NE-Agent: Bringing Agentic AI to Northeast India's Low-Resource Languages

Badal Nyalang
MWire Labs
Shillong, Meghalaya, India
nyalang@mwirelabs.com

Abstract

Northeast India is home to over 220 indigenous languages, yet none are served by existing agentic AI systems. **NE-Agent** addresses this gap: a terminal-based agentic system that integrates the NE-Stack, a suite of purpose-built language AI components for Northeast India, into a unified conversational pipeline that dynamically selects between retrieval, translation, and transcription tools based on the user's query. The system combines language identification across 11 languages (NE-LID), multilingual dense retrieval (NE-Embed, a LaBSE-based model fine-tuned for 10 Northeast Indian languages), Khasi-to-English translation (fine-tuned NLLB), speech transcription across eight languages (NE-ASR), and generation via a locally-hosted open-source LLM, requiring no API keys or cloud infrastructure. A lightweight LLM-based tool router dispatches each query to the appropriate tool, distinguishing NE-Agent from purely retrieval-augmented systems. NE-Agent is open-source, pip-installable (`pip install ne-agent`), and ships with a multilingual corpus spanning four languages. To our knowledge, it is the first publicly available agentic system built on indigenous Northeast Indian language AI infrastructure: https://github.com/MWirelabs/ne-agent.

1 Introduction

Northeast India is home to over 220 indigenous languages spoken across eight states (Census of India, 2011). Despite this extraordinary linguistic diversity, the region remains severely underserved by mainstream NLP research and commercial AI systems. Major language models and retrieval systems either lack coverage of Northeast Indian languages entirely or treat the region as a monolith, failing to distinguish between typologically distinct language families such as Tibeto-Burman (Garo, Bodo, Mizo), Austroasiatic (Khasi), and Indo-Aryan (Assamese, Nagamese). Even large-scale Indian language initiatives such as IndicNLP-Suite (Kakwani et al., 2020) and IndicTrans2 (Gala et al., 2023) cover only the scheduled languages of the region, leaving the majority of Northeast Indian languages without foundational resources.

Recent years have seen foundational resources emerge for the region, including language identification, multilingual encoders, embedding models, and multilingual speech recognition (Section 2.1). However, these components have largely remained siloed research artifacts without integration into usable, tool-using end-to-end systems.

The rise of agentic AI systems, which combine retrieval, reasoning, and generation in a tool-using loop, presents an opportunity to make language AI infrastructure accessible to non-technical users in underserved communities. Systems such as LangChain (Chase, 2022), LlamaIndex (Liu, 2022), and AutoGPT (Significant Gravitas, 2023) have demonstrated the potential of such architectures, but none address low-resource language contexts where standard tokenizers, embeddings, and language models fail.

NE-Agent is a terminal-based agentic system that addresses this gap. It chains NE-LID for language identification, an LLM-based tool router, NE-Embed for dense retrieval, a fine-tuned NLLB model for Khasi-English translation, NE-ASR for speech transcription, FAISS (Johnson et al., 2021) for vector indexing, and a locally-deployed open-source language model for generation and routing. The system operates entirely offline after setup, requires no API subscriptions, and is designed to run on consumer-grade CPU or single-GPU hardware, in line with the broader shift toward edge deployment of language models for latency, privacy, and connectivity reasons (Zheng et al., 2025).

Our contributions are as follows:

- An end-to-end agentic pipeline for Northeast Indian languages that dynamically routes be-

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 25–32
November 6-10, 2026 ©2026 Association for Computational Linguistics

tween retrieval, translation, and transcription tools, rather than executing a fixed retrieval-only path.

- NE-Agent, a publicly available open-source system with a pip package (`ne-agent`) and a multilingual corpus of 2,000 sentences across four languages.
- NE-Embed, a multilingual embedding model developed to meet the system's retrieval requirements where no existing model sufficed (Section 3.3), integrated with a fine-tuned NLLB translation model and NE-ASR into a single deployable agent.
- A terminal user interface designed for accessibility with a distinctive visual identity.

2 Background and Related Work

2.1 Low-Resource NLP for Northeast India

NLP research on Northeast Indian languages has accelerated in recent years, producing language identification across 11 languages (Nyalang, 2026b), multilingual encoder models for nine languages of the region (Nyalang, 2026a), machine translation for Kokborok, Garo, and Khasi, and multilingual speech recognition spanning eight languages. Despite this progress, prior work has largely stopped at publishing individual components rather than integrating them into a user-facing system capable of deciding, at inference time, which component a given query requires. Comparable integration efforts exist in other low-resource contexts: Ismail et al. (2025) combine RAG, translation, and text-to-speech for drug information access in local African languages, demonstrating the practical value of composing low-resource components into deployed systems. NE-Agent brings this integration step to Northeast India.

2.2 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) grounds language model outputs in retrieved documents, with dense bi-encoder retrieval (Karpukhin et al., 2020) over FAISS indexes (Johnson et al., 2021) as the standard approach, and hybrid dense-sparse retrieval (Arivazhagan et al., 2023) offering gains for morphologically rich languages. In multilingual settings, cross-lingual retrieval can help low-resource languages by drawing on high-resource evidence (Nie et al., 2023), but generation models utilize retrieved passages inconsistently across languages (Qi et al., 2025), motivating translation-based pipelines (Ranaldi et al., 2026). NE-Agent takes a complementary approach, filtering retrieved documents to the detected query language to guarantee monolingual context. More broadly, agentic RAG extends the fixed retrieve-then-generate routine with active tool selection (Deng et al., 2026), the paradigm NE-Agent adopts through its tool router.

	LangChain	LlamaIndex	NE-Agent
Offline by default	×	×	✓
No API keys	×	×	✓
NE language support	×	×	11 langs
Routing	chains	pipelines	NE tool router

Table 1: Comparison with general-purpose agentic frameworks. LangChain and LlamaIndex can be configured with local model backends, but their default integrations assume hosted APIs and neither provides components for Northeast Indian languages. NE-Agent ships offline-first with purpose-built NE-Stack components.

2.3 Agentic NLP Systems

Agentic NLP systems combine language models with tool use, memory, and multi-step reasoning (Wang et al., 2024), from interleaved reasoning and action (Yao et al., 2023) to tool-augmented reasoning frameworks (Wu et al., 2025) and automated workflow optimization (Wang et al., 2025). LangChain (Chase, 2022) and LlamaIndex (Liu, 2022) are the dominant frameworks for building such systems, but they assume high-resource language support and hosted API access, limiting their applicability in low-resource and offline settings. Table 1 summarizes the differences most relevant to our deployment context.

Agentic approaches to machine translation have also emerged recently, pairing off-the-shelf LLMs with self-refine workflows (Grubišić and Korencic, 2025) and using LLM agents for document-level translation with discourse modeling (Dutta et al., 2025). NE-Agent differs from these in its use of purpose-built indigenous language components across three distinct tool modalities (retrieval, translation, transcription) rather than general-purpose multilingual models applied to a single task.

3 System Architecture

NE-Agent follows a five-stage pipeline: language identification, tool routing, tool-specific process-

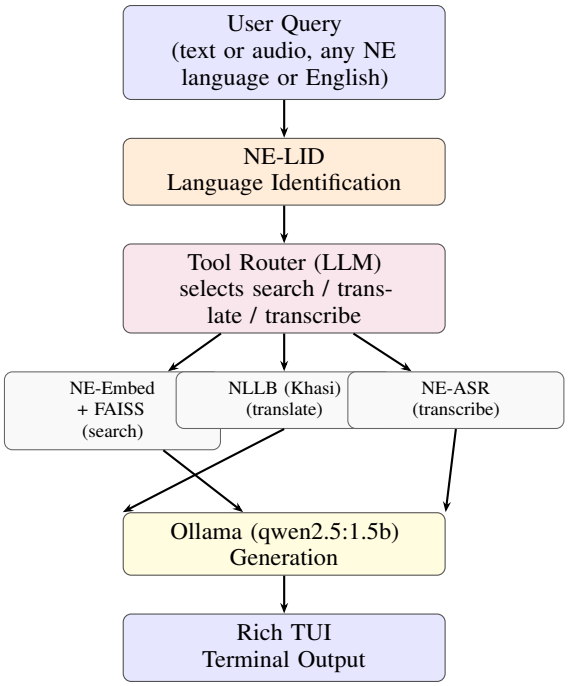


Figure 1: NE-Agent system architecture (v0.2). The tool router inspects the language-identified query and dispatches it to one of three tools; retrieval and transcription outputs are optionally passed to the local LLM for response synthesis, while translation outputs are returned directly.

ing (retrieval, translation, or transcription), context-grounded generation where applicable, and terminal rendering. Figure 1 illustrates the full system architecture, and the following subsections describe each stage in pipeline order.

3.1 Language Identification: NE-LID

All user inputs are first passed to NE-LID, a fastText character n-gram language identification model trained on 11 Northeast Indian languages: Assamese, Bodo, English, Garo, Hindi, Khasi, Kokborok, Meitei, Mizo, Nagamese, and Nyishi. NE-LID achieves 99.09% macro-averaged accuracy and produces a language label and confidence score for each input. The detected language is used downstream both to filter retrieved documents and to select the appropriate translation path.

In our evaluation on live queries, NE-LID correctly identified Khasi input with confidence 1.0000 and Garo input with confidence 0.9999, demonstrating reliable performance on natural user inputs.

3.2 Tool Router

A key addition distinguishing NE-Agent from a purely retrieval-augmented system is the tool router: a lightweight prompting layer that uses the same locally-hosted `qwen2.5:1.5b` model to select among three available tools (`search`, `translate`, `transcribe`) based on the query text. The router receives the raw query and a short instruction listing the available tools, and returns a single tool name, which is validated against the fixed tool set before dispatch; an unrecognized or malformed response defaults to `search`. This design keeps routing decisions local and dependency-free, avoiding the need for a separate classifier or external API.

In practice, queries containing explicit translation cues (e.g., "translate this: ...") are routed to the translation tool, audio input is routed directly to transcription without invoking the router, and all other text queries default to retrieval-augmented search. This lets the same pipeline entry point handle heterogeneous request types without requiring the user to specify a mode.

3.3 Dense Retrieval: NE-Embed

Retrieval in NE-Agent is powered by NE-Embed (MWire Labs, 2026), a multilingual dense embedding model developed specifically to meet the retrieval requirements of this system: at the time of development, no publicly available embedding model produced reliable representations for languages such as Garo, Khasi, Nyishi, or Kokborok, making off-the-shelf dense retrieval unusable for our target languages.

NE-Embed is fine-tuned from `sentence-transformers/LaBSE` (Feng et al., 2022) using MultipleNegativesRankingLoss (Henderson et al., 2017) on 201,738 English-to-NE-language parallel pairs across 10 languages, combining AI4Bharat BPCC (Gala et al., 2023) for Assamese, Bodo, and Meitei with the MWire Labs parallel corpus for Garo, Khasi, Mizo, Nyishi, Kokborok, Pnar, and Nagamese. The model produces 768-dimensional L2-normalized embeddings with a maximum sequence length of 128 tokens.

Table 2 reports retrieval performance (R@1) and cross-language retrieval interference (CLRI) against the LaBSE zero-shot baseline, where CLRI is the percentage of queries whose top-1 retrieved document belongs to a different language than the query. NE-Embed yields substantial gains across all languages, most dramatically for Garo (+77.6 R@1), Khasi (+67.0), and Nyishi (+64.8), while Meitei remains the most challenging (34.2 R@1),

Language	LaBSE (zs)		NE-Embed v1	
	R@1	CLRI%	R@1	CLRI%
Assamese	95.6	1.8	97.4	4.6
Bodo	55.8	61.0	99.8	3.0
Garo	13.2	88.8	90.8	3.0
Khasi	28.6	65.0	95.6	3.4
Mizo	46.6	58.4	91.8	9.4
Meitei	13.6	90.8	34.2	19.8
Nyishi	10.2	71.0	75.0	17.4
Pnar	27.2	79.6	86.2	8.0
Kokborok	26.4	63.8	71.6	11.8
Nagamese	77.0	17.8	88.0	8.4

Table 2: Retrieval performance of LaBSE zero-shot (zs) vs NE-Embed v1. R@1 and CLRI evaluated on a 6,000-vector mixed-language FAISS index (500 per language). CLRI is the percentage of queries where the top-1 result belongs to the wrong language.

likely due to underrepresentation of the Meitei Mayek script in LaBSE pretraining. Fine-tuning also mitigates an *Assamese attractor effect*, where the better-pretrained Assamese otherwise dominates mixed-index retrieval.

In NE-Agent, NE-Embed encodes user queries and corpus documents into a shared 768-dimensional space. Retrieval is performed via inner product search over a FAISS `IndexFlatIP` index, equivalent to cosine similarity since embeddings are L2-normalized. Retrieved documents are filtered to match the language detected by NE-LID before being passed to the language model, ensuring monolingual context for generation.

3.4 Translation Tool

When the router selects `translate`, NE-Agent invokes a fine-tuned NLLB model for Khasi-to-English translation. The model is fine-tuned on a Khasi-English parallel corpus and exposes a single language code (`kha_Latn`) for the source side. Given a query, the tool strips routing cue phrases (e.g., "translate this:"), encodes the remaining text, and generates an English translation directly, without passing the output through the retrieval or generation stages. This tool currently supports Khasi only; extending coverage to Garo, Kokborok, and other NE-Stack languages is discussed in Section 6.

A separately released multilingual translation model, KrenTransV0-1, is publicly available and covers additional NE-Stack languages, including a first-place primary submission on Khasi and Kokborok at WMT 2026.

3.5 Transcription Tool

When the input is audio rather than text, NE-Agent bypasses the router and invokes NE-ASR, a `whisper-medium` model fine-tuned on 150,483 training samples across eight Northeast Indian languages (Khasi, Garo, Mizo, Nagamese, Kokborok, Assamese, Chakma, Wancho). Audio is resampled to 16kHz where necessary and transcribed using a shared decoder language token across all non-Assamese languages. The resulting transcription is returned as the agent's answer; in future versions this may be chained into the retrieval or translation tools for downstream processing.

3.6 Generation: Local Language Model

For `search` queries, NE-Agent uses Ollama (Ollama Team, 2023) to serve an open-source language model locally. The default model is `qwen2.5:1.5b` (Qwen Team, Alibaba Cloud, 2025), a 1.5 billion parameter instruction-tuned model that fits within 2GB of memory, enabling CPU deployment. The same model instance is reused for tool routing (Section 3.2), avoiding the overhead of loading a separate classifier. The system constructs a structured prompt containing the detected language, retrieved documents with language tags, and the user query. No API key or internet connection is required at inference time.

3.7 Terminal User Interface

NE-Agent presents a Rich (Willmott, 2020) terminal interface with a distinctive visual identity: an ASCII block-letter header, a multilingual greeting in four Northeast Indian languages, and a status line listing active components. Each response displays the detected language and confidence (or the invoked tool name), the number of retrieved snippets where applicable, and the answer in a bordered panel.

4 Implementation

4.1 Corpus and Indexing

The default corpus distributed with NE-Agent contains 500 monolingual sentences per language for Assamese, Garo, Khasi, and Mizo, sampled from the MWire Labs corpus and AI4Bharat BPCC (Gala et al., 2023). All corpora are CC-BY-4.0 and were deduplicated prior to training. Documents are encoded offline using NE-Embed and indexed in a FAISS `IndexFlatIP` structure, yielding a 2,000-vector index. Users may replace or



Figure 2: NE-Agent terminal interface (v0.2), showing tool routing across search, translate, and transcribe queries in a live session.

extend the corpus with domain-specific documents.

Retrieval proceeds as follows: the query is encoded with NE-Embed, the top-50 nearest neighbors are retrieved from FAISS, and results are filtered to retain only documents matching the NE-LID detected language, returning the top-3 matching documents as context.

4.2 Package and Distribution

NE-Agent is distributed as a pip package (`pip install ne-agent`, version 0.2.3) requiring Python 3.9 or later; Ollama must be installed separately and a model pulled before use. The package structure separates language identification, retrieval, translation and transcription tools, generation and routing, pipeline orchestration, and terminal rendering into individual modules, enabling modular replacement of any component. The CLI additionally exposes an `-audio` flag for direct transcription outside the interactive query loop. Full installation instructions are provided in the repository README. All released components, including the translation model, are distributed under the CC-BY-4.0 license.

5 Evaluation

We evaluate NE-Agent along the stages of its pipeline: language identification, retrieval, tool routing, and end-to-end behavior including latency.

5.1 Language Identification

We evaluated NE-LID on a set of natural user queries in four languages. Table 3 reports detection accuracy and confidence scores on representative inputs.

Lang.	Query (excerpt)	Pred.	Conf.
Khasi	*Kumno ngi tip u ia u?*	khasi	1.0000
Garo	*Angni manda?*	garo	0.9991
Garo	*Maina nang·ni namnik...*	garo	0.9999
Mizo	*Eng nge nia?*	nyishi	0.9953

Table 3: NE-LID performance on live user queries in NE-Agent. The Mizo query was misidentified as Nyishi, a known limitation of the model.

Query type	Query (excerpt)	Routed tool
Search	*Kumno phi long?*	search
Translate	*translate this: Kumno phi long?*	translate
Search	*Angni manda?*	search

Table 4: Tool router decisions on representative queries. The router correctly distinguishes explicit translation requests from general queries, defaulting to search in the absence of translation cues.

Three of four queries were correctly identified with confidence above 0.999. The Mizo query was misidentified as Nyishi, a known limitation of character n-gram models on closely related Tibeto-Burman languages with shared phonological features (Nyalang, 2026b).

5.2 Retrieval Quality

Retrieval quality is measured by whether the language filter successfully retrieves same-language documents for a given query. On our four-language corpus, language-filtered retrieval returns relevant monolingual documents for all correctly identified queries. Retrieved document scores range from 0.49 to 0.61 in cosine similarity, consistent with cross-lingual retrieval behavior observed in the NE-Embed evaluation (Section 3.3).

5.3 Tool Routing Accuracy

We evaluated the tool router on a small set of representative queries spanning all three tools. Table 4 reports routing decisions.

The router correctly dispatched explicit translation cues to `translate` while defaulting general queries to `search`, confirming routing is not simply a fixed default. In manual testing we also observed a correctly routed and retrieved Khasi query where `qwen2.5:1.5b` nevertheless ignored the context and hallucinated, showing that correct routing and retrieval do not guarantee grounded generation at 1.5B parameters. A systematic evaluation over a larger, held-out query set is left to future work.

Tool	GPU (warm, avg)	CPU (avg)
Translate	0.68s	1.76s
Transcribe	0.43s (cold: 14.2s)	12.38s
Search	2.32s	n/a

Table 5: Per-tool latency averaged over three runs. Search invokes the local Ollama LLM for generation, which is not isolated from GPU/CPU device selection in the current implementation, so a directly comparable CPU-only search figure is not reported. Transcribe's GPU cold-start figure reflects first-invocation model loading; subsequent GPU invocations reflect warm-cache latency.

5.4 End-to-End Pipeline

We demonstrate the end-to-end pipeline across all three tools:

Search (Garo): *Angni manda?* NE-LID detects Garo with confidence 0.9991, the router selects `search`, retrieval returns a Garo document, and `qwen2.5:1.5b` generates a grounded response.

Translate (Khasi): *Kumno phi long?* NE-LID detects Khasi, the router selects `translate`, and the fine-tuned NLLB model returns "How are you doing?" directly.

Transcribe (Khasi, audio): A synthesized Khasi utterance intended as "Kumno ngan thung ki jhur jong nga?" is transcribed by NE-ASR as "Kumno ngan thung ki jhur jong a?", differing from the reference only in the final pronoun, consistent with NE-ASR's 16.89% test WER on Khasi.

5.4.1 Latency

We measured per-tool latency on a single RTX A4000 GPU (20GB VRAM) and on a 48-core AMD EPYC 7352 CPU, averaged over three runs per condition. Table 5 reports results.

Translate and transcribe both show the expected GPU speedup over CPU-only inference, with transcribe showing the largest relative gap (approximately 29x for warm GPU vs. CPU) due to the whisper-medium model's size. Search latency includes both retrieval and LLM-based generation via Ollama and was measured only on GPU, since isolating the search path to CPU-only would require reconfiguring the Ollama server itself rather than a Python-level device argument.

6 Limitations

First, response quality depends heavily on corpus coverage; the default 500-sentence corpus per language is sufficient for demonstration but shallow for real-world use. Second, `qwen2.5:1.5b` has limited knowledge of Northeast Indian languages and cultures and may hallucinate when retrieved context is insufficient, as observed once during evaluation (Section 3.2). Third, NE-LID misidentifies closely related languages, particularly Mizo and Nyishi. Fourth, Meitei and other non-Latin scripts may experience tokenization artifacts in generation. Fifth, translation currently supports only Khasi. Sixth, the router shares the 1.5B generation model and has been evaluated only on representative queries; misrouted queries default to search. Future work includes expanded corpora, a larger labeled router evaluation, additional retrieval metrics against multilingual baselines, groundedness evaluation, and isolated CPU-only search latency.

7 Conclusion

We presented NE-Agent, an open-source terminal-based agentic system for low-resource Northeast Indian languages, integrating language identification, an LLM-based tool router, dense retrieval, translation, transcription, and local generation into a complete offline pipeline that dynamically selects among tools rather than executing a single fixed path. Publicly available via GitHub and PyPI and designed for consumer hardware, NE-Agent demonstrates that purpose-built regional language AI infrastructure can be composed into practical, tool-using end-to-end systems for the linguistically rich Northeast Indian region.

Ethics Statement

Corpora were collected with community-informed consent from speakers of Northeast Indian languages. All models and data are released under CC-BY-4.0.

Acknowledgments

The author thanks the team at MWire Labs for their support in building the individual models that make up the NE-Stack, and the local language communities of Northeast India whose participation made this work possible.

References

Manoj Ghuhan Arivazhagan, Lan Liu, Peng Qi, Xinchi Chen, William Yang Wang, and Zhiheng Huang. 2023. Hybrid hierarchical retrieval for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL 2023*,

pages 10680–10689, Toronto, Canada. Association for Computational Linguistics.

Census of India. 2011. Census of India 2011: Language atlas.

Harrison Chase. 2022. LangChain.

Jingwen Deng, Jihao Huang, Zhen Hao Wong, Hao Liang, Quanqing Xu, Bin Cui, and Wentao Zhang. 2026. Data-centric perspectives on agentic retrieval-augmented generation: A survey. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 1570–1588, San Diego, California, United States. Association for Computational Linguistics.

Himanshu Dutta, Sunny Manchanda, Prakhar Bapat, Meva Ram Gurjar, and Pushpak Bhattacharyya. 2025. GRAFT: A graph-based flow-aware agentic framework for document-level machine translation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 2405–2428, Suzhou, China. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Jay Gala, Pranjal A. Chitale, Raghavan AK, Varun Gumma, Sumanth Doddapaneni, Aswanth Kumar M, Janki Atul Nawale, Anupama Sujatha, Ratish Puduppully, Vivek Raghavan, Pratyush Kumar, Mitesh M. Khapra, Raj Dabre, and Anoop Kunchukuttan. 2023. IndicTrans2: Towards high-quality and accessible machine translation models for all 22 scheduled Indian languages. *Transactions on Machine Learning Research*.

Ivan Grubišić and Damir Korencic. 2025. IRB-MT at WMT25 translation task: A simple agentic system using an off-the-shelf LLM. In *Proceedings of the Tenth Conference on Machine Translation*, pages 753–764, Suzhou, China. Association for Computational Linguistics.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, Laszlo Lukacs, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. In *arXiv preprint arXiv:1705.00652*.

Ahmad Ibrahim Ismail, Bashirudeen Opeyemi Ibrahim, Olubayo Adekanmbi, and Ife Adebara. 2025. Retrieval-augmented generation meets local languages for improved drug information access and comprehension. In *Proceedings of the Sixth Workshop on African Natural Language Processing (AfricaNLP 2025)*, pages 108—-114, Vienna, Austria. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961, Online. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Jerry Liu. 2022. LlamaIndex.

MWire Labs. 2026. NE-Embed: A fine-tuned multilingual text embedding model for low-resource Northeast Indian languages. HuggingFace model release.

Ercong Nie, Sheng Liang, Helmut Schmid, and Hinrich Schütze. 2023. Cross-lingual retrieval augmented prompt for low-resource languages. In *Findings of the Association for Computational Linguistics: ACL 2023*, page 8320–8340, Toronto, Canada. Association for Computational Linguistics.

Badal Nyalang. 2026a. NE-BERT: A multilingual language model for nine Northeast Indian languages. In *Proceedings of the Second Workshop on Language Models for Low-Resource Languages (LoResLM 2026)*, pages 1–12, Rabat, Morocco. Association for Computational Linguistics.

Badal Nyalang. 2026b. NE-LID: A fast and accurate language identification system for Northeast Indian languages. In *Proceedings of the 8th Workshop on Indian Language Data: Resources and Evaluation*, pages 104–108, Palma, Mallorca, Spain. European Language Resources Association (ELRA).

Ollama Team. 2023. Ollama: Run large language models locally.

Jirui Qi, Raquel Fernández, and Arianna Bisazza. 2025. On the consistency of multilingual context utilization in retrieval-augmented generation. In *Proceedings of the 5th Workshop on Multilingual Representation Learning (MRL 2025)*, Suzhou, China. Association for Computational Linguistics.

Qwen Team, Alibaba Cloud. 2025. Qwen2.5: A large language model series.

Leonardo Ranaldi, Barry Haddow, and Alexandra Birch. 2026. Multilingual retrieval-augmented generation for knowledge-intensive question answering task. In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 697–716. Association for Computational Linguistics.

Significant Gravitas. 2023. Auto-GPT: An autonomous GPT-4 experiment.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024. A survey on large language model-based autonomous agents. *Frontiers of Computer Science*, 18(6).

Yingxu Wang, Siwei Liu, Jinyuan Fang, and Zaiqiao Meng. 2025. EvoAgentX: An automated framework for evolving agentic workflows. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 643–655, Suzhou, China. Association for Computational Linguistics.

Will Willmott. 2020. Rich: Rich text and beautiful formatting in the terminal.

Junde Wu, Jiayuan Zhu, Yuyuan Liu, Min Xu, and Yueming Jin. 2025. Agentic reasoning: A streamlined framework for enhancing LLM reasoning with agentic tools. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28489–28503, Vienna, Austria. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Yue Zheng, Yuhao Chen, Bin Qian, Xiufang Shi, Yuanchao Shu, and Jiming Chen. 2025. A review on edge large language models: Design, execution, and applications. *ACM Computing Surveys*, 57(8):1–35.

A Installation and Usage

Installation:

```
pip install ne-agent
# Install Ollama: https://ollama.com
ollama pull qwen2.5:1.5b
ne-agent
```

Demonstration Video: https://youtu.be/hWGMPMPZ40M

Links:

- GitHub: https://github.com/MWirelabs/ne-agent
- PyPI: https://pypi.org/project/ne-agent/
- NE-Embed (HuggingFace): https://huggingface.co/MWirelabs/ne-embed
- NE-LID (HuggingFace): https://huggingface.co/MWirelabs/ne-lid
- NE-ASR (HuggingFace): https://huggingface.co/MWirelabs/ne-asr
- KrenTransV0-1 (HuggingFace): https://huggingface.co/MWirelabs/krentransv0-1
- MWire Labs: https://mwirelabs.com