     acc=0.000, LLM=0.000, difflib=0.040

Epoch 9
Loss: 1.4651
STM     acc=0.000, LLM=0.000, difflib=0.057
LTM     acc=0.000, LLM=0.000, difflib=0.042

Epoch 10
Loss: 1.4145
STM     acc=0.000, LLM=0.000, difflib=0.059
LTM     acc=0.000, LLM=0.000, difflib=0.042

----------------------------------------
4 full_memory
----------------------------------------

Epoch 1
Loss: 2.5147
STM     acc=0.000, LLM=0.000, difflib=0.032
LTM     acc=0.000, LLM=0.000, difflib=0.019
```

```
499  Epoch 2
500  Loss: 2.1149
501  STM     acc=0.000, LLM=0.000, difflib=0.031
502  LTM     acc=0.000, LLM=0.000, difflib=0.024
503
504  Epoch 3
505  Loss: 1.9569
506  STM     acc=0.000, LLM=0.000, difflib=0.034
507  LTM     acc=0.000, LLM=0.000, difflib=0.027
508
509  Epoch 4
510  Loss: 1.8441
511  STM     acc=0.000, LLM=0.000, difflib=0.037
512  LTM     acc=0.000, LLM=0.000, difflib=0.028
513
514  Epoch 5
515  Loss: 1.7479
516  STM     acc=0.000, LLM=0.000, difflib=0.043
517  LTM     acc=0.000, LLM=0.000, difflib=0.030
518
519  Epoch 6
520  Loss: 1.6693
521  STM     acc=0.000, LLM=0.000, difflib=0.046
522  LTM     acc=0.000, LLM=0.000, difflib=0.028
523
524  Epoch 7
525  Loss: 1.6001
526  STM     acc=0.000, LLM=0.000, difflib=0.041
527  LTM     acc=0.000, LLM=0.000, difflib=0.031
528
529  Epoch 8
530  Loss: 1.5384
531  STM     acc=0.000, LLM=0.000, difflib=0.059
532  LTM     acc=0.000, LLM=0.000, difflib=0.033
533
534  Epoch 9
535  Loss: 1.4816
536  STM     acc=0.000, LLM=0.000, difflib=0.053
537  LTM     acc=0.000, LLM=0.000, difflib=0.041
538
539  Epoch 10
540  Loss: 1.4289
541  STM     acc=0.000, LLM=0.000, difflib=0.071
542  LTM     acc=0.000, LLM=0.000, difflib=0.062
```

Dataset References

- **Synthetic Dataset**: The SkillMiner QA dataset was generated from a ChatGPT conversation OpenAI (2025). This AI-generated dataset was created specifically for evaluating memory-augmented language models in a question-answering context.

- **Dog-Cat Dataset**: The Dog-Cat QA dataset Shahi (2024) was obtained from Kaggle and contains 200 question-answer pairs focusing on pet care and behavior.

- **Implementation**: Our implementation code is publicly available on GitHub Jiang et al. (2025).

References

Aaron Jiang, Jay Wu, and Leo Yao. Memory-augmented lstm implementation, 2025. URL https://github.com/JayWu0512/memory-augmented-lstm-research. GitHub repository containing implementation code for memory-augmented LSTM models.

OpenAI. Synthetic skillminer qa dataset, 2025. Dataset generated from ChatGPT conversation.

Bishnu Shahi. Dog-cat-qa, 2024. URL https://www.kaggle.com/dsv/9016983.