Reproducibility Work Report (FTEC5660)

Student: Shiliang Chen
Student ID: 1155245855
Date: 2026-02-17

Project summary

- **Reproduced Project**: MetaMind — a multi-agent framework for social reasoning / Theory of Mind (ToM).
- **Repo location**: `homeworks/Reproducibility Work/MetaMind/`
- **Introduction**: the system runs a multi-stage pipeline (ToM Agent → Domain Agent → Response Agent) with multi-step reasoning and internal selection/refinement.

Setup notes (env, data, keys, compute)

- **Python**: 3.12.12
- **Install**:

```
cd "homeworks/Reproducibility Work/MetaMind"
python3 -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
```

- **LLM provider**: DeepSeek via Volcengine Ark (OpenAI-compatible endpoint).
  - **API key**: API key loaded from environment variables / local `.env`.
  - **Example env**: see `.env.example`.
  - **Controlled change (model swap)**: the upstream project examples reference OpenAI models (e.g., GPT-4). I used DeepSeek (Ark) to make the pipeline runnable with accessible credentials. Therefore, headline numbers in the paper (GPT-4 ToMBench average) are not directly comparable to my measurements.
- **Dataset**: ToMBench (JSONL) shipped as `evaluations.zip` and extracted under `evaluations/`.

Reproduction target(s) + metric definition

Target

Reproduce **ToMBench multiple-choice accuracy** on a clearly-defined subset:

- **Benchmark**: ToMBench
- **Task**: `Ambiguous Story Task`

- **Subset**: first N examples where $N \in \{10, 20, 30\}$

This is a scoped reproduction (allowed by the assignment), focusing on a single well-defined task subset.

Metric

- **Accuracy** (%) $= \frac{\text{correct}}{\text{total}} \times 100$
- **Correctness rule**: predicted option letter in $\{A, B, C, D\}$ matches the dataset label.

Results

My measured results

Setup: DeepSeek Ark; temperature=0; hypothesis_count=7.

Command used:

```
cd "homeworks/Reproducibility Work/MetaMind"
LLM_TEMPERATURE=0 LLM_MAX_TOKENS=200 TOM_HYPOTHESIS_COUNT=7 python evaluations/tombench/eva
```

Checkpoint accuracies (cumulative):

Subset size	Correct	Accuracy (%)
10	7	70.00
20	11	55.00
30	15	50.00

Repeated runs (3 trials; final accuracy on 30 examples; hypothesis_count=7):

Trial	Correct	Accuracy (%)
1	15/30	50.00
2	21/30	70.00
3	17/30	56.67

Summary statistics (accuracy in percentage points):

- Mean: **58.89%**
- Sample variance (n-1): **103.70**
- Sample standard deviation: **10.18**

Caveat (evaluation setup): the evaluation script reuses one `MetamindApplication` instance and updates `SocialMemory` for the same `user_id` across questions, so

ToMBench items are not strictly independent in this setup. This can contribute to accuracy fluctuations as N increases.

Reference numbers and comparability caveat

The upstream repo/paper reports **overall ToMBench** accuracy numbers under specific model settings (e.g., the project README cites **74.8%** average accuracy for base GPT-4).

In this reproduction, I use a different underlying model/provider (DeepSeek via Ark) and evaluate only a scoped subset (the first 30 items of `Ambiguous Story Task`), so my results are **not directly comparable** to those headline ToMBench averages.

Instead, I treat this as a clearly-defined, measurable subset reproduction target.

Modification + results after modification

Planned modification (small, isolated, measurable):

- **Change**: `TOM_AGENT_CONFIG["hypothesis_count"]` (**7** → **3**; still overrideable via `TOM_HYPOTHESIS_COUNT`)
- **Why**: this is a key ToM-stage parameter (number of hypotheses generated), and the repo discusses sensitivity to hypothesis count.
- **Measurement**: re-run the same task subset and compare Accuracy before/after.

Command used:

```
cd "homeworks/Reproducibility Work/MetaMind"
LLM_TEMPERATURE=0 LLM_MAX_TOKENS=200 TOM_HYPOTHESIS_COUNT=3 python evaluations/tombench/eval
```

Checkpoint accuracies (cumulative; after modification, hypothesis_count=3):

Subset size	Correct	Accuracy (%)
10	7	70.00
20	14	70.00
30	19	63.33

Repeated runs (3 trials; final accuracy on 30 examples; hypothesis_count=3):

Trial	Correct	Accuracy (%)
1	19/30	63.33
2	18/30	60.00
3	20/30	66.67

Summary statistics (accuracy in percentage points):

- Mean: **63.33%**
- Sample variance (n-1): **11.12**
- Sample standard deviation: **3.34**

Analysis (why lower hypothesis_count can improve accuracy here)

- **Parsing artifact (original evaluation design)**: the evaluation script (as originally provided) extracts the **first** standalone option letter using regex `\\b[A-D]\\b` from `final_response`. When `hypothesis_count` is larger, the multi-stage pipeline tends to produce longer outputs (more option letters mentioned during reasoning), increasing the chance that the **first** A/B/C/D is not the intended final choice.
- **Hypothesis noise**: increasing `hypothesis_count` can add lower-quality or misleading ToM hypotheses; the Domain/Response stages may follow a plausible but incorrect hypothesis more often. With fewer hypotheses (k=3), selection is simpler and can be more stable on this small subset.
- **Non-independent evaluation due to SocialMemory**: the evaluator reuses a single `MetamindApplication` instance and updates `SocialMemory` for the same default user across questions, so items are not strictly independent. Changing `hypothesis_count` changes what gets written into memory and can influence later questions.
- **Variance + small sample**: the subset is only 30 examples, so differences of 1–2 questions can shift accuracy by several percentage points. Even with `temperature=0`, provider-side variability or subtle nondeterminism can contribute to run-to-run fluctuations.

Variance observation (k=7 vs k=3)

- Across 3 trials, `hypothesis_count=7` shows **higher run-to-run variance** than `hypothesis_count=3` (sample sd **10.18** vs **3.34** percentage points in this setup).
- This is an observational result on a small subset, but it is consistent with the idea that a larger hypothesis count can make the overall pipeline more sensitive to small differences.

Possible reasons (why larger `hypothesis_count` may increase variance)

- **Error accumulation (plausible)**: larger `hypothesis_count` typically implies more upstream generation/refinement and longer multi-stage reasoning. Even small run-to-run differences can compound across stages and lead to larger variability in the final prediction.

- **Larger branching / selection sensitivity (plausible)**: with more candidate hypotheses, the Domain/Response stages may face more near-ties. Small differences across runs can flip the selected hypothesis, which can cascade into different downstream responses and predictions.

Debug diary (main blockers + resolutions)

- **Blocker**: dependency install failed under system-managed Python (PEP 668).
  - **Fix**: created a local virtualenv `.venv` and installed `requirements.txt`.
- **Blocker**: ToMBench JSONL files not found at `evaluations/tombench/` after unzip.
  - **Fix**: dataset unzipped into nested path `evaluations/evaluations/tombench/`; evaluation script updated to auto-detect both layouts.
- **Blocker**: `ModuleNotFoundError: No module named 'main'` when running eval script.
  - **Fix**: added project-root `sys.path` injection in `eval_tombench.py` so it can import `main.py` when executed directly.
- **Blocker**: ToMBench label key mismatch across JSONLs.
- **Blocker**: evaluation is slow.
  - **Reason**: multiple LLM calls per example (ToM hypotheses + domain + response + possible revisions).
  - **Mitigation**: use a scoped subset (30 items) and checkpoint reporting.

Conclusions

- **What is reproducible**: the MetaMind pipeline runs end-to-end under the documented setup (DeepSeek via Volcengine Ark) and yields measurable ToMBench accuracy on the scoped subset (`Ambiguous Story Task`, first 30 examples) using the repo evaluation script.
- **Reproducible measured outcomes (scoped subset)**:
  - `hypothesis_count=7`: **58.89%**
  - `hypothesis_count=3` (single-parameter modification): **63.33%**
- **What is not directly reproducible / comparable**: paper / upstream headline ToMBench averages under GPT-4 (e.g., 74.8%) are not directly reproducible in this setting because the underlying model/provider differs and this report evaluates only a small subset rather than the full benchmark.
- **Why (main reasons)**: outcomes are sensitive to evaluation details (small sample size, letter-extraction parsing, and non-independent `SocialMemory` updates across items), so accuracy can fluctuate and should be interpreted as a scoped measurement rather than a universal ranking.

Key learnings

- **Evaluation choices get amplified in multi-agent pipelines**: each stage can add text. A simple answer parser (e.g., extracting the first standalone A/B/C/D) can interact with longer multi-stage outputs and affect measured accuracy. Sometimes the metric interface is brittle.
- **The ToM stage is a real control knob**: changing `hypothesis_count` changes how many candidate interpretations are generated upstream, which changes what the Domain Agent sees and what the Response Agent conditions on. A single upstream parameter can shift final accuracy.
- **SocialMemory makes the system stateful**: the evaluator updates memory across questions, so items are not strictly independent. Changing `hypothesis_count` can also change what gets written into memory, which can influence later questions in hard-to-predict ways.
- **Cost/latency scales with hypotheses and steps**: more hypotheses typically means more calls and more runtime, so scoped evaluation is a practical way to get measurable results within a budget.