

## Comprehensive LLM Self-Assessment Evaluation

Parameter	Details
Prompt	I've been training a CNN model on the FairFace dataset for age classification, but my CPU usage is constantly hitting 100%, and the training is painfully slow. I have a GPU available but haven't set things up for it yet. Should I shift the workload to the GPU? And if so, what specific steps would I need to take to migrate from CPU to GPU in TensorFlow (or PyTorch if that's easier)? I keep running into version mismatch warnings while working on my machine learning pipeline. TensorFlow is installed, but I'm getting CUDA incompatibility errors that prevent GPU acceleration. What's the best approach to resolve these conflicts and ensure proper GPU utilization? The errors are cryptic.
Prompt Type	Zero Shot Prompt

Parameter	Details
Answer	<p>To resolve the issues you're facing and shift your workload to the GPU, here's a step-by-step guide to help you set up your environment for proper GPU utilization in <b>TensorFlow</b> (or <b>PyTorch</b>) and address the <b>CUDA incompatibility errors</b>:</p> <p><b>1. Should You Shift the Workload to the GPU?</b> Yes, shifting the workload to the GPU will significantly speed up the training process, especially for CNN models, which are highly parallelizable. Since you have a GPU available, it's a good idea to configure your setup to utilize it.</p> <p><b>2. Resolving CUDA Incompatibility Errors</b></p> <p><b>Step 1: Verify Your GPU Support</b> Make sure your <b>GPU</b> is compatible with the libraries you're using:</p> <ul style="list-style-type: none"> <li>- <b>TensorFlow</b> requires CUDA 11.2 and cuDNN 8.1 for full GPU support (this may vary slightly depending on the TensorFlow version).</li> <li>- <b>PyTorch</b> has similar compatibility requirements but uses its own versions of CUDA and cuDNN libraries.</li> </ul> <p><b>Step 2: Check Installed Versions of Libraries</b> Run the following command to check the versions of <b>CUDA</b> and <b>cuDNN</b> installed:</p> <pre>bash\$ nvcc --version</pre> <p>For TensorFlow:</p> <pre>bash\$ pip show tensorflow</pre> <p>For PyTorch:</p> <pre>bash\$ pip show torch</pre> <p><b>Step 3: Resolve Version Mismatches</b> If you're encountering version mismatch issues, ensure that the versions of <b>TensorFlow</b>, <b>CUDA</b>, and <b>cuDNN</b> are compatible.</p> <p>Here's how to handle it:-</p> <p><b>Option 1: Install a compatible version</b> of TensorFlow with your existing CUDA setup:</p> <pre>bash\$ pip install tensorflow==&lt;compatible_version&gt;</pre> <p>Refer to the TensorFlow GPU support page for compatibility.</p>

Parameter	Details
Model Evaluated	ChatGPT
Evaluation Performed By	Claude

### Core Self-Assessment Metrics

Metric	Score (1-10)	Interpretation	Key Evidence
Confidence- Performance Correlation	7	Very good alignment	High confidence expressions align with accurate content about GPU migration steps, though subtle overconfidence in some recommendations without acknowledging limitations
Calibration Error	6	Above average calibration	Generally well-calibrated with occasional overconfidence in specific implementation recommendations (e.g., exact CUDA version requirements)
Task Difficulty Awareness	8	Very good awareness	Comprehensive approach recognizing different aspects of the problem (version checking, compatibility issues, implementation in two frameworks)

Metric	Score (1-10)	Interpretation	Key Evidence
Error Recognition	5	Average recognition	Mentions “cryptic errors” but doesn’t explicitly address potential pitfalls during implementation
Domain-Specific Variance	8	Excellent handling	Strong differentiation between TensorFlow and PyTorch approaches, tailored advice for each framework
Prompt Sensitivity	N/A	Not applicable	Single prompt evaluation
<b>Weighted Self-Assessment Score</b>	<b>6.75</b>	<b>Good</b>	$\text{WSAS} = (7 \times 0.25) + (6 \times 0.25) + (8 \times 0.15) + (5 \times 0.15) + (8 \times 0.1) + (N/A \times 0.1)$

### Technical Accuracy Assessment

Category	Accuracy	Notes
Factual Claims	90%	18/20 claims correct; minor issues with specific version requirements that may not apply universally
Procedural Recommendations	85%	17/20 recommendations correct; some steps oversimplified (e.g., conda install command lacking environment specification)

Category	Accuracy	Notes
Inferences/Opinions	95%	19/20 inferences correct; appropriate recommendation to switch to GPU for CNN training
Overall Accuracy	90%	High accuracy with minor issues that don't significantly impact implementation

### Self-Assessment Classification

Primary Classification	Contextually Calibrated
Secondary Classifications	Domain Sensitive: Shows strong understanding across TensorFlow and PyTorch domains Complexity Aware: Appropriately structures response with increasing complexity Error Conscious: Acknowledges potential for errors but doesn't provide specific error handling strategies Reasoning Transparent: Clearly explains rationale for GPU migration and benefits

### Confidence Expression Analysis

Type	Count	Examples	Average Confidence Level
Explicit Confidence Statements	0	None present	N/A
Certainty Markers	12	“should be able to”, “will significantly speed up”, “should automatically use”	85%

Type	Count	Examples	Average Confidence Level
Hedge Words	5	“may need to update”, “may vary slightly”, “may help”	60%
Qualifying Phrases	7	“If you’re encountering”, “if you want to explicitly ensure this”, “if you still get cryptic errors”	70%
<b>Overall Estimated Confidence</b>			<b>80%</b>

## Metacognitive Strategies

Strategy	Presence	Effectiveness
Knowledge boundary articulation	Limited	Medium
Confidence calibration	Limited	Medium
Reasoning transparency	Strong	High
Alternative consideration	Medium	High
Information source qualification	Limited	Medium
Temporal qualification	None	N/A
Logical qualification	Medium	Medium
Uncertainty decomposition	Limited	Low

## Key Improvement Recommendations

1. Include explicit statements about knowledge limitations, particularly regarding hardware-specific compatibility issues
2. Provide specific error recognition and troubleshooting guidance for common CUDA errors
3. Include confidence calibration statements when giving version-specific recommendations
4. Add temporal qualification about software versions changing over time
5. Incorporate more uncertainty decomposition by explaining the various factors that could lead to incompatibility issues