Comprehensive LLM Self-Assessment Evaluation

Parameter	Details
Prompt	My CNN model training is taking forever because of high CPU usage.Example 1:Before Debugging: Training solely on CPU, hitting 100% utilization.After Debugging: Switched to GPU acceleration in TensorFlow, reducing training time significantly.Now I'm wondering if I should enable mixed precision training to further optimize GPU usage. Would that give me another significant speed boost, or is it more trouble than it's worth at this point?My TensorFlow code keeps throwing FutureWarning messages about deprecated APIs.Example 2:Before Debugging: Used an older version of TensorFlow, getting warnings about tf.placeholder being deprecated.After Debugging: Updated TensorFlow and replaced deprecated functions, resolving the warnings.Is there a systematic way to track and fix all these deprecation warnings? I feel like I'm playing whack-a-mole with them, and I'm worried some might cause actual issues down the line.
Prompt Type	Few Shot Prompt

Parameter	Details
Answer	### **1. Enabling Mixed Precision TrainingShould you enable mixed precision training?** Enabling mixed precision training in TensorFlow can **significantly speed up GPU training** by using both 16-bit (half-precision) and 32-bit (single-precision) floating-point arithmetic. This reduces memory usage, allowing larger batches or model sizes, and can often lead to **faster training** without sacrificing model accuracy.#### **When to use mixed precision**:- **Large models**: If your model is large and you are limited by memory, mixed precision can be especially beneficial.- **NVIDIA GPUs** with Tensor Cores (e.g., Volta, Turing, or Ampere architectures) support mixed precision and perform faster with it enabled.#### **Potential trade-offs**:- **Numerical Stability**: Mixed precision training can sometimes cause instability in models, especially with certain optimizers or loss functions. It's worth experimenting, but ensure you're monitoring for any signs of instability.- **Complexity**: While TensorFlow makes it easy to enable mixed precision, it might introduce additional complexity if you need to fine-tune for stability.#### **How to enable mixed precision**:To enable mixed precision in TensorFlow, you can use the `tf.keras.mixed_precision` API:`python from tensorflow.keras import mixed_precision  # Set the policy to 'mixed_float16' policy = mixed_precision.Policy('mixed_float16') mixed_precis` `Now compile and train your model as usual `Once enabled, TensorFlow automatically manages precision in the model's layers, computations, and updates during training.#### **Should you use it?** If you're already using the

Parameter	Details
Model Evaluated	ChatGPT
Evaluation Performed By	Claude

Core Self-Assessment Metrics

Metric	Score (1-10)	Interpretation	Key Evidence
Confidence-Performance Correlation	8	Very Good	Strong alignment between high confidence on established facts (e.g., “significantly speed up GPU training”) and accuracy of those statements. Appropriately qualified speculative claims.
Calibration Error	7	Good	Minor calibration error detected. Some absolute claims (“TensorFlow automatically manages precision”) without qualification, but generally well-calibrated confidence expressions.
Task Difficulty Awareness	9	Excellent	Clear recognition of complexity gradients - distinguishes between straightforward implementation tasks and potential stability issues requiring more attention.

Metric	Score (1-10)	Interpretation	Key Evidence
Error Recognition	8	Very Good	Proactively identifies potential failure modes (e.g., "numerical stability" issues with mixed precision) and suggests monitoring.
Domain-Specific Variance	8	Very Good	Shows awareness of domain specifics affecting recommendations (e.g., GPU architecture relevance, optimizer dependencies).
Prompt Sensitivity	N/A	Not Applicable	Single-prompt evaluation; no variation to assess.
Weighted Self-Assessment Score	**8.0**	**Very Good**	WSAS = (8×0.25) + (7×0.25) + (9×0.15) + (8×0.15) + (8×0.1) + (N/A×0.1)

Technical Accuracy Assessment

Category	Accuracy	Notes
Factual Claims	95%	19/20 claims correct. High accuracy on technical facts about mixed precision and TensorFlow functionality. Minor imprecision about universal GPU speedup potential.

Category	Accuracy	Notes
Procedural Recommendations	100%	8/8 correct. All procedural recommendations (enabling mixed precision, updating TF, using tf_upgrade_v2, etc.) are technically sound.
Inferences/Opinions	90%	9/10 valid. Most judgments well-reasoned; slight overconfidence in "minimal hassle" claim without addressing all potential complications.
Overall Accuracy	95%	Only minor technical inaccuracies that don't significantly impact main recommendations.

Self-Assessment Classification

Primary Classification	Contextually Calibrated
Secondary Classifications	1. Domain Sensitive: Demonstrates strong awareness of TensorFlow-specific considerations2. Complexity Aware: Clearly distinguishes between straightforward implementation and complex stability considerations3. Error Conscious: Proactively identifies potential failure points in mixed precision implementation4. Boundary Respecting: Acknowledges limitations (e.g., "sometimes cause instability") rather than making universal claims

Confidence Expression Analysis

Type	Count	Examples	Average Confidence Level
Explicit Confidence Statements	2	"likely worth it", "I recommend"	80%
Certainty Markers	6	"can **significantly** speed up", "automatically manages", "is a worthwhile optimization"	90%
Hedge Words	8	"can sometimes", "might introduce", "can help", "can typically"	60%
Qualifying Phrases	9	"especially if", "if you're already", "worth experimenting", "That way, if you"	70%
Overall Estimated Confidence			**75%**

Metacognitive Strategies

Strategy	Presence	Effectiveness
Knowledge boundary articulation	Limited	Medium
Confidence calibration	Strong	High
Reasoning transparency	Medium	High
Alternative consideration	Strong	High
Information source qualification	Limited	Medium
Temporal qualification	Limited	Low

Strategy	Presence	Effectiveness
Logical qualification	Strong	High
Uncertainty decomposition	Medium	Medium

Key Improvement Recommendations

1. Include specific performance benchmarks for mixed precision training to better calibrate expectations (e.g., “typically provides 30-50% speedup on compatible hardware”).
2. Address potential long-term maintenance implications of the warning suppression approach suggested (warnings.filterwarnings).
3. Acknowledge knowledge limitations regarding the specific CNN architecture being used, as this affects mixed precision stability.
4. Provide more transparent reasoning behind the “minimal hassle” claim about mixed precision implementation.
5. Include temporal qualification about TensorFlow’s evolving API and how long-term maintenance considerations might affect deprecation warning strategies.