# OpenCode LLM Benchmarks: IndexNow & Migration Mapping

This document provides a technical comparison of various Large Language Models (LLMs) evaluated using OpenCode. Testing focused on agentic workflow performance across two distinct domains: Model Context Protocol (MCP) implementation and structured data transformation.

## Executive Summary

The following table summarises model performance across both test tasks.

**Metric Definitions:**

- **MCP Dev (Py/TS):** A 'Pass' indicates the model produced a functional MCP server in either Python or TypeScript, adhering to the Model Context Protocol specification, including correct tool/resource definitions and transport handling.
- **Migration Map (% errors):** The error rate calculated as `(failed sources ÷ 80 expected)`, capped at 100%. A lower percentage indicates higher reliability.
- **Dash (—):** Indicates the model was not tested for that specific task.

| Model | MCP Dev (Py/TS) | Migration Map (% errors) |
|---|---|---|
| Qwen 3.5 27b Q3_XXS | Pass | 5.0% |
| Gemma 4 26B IQ4_XS | Pass | 6.3% |
| Nemotron 3 Super 120B IQ3_XXS (llama.cpp) | Pass | 6.3% |
| minimax-m2.5-free (OpenCode Zen) | Pass | 6.3% |
| Gemma 4 31B IQ3_XXS | Pass | 7.5% |
| Qwen3-Coder-Next UD-IQ4_XS (llama.cpp) | Pass | 8.8% |

| Model | MCP Dev (Py/TS) | Migration Map (% errors) |
|---|---|---|
| Nemotron 3 (OpenCode Zen) | Pass | 8.8% |
| Qwen 3.5 27b Q3_M | Pass | 10.0% |
| Bigpicle (OpenCode Zen) | Pass | 12.5% |
| Qwen 3.6-plus-free (OpenCode Zen) | Pass | 16.3% |
| Qwen 3.6 UD-IQ4_XS (llama.cpp) | Pass | 45.0% |
| mimo-v2-flash-free (OpenCode Zen) | Pass | 53.8% |
| Qwen 3.5 35b IQ3_S | Pass | 65.0% |
| Qwen 3.5 122B IQ3_S | Pass | 80.0% |
| Qwen 3.5 122B IQ3_XXS | Pass | 90.0% |
| Qwen 3.5 35b IQ4_XS | Pass | 98.8% |
| Qwen 3.6 35b UD-IQ3_XXS | Pass | 98.8% |
| GLM-4.7 Flash IQ4_XS | Pass | 100% |
| GLM-4.7 Flash REAP 23B IQ4_XS | Pass | 100% |
| Qwen3.5 27B IQ3_XXS Bart. | Pass | 100% |
| GPT-OSS 20b (high thinking) | Pass | — |
| Nemotron Cascade 2 30B IQ4_XS | Fail | 96.3% |
| devstral-small-2:24b | Fail | — |
| GPT-OSS 20b (default) | Fail | — |
| Qwen 3 14b | Fail | — |
| qwen3-coder:30b | Fail | — |
| qwen3.5:9b | Fail | — |

| Model | MCP Dev (Py/TS) | Migration Map (% errors) |
|---|---|---|
| qwen3.5:9b-q8_0 | Fail | — |

# Methodology

## Task 1: MCP Service Development (Python & TypeScript)

Models were prompted to architect and implement a Model Context Protocol (MCP) server. The task required creating a functional service in either Python or TypeScript capable of exposing specific tools and resources to an MCP client. Success was measured by protocol compliance, correct dependency management (e.g., `package.json` or `requirements.txt`), and passing unit tests for tool execution.

## Task 2: Website Migration Mapping

Models were tasked with generating a mapping from legacy blog URL formats (e.g., `/post/2024/10/slug/`) to new topic cluster formats. **Constraints:**

1. The slug (final path segment) must remain identical.
2. Target URLs must use the new cluster path, not the old `/post/` structure.
3. All 80 expected source URLs must be accounted for.

## Infrastructure & Resources

- **Local Hosting:** Most models were run via Ollama or llama.cpp.
- **Cloud/Zen:** Large models (e.g., Bigpicle) were accessed via OpenCode Zen.
- **Hardware Context:** Benchmarks were performed on a 16GB VRAM setup. For raw throughput data, see 16 GB VRAM LLM benchmarks.

# High-Level Recommendations

## ? Recommended for Local Use

- **Qwen 3.5 27b (Q3_XXS) on llama.cpp:** The primary choice for local OpenCode sessions. Delivered a complete, functional MCP service with 8/8 passing tests and 34 tokens/sec on 16GB VRAM.

- **Gemma 4 26B (IQ4_XS):** Strong performer; includes helpful implementation of configuration schemas.

### ?? Use with Caution (Validation Required)

- **Qwen 3.5 35b (llama.cpp):** Excellent for agentic coding, but failed significantly on structured migration tasks (slug errors and path collapses).
- **GPT-OSS 20b (High Thinking Mode):** Only viable when 'High Thinking' is enabled; default mode fails to progress past dead-end web fetches.
- **Bigpicle (OpenCode Zen):** Extremely fast and demonstrates superior research capabilities (using Exa Code Search) to understand protocol specifications.

### ? Avoid for Agentic Coding

- **GPT-OSS 20b (Default), Qwen 3 14b, Devstral-small-2:** These models exhibit high hallucination rates or stall during tool-calling/web-fetching tasks.

---

## Detailed Model Analysis

### MCP Development Task: Detailed Analysis

#### Qwen Series

- **Qwen 3.5 27b (IQ3_XXS): Top Performer.** Highly efficient; produced full documentation and tests for the MCP service.
- **Qwen 3.5 27b (Bartowsky Quant):** Shows significant variance compared to Unsloth quants; failed migration tasks due to category-path collapse.
- **Qwen 3.6 (35b variants):** Performance varies wildly by quantization. `IQ4_XS` is good for coding; `IQ3_XXS` suffers from massive category-path collapse in structured tasks.
- **Qwen3-Coder-Next:** Very fast (53s) and produces clean code on the first attempt, though lacks automated README/test generation.

#### Gemma Series

- **Gemma 4 26B:** Solid MCP service generation with built-in configuration support.
- **Gemma 4 31B:** Functional, but lacks the advanced features/documentation of the 26B variant.

#### Nemotron Series

- **NVIDIA-Nemotron-3-Super-120B:** Capable of high-quality output but requires manual intervention to write files/compile. Extremely resource-intensive.
- **Nemotron Cascade 2 30B:** Generally fails to produce functional code without multiple corrective prompts.

### GLM Series

- **GLM-4.7 Flash:** Highly efficient and fast (sub-60s), but requires a second prompt to complete the build/compilation.
- **GLM-4.7 Flash REAP 23B:** The most comprehensive default output (includes unit tests, config files, and multiple documentation files).

### Other Notable Results

- **GPT-OSS 20b (High Thinking):** A meaningfully different story from the default mode. With high thinking enabled, the model recovered from dead-end fetches and managed to build a complete, working MCP server with proper tool and resource definitions.
- **Bigpicle (big-pickle):** The standout performer. It used Exa Code Search to research the MCP specification before coding, ensuring correct tool implementation on the first try.
- **qwen3.5:9b:** Complete failure. It went through the thinking process but never actually implemented the MCP server logic or called any tools.
- **Qwen 3 14b:** Classic hallucination. Fabricated incorrect API/Protocol details rather than admitting it couldn't find the specification.

---

## Migration Mapping: Statistical Deep-Dive

The migration task exposed a critical failure mode in many models: **Slug Drift** and **Category Collapse**.

### Key Failure Modes Identified:

1. **2022 Prefix Stripping:** Almost all models failed to preserve numeric prefixes in older slugs (e.g., converting `/06-git-cheatsheet/` to `/git-cheatsheet/`).
2. **Category-Path Collapse:** Large models (Qwen 3.5 35b/122b and Bartowsky quants) frequently collapsed individual page URLs into their parent category URLs.

3. **SEO Rewriting:** Some models (Qwen 3.6 35b) prioritised generating "clean" SEO slugs over preserving the required source slugs.

## Migration Error Table (Detailed)

| Model | Lines | Mismatches | Error Rate | Primary Failure Mode |
|---|---|---|---|---|
| Qwen 3.5 27b Q3 XXS | 80 | 4 | 5.0% | 2022 Prefix Stripping |
| Gemma 4 26B it | 81 | 5 | 6.3% | 2022 Prefix Stripping / Layout Error |
| Nemotron 3 Super 120B | 81 | 5 | 6.3% | 2022 Prefix Stripping |
| Qwen3-Coder-Next | 81 | 7 | 8.8% | Minor Slug Renaming |
| Qwen 3.6 35B UD-IQ4_XS | 81 | 36 | 45.0% | SEO Title Rewriting |
| Qwen 3.5 35b IQ4_XS | 80 | 79 | 98.8% | Category-Path Collapse |
| Qwen 3.6 35B UD-IQ3_XXS | 67 | 79 | 98.8% | Uniform Category-Path Collapse |

Generated using AI hosted in the NREE by NCIA.

---

Revision #2

Created 2026-07-03 09:16:11 UTC by Carsten

Updated 2026-07-05 14:29:57 UTC by Carsten