Benchmarking Large Language Models for Text Input in Chinese and Japanese

Phonetic-to-Character Generation and Textual Error Correction

Yuchun Zou[1] Tedd Lee[1] Xiaodi Fan[2] Jun Li[1]

[1]City University of New York (CUNY)

[2]Meta

LREC 2026 – Palma, Spain

The Challenge of Logographic Input

How IMEs Work

- Phonetic keystrokes → logographic characters
- Pinyin (Chinese), Romaji (Japanese)

The Homophone Problem

- `yi yi` → {意义, 异议, ...}
- `ta men` → {他们, 她们, 它们}
- Context needed for disambiguation

Where Traditional IMEs Fail

- Short n-gram context window
- Long-range semantic dependencies
- Noisy / mistyped phonetic input

Actual IME Candidate List



In sentence: `wo jin tian qu shanghai`
Correct: 我今天去上海
Context should push 上海 to rank 1.

Two Core Tasks Evaluated

Task 1: Phonetic-to-Character Generation

- **Word-level:** single word conversion
  `shang hai` → 上海
- **Sentence-level:** full phonetic sentence to logographic text

Chinese (Pinyin) & Japanese (Romaji/Kana)

Task 2: Textual Error Correction

- **Phonetic correction:** fix noisy input
  `woo jintna` → `wo jintian`
- **Character correction:** fix wrong output characters
  的/地/得 confusion

Evaluated for **Chinese** and **Japanese** | Japanese word generation split by script: Kanji / Hiragana / Katakana

Models Benchmarked

LLMs (17 models total)

- *Online proprietary:* GPT-4, GPT-4o, GPT-4o-mini, Gemini 2.0/2.5 Flash, Claude 3.5 Haiku
- *Offline open-weight:* Llama3-8B, Llama3.2-3B, Mistral-7B, GLM4-9B
- *Chinese-specific:* Qwen2-7B, Baichuan2-7B
- *Japanese-specific:* RakutenAI-7B, Nekomata-7B, Swallow-8B

Traditional Baselines

- macOS native Pinyin IME
- macOS native Romaji IME
- Pinyin2Hanzi (HMM & DAG)
- pycorrector
- OpenNMT (seq2seq)

Metrics

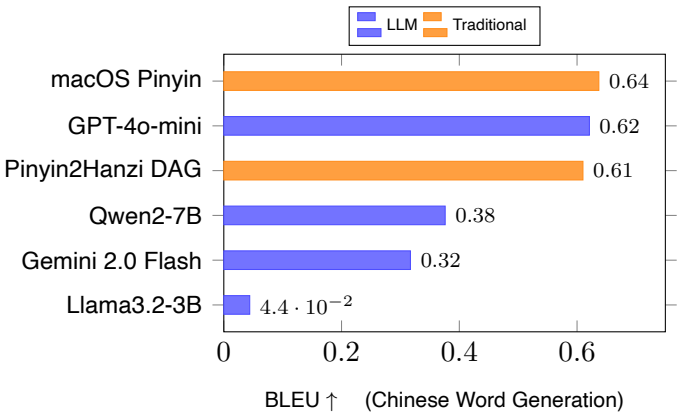
- Accuracy: SimHash, BERTScore, BLEU, CER
- Speed: Completion Time, TTFT, TPS

Datasets

Language	Task	Source	Size
Chinese	Word Generation	Phrase Pinyin Data	7,260 words
Chinese	Sentence Generation	SIGHAN 2015 Bake-off Corpus	4,440 sentences
Chinese	Phonetic Error Corr.	Custom (injected errors)	3,502 sentences
Chinese	Character Error Corr.	SIGHAN 2015 (annotated)	included above
Japanese	Word Generation	Yomichan JLPT Vocabulary	multi-level vocab
Japanese	Sentence Generation	Snow Simplified Corpus	$>$10,000 sentences
Japanese	Phonetic Error Corr.	Custom (injected errors)	$>$10,000 sentences
Japanese	Character Error Corr.	Kyoto Univ. JWTD	pre-annotated

Custom error datasets: errors injected by substituting, deleting, or inserting phonetic keystrokes.
All datasets, evaluation scripts, and results are **publicly released**.

Finding 1: Word Generation — Traditional Baselines Win



Key Takeaway

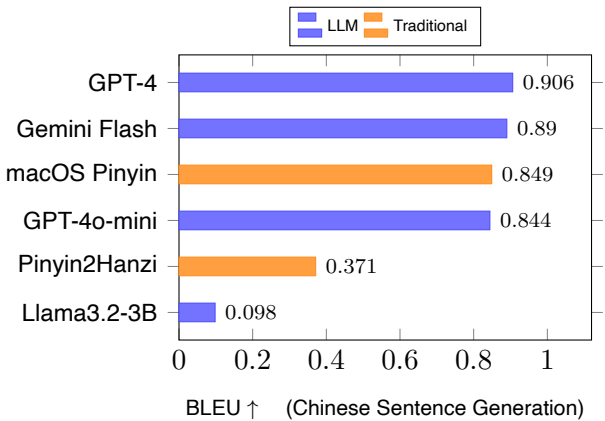
Traditional systems win on **isolated word conversion** — speed and accuracy.

Japanese Word Generation (SimHash)

- Gemini 2.0 Flash: **0.831**
- GPT-4o-mini: 0.791
- macOS Romaji: 0.736
- Llama3.2-3B: 0.618

Word-level conversion is largely a lookup problem; n-gram statistics suffice.

Finding 2: Sentence Generation — LLMs Dominate



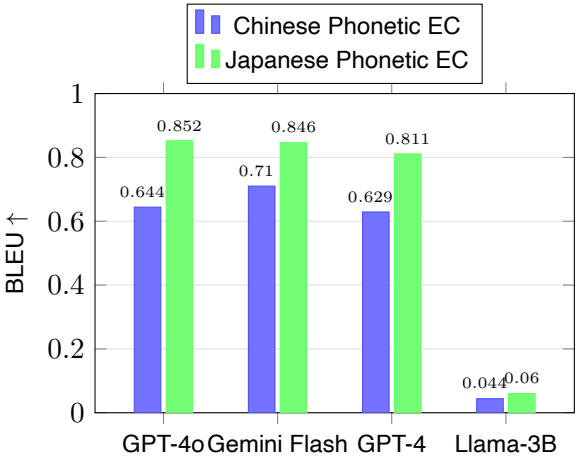
Japanese Sentence Generation (BLEU)

- Gemini 2.0 Flash: **0.867**
- macOS Romaji: 0.849
- GPT-4: 0.827
- Swallow-8B (Ja-specific): 0.448
- Nekomata-7B (Ja-specific): **0.083**

Two Key Findings

- Pinyin2Hanzi collapses: BLEU **0.371** (was 0.610 at word level)
- Language-specific LLMs **fail**: Nekomata 0.083 vs Gemini 0.867

Finding 3: Phonetic Error Correction — LLMs Fill a Critical Gap

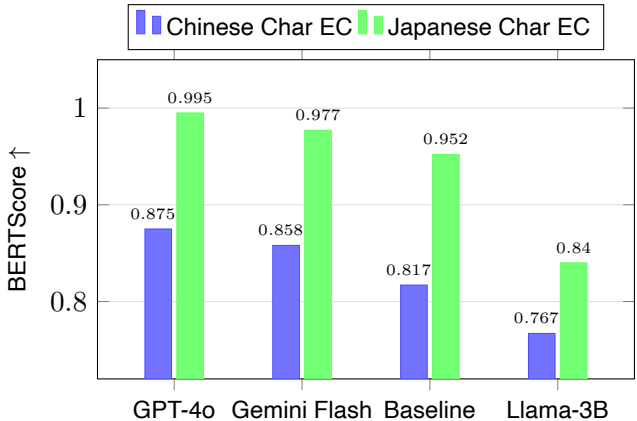


No Traditional Baseline

Standard IMEs **cannot** operate on noisy phonetic input.
LLMs are the **only viable solution**.

- Gemini Flash leads on Chinese EC (BLEU **0.710**)
- GPT-4o leads on Japanese EC (BLEU **0.852**)
- Llama-3B completely fails: BLEU **0.044 / 0.060**
  (insufficient capacity to infer user intent from corrupted input)

Finding 4: Character Error Correction — Near-Perfect Accuracy



Baseline = pycorrector (Chinese) / OpenNMT (Japanese)

Standout: GPT-4o on Japanese

BERTScore **0.995**
CER **0.013** — near-perfect

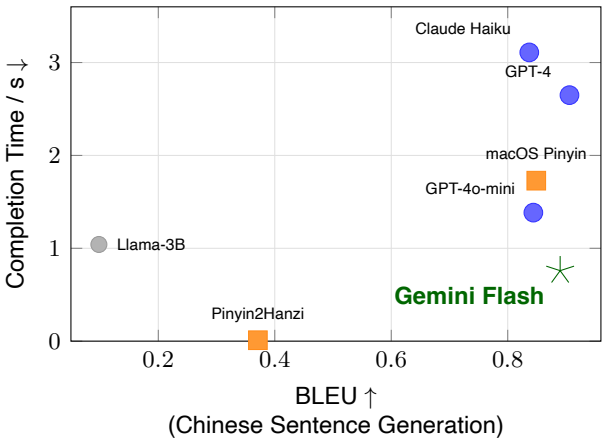
vs. OpenNMT (specialized)

BERTScore 0.952, CER 0.085
Time: **4.2s** per sentence

Gemini 2.0 Flash

BERTScore 0.977
Time: **1.1s** — 4× faster

The Accuracy–Latency Trade-Off



IME constraint: near-zero latency

- GPT-4: best BLEU (0.906) but **2.6s** — too slow
- Gemini Flash: BLEU **0.890** in **0.76s**
  Pareto-optimal point
- Llama-3B: fast but BLEU **0.098** — unusable
- macOS Pinyin: BLEU 0.849 but no error handling

No single model satisfies both constraints ⇒ **hybrid architecture**.

Conclusion: The Future is Hybrid

Summary of Findings

1. **Word-level:** Traditional baselines win
2. **Sentence-level:** LLMs dominate; n-gram models collapse
3. **Error correction:** LLMs are the only viable solution
4. **Best overall:** Gemini 2.0 Flash (accuracy + speed)
5. **Surprise:** Language-specific models don't beat frontier LLMs

Hybrid IME Architecture

- **Fast local path:** Traditional algorithm for short word inputs $\rightarrow$ zero-latency typing
- **Cloud LLM path:** Route long sentences and error correction to a high-fidelity LLM in background
- Dynamic routing by input length and complexity

All datasets, evaluation scripts, and results are **publicly available**:

`https://github.com/junli-cuny/llm-ime-benchmark`

Thank You & Q&A

Thank You!

- **Paper:** *Benchmarking LLMs for Text Input in Chinese and Japanese*
- **Authors:** Yuchun Zou, Tedd Lee, Xiaodi Fan, Jun Li
- **Contact:** jun.li@qc.cuny.edu

Best Model per Task

Task	Best Model
Word generation	macOS IME
Sentence generation	Gemini 2.0 Flash
Phonetic EC	Gemini 2.0 Flash
Character EC (Zh)	GPT-4o
Character EC (Ja)	GPT-4o

Thank you to the LREC 2026 organizers and attendees!