Feasibility Study: Local-First Real-Time Screen Translation Ecosystems for Indic Languages

1. Introduction and Market Context

The global digital landscape is undergoing a profound transformation, shifting from centralized, cloud-dependent architectures toward edge computing and local-first Artificial Intelligence (AI). Within this paradigm shift lies a critical, under-served intersection: the need for real-time, privacy-preserving language accessibility tools tailored specifically for the linguistic complexity of the Indian subcontinent. The user's query proposes a distinct and technically ambitious concept: a local, open-source browser extension capable of performing real-time screen translation (Optical Character Recognition coupled with Neural Machine Translation) for Indian languages without reliance on internet connectivity.

This report provides an exhaustive analysis of the feasibility, novelty, and existing ecosystem surrounding this proposition. Currently, the market for translation tools is bifurcated. On one side are ubiquitous cloud-based browser extensions like Google Translate and DeepL, which offer high accuracy but demand constant connectivity and sacrifice user privacy by transmitting screen data to remote servers.[1] On the other side are standalone desktop applications like Crow Translate or specialized gaming overlays like Translumo, which offer varying degrees of offline capability but lack the seamless integration and ubiquity of a browser-native environment.[3]

The proposed solution—an "extension type" tool that reads the screen in real-time and overlays translated text locally—occupies a unique "white space" in this hierarchy. While the constituent technologies (WebAssembly, Tesseract.js, Transformers.js, and AI4Bharat’s IndicTrans2) have matured individually, their synthesis into a cohesive, user-friendly browser extension remains largely unexplored. This report argues that while technically demanding, the convergence of WebGPU acceleration and optimized Indic-centric models makes such a tool not only feasible but necessary to bridge the digital divide for Indian language users.

1.1 The Imperative for "Indianization"

A central theme of this investigation is the requirement for the tool to be "properly Indianized." In technical terms, this extends far beyond simple language support codes. It necessitates a deep architectural handling of Abugida writing systems, complex text segmentation, and the unique socio-linguistic phenomenon of code-mixing (e.g., Hinglish). Current generic OCR engines, predominantly trained on Latin scripts, often fail to correctly parse the *shirorekha* (headline) in Devanagari or the agglutinative morphology of Dravidian languages.[5] Therefore, a truly "Indianized" solution must leverage specialized datasets and models, such as those developed by AI4Bharat, rather than relying on generic global baselines.[6]

2. The Current Landscape of Translation Technologies

To rigorously assess the novelty of the proposed idea, it is essential to map the existing terrain of translation and OCR tools. This analysis reveals a spectrum of solutions, none of which perfectly align with the user's vision of a unified, offline, browser-based overlay for Indic languages.

2.1 Cloud-Dependent Browser Extensions

The dominant modality for browser translation is the cloud-proxy model. Extensions in this category function as thin clients, capturing text or images and forwarding them to powerful server farms for processing.

2.1.1 Google Translate and the Chrome Ecosystem

The Google Translate extension represents the incumbent standard. Historically, it has provided context-menu options to translate static images, but recent updates have integrated page translation natively into the Chrome browser (v2.14+), deprecating some extension-specific features.[7]

- **Mechanism:** It utilizes the Google Cloud Translation API, which supports over 100 languages including a wide array of Indic languages.[8]
- **Limitation:** Crucially, its functionality is strictly online. The "offline" capabilities mentioned in support documentation often refer to the Android mobile application, where language packs can be downloaded, rather than the desktop browser extension.[1]
- **Screen Reading:** While Google Lens offers "screen reading" capabilities, it typically requires a user to capture a screenshot or upload an image manually, disrupting the workflow for real-time tasks like watching a video or playing a web-based game.[9]

2.1.2 DeepL and Niche Competitors

DeepL is frequently cited for its superior translation quality, particularly in preserving nuance. However, its extension shares the same cloud dependency as Google's.

- **Language Gap:** DeepL supports approximately 32 languages. While it has begun to include major global languages, its coverage of the diverse Indian linguistic landscape (e.g., Santali, Dogri, Maithili) is virtually non-existent compared to the requirements of a "properly Indianized" tool.[2]
- **XTranslate:** This extension offers a middle ground, caching some data for speed, but predominantly relies on public APIs from Google, Yandex, and Bing. It facilitates text selection translation but does not possess the computer vision capabilities to "read" pixel-based text from a video feed or dynamic web element in real-time.[2]

2.2 Desktop-Native OCR and Translation Utilities

The functionality requested by the user—real-time screen reading—is currently best serviced by standalone desktop applications rather than browser extensions. These tools operate outside the browser sandbox, granting them unrestricted access to screen content via OS APIs.

2.2.1 Crow Translate

Crow Translate stands out as a sophisticated open-source application written in C++ and Qt. It exemplifies the "hybrid" approach.

- **Architecture:** It uses a modular provider system, allowing it to switch between Google, Yandex, and Bing APIs. Crucially, it supports OCR via Tesseract, theoretically enabling offline text extraction.[10]
- **Offline Reality:** While the architecture supports offline OCR, the translation step defaults to online APIs. Configuring it for fully offline translation requires setting up a local server (like LibreTranslate) or using heavy local models, which is a significant barrier for average users.[3]
- **Indian Language Support:** It supports 125 languages, including major Indic ones, but relies on the underlying quality of the connected API or the installed Tesseract data files.[11]

2.2.2 Translumo and Gaming Overlays

Translumo is perhaps the closest functional analog to the user's "real-time" requirement, designed specifically for gamers who need to translate dialogue in real-time.

- **Mechanism:** It captures screen regions and processes them through a pipeline of OCR engines (Tesseract, Windows OCR, EasyOCR) and translation APIs (DeepL, Google, etc.).[4]
- **Innovation:** It uses machine learning to "score" the OCR output from different engines, selecting the most probable text before sending it for translation.[4] This improves accuracy significantly on noisy backgrounds.
- **Limitation:** It is a heavy Windows executable, not a browser extension. Its reliance on the.NET framework and python backends makes it platform-dependent, whereas a browser extension would be cross-platform.[4]

2.2.3 PowerToys Text Extractor

For pure offline OCR on Windows, PowerToys Text Extractor is the standard. It leverages the Windows.Media.Ocr API built into Windows 10/11.

- **Performance:** It is extremely fast and runs entirely locally.
- **Indic Support:** It supports any language installed in the Windows Language settings. However, user reports indicate inconsistent reliability for complex Indic scripts compared to English or European languages, often struggling with specific ligatures or yielding lower accuracy than cloud-based alternatives.[12]

- **Gap:** It extracts text to the clipboard; it does not translate it or overlay it back onto the screen.[14]

2.3 The "Project Naptha" Paradigm and Its Legacy

No discussion of browser-based OCR is complete without referencing Project Naptha. Launched nearly a decade ago, it was a revolutionary Chrome extension that used algorithms (Stroke Width Transform) to detect text in images and allow users to highlight and copy it.[15]

- **Relevance:** It proved that JavaScript (even before WebAssembly was mature) could perform basic computer vision tasks in the browser.
- **Demise:** The project is now largely unmaintained.[16] It relied on older versions of Tesseract and did not focus on the high-fidelity translation overlay requested by the user. It serves as a proof of technical possibility but is not a viable modern solution.[17]

2.4 Firefox Translations (Project Bergamot)

Mozilla's Project Bergamot represents the state-of-the-art in *offline browser translation*.

- **Breakthrough:** By porting the MarianNMT engine to WebAssembly and optimizing for SIMD (Single Instruction, Multiple Data) instructions, Mozilla enabled high-performance neural translation on the client side CPU.[18]
- **Missing Link:** Currently, Firefox Translations only translates the DOM (text code). It has no OCR capability and cannot translate text embedded in images or videos. Furthermore, its language roadmap has been Euro-centric, with Indic language support still in nascent stages.[20]

2.5 Comparative Analysis of Solution Architectures

The following table contrasts the proposed idea against the most relevant existing tools, highlighting the gap in the current market.

Feature	Proposed	Crow	Translumo	Firefox	Google

Set	Solution	Translate		Translations	Lens
Environment	Browser Extension	Desktop App (Qt)	Desktop App (.NET)	Browser Feature	Web/Mobile
Network	**Offline**	Hybrid	Hybrid	**Offline**	Online
OCR Engine	**WASM (Tesseract)**	Tesseract/ Windows	Windows/EasyOCR	None	Cloud Vision
Translation	**WASM (IndicTrans2)**	Online API	Online API	WASM (Marian)	Cloud API
Interaction	**Real-time Overlay**	Static Selection	Real-time Overlay	Page Replace	AR Overlay
Indic Focus	**High**	Medium	Medium	Low	High

As the data indicates, the **Proposed Solution** combines the "Offline" capability of Firefox Translations, the "Real-time Overlay" of Translumo/Lens, and the "Browser Environment" of Naptha. This specific configuration is highly novel.

3. Technical Architecture for Local-First Indic Translation

Designing a browser extension that performs both OCR and NMT offline requires a sophisticated architecture that circumvents the traditional performance bottlenecks of JavaScript. The system must rely heavily on **WebAssembly (WASM)** and the emerging **WebGPU** standard to handle the computational load of neural networks on consumer hardware.

3.1 The Capture Layer: Accessing the Screen

The first hurdle is capturing the visual data. Chrome provides the chrome.desktopCapture API, which allows extensions to request a stream of the user's screen, window, or tab.[22]

- **Mechanism:** The extension requests a MediaStream ID. The user is presented with a native picker dialog to select the source (e.g., "Current Tab").
- **Privacy & Friction:** This API requires explicit user permission every time it is invoked, which acts as a privacy safeguard but adds friction compared to native apps.[23]
- **Optimization:** To achieve real-time performance, the extension cannot process every frame of the 60fps stream. Instead, it must implement a "sampling" strategy, grabbing a frame only when the screen content changes significantly (scene change detection) or at a fixed interval (e.g., 1-2 Hz), or triggered by user interaction (mouse hover).[15]

3.2 The Detection and OCR Layer

Once a frame is captured, the system must identify text. Processing the entire image through an OCR engine is prohibitively slow. A two-stage pipeline is required.

3.2.1 Text Detection (finding the "where")

Before reading text, the system must find it. Lightweight detection models like **EAST (Efficient and Accurate Scene Text Detector)** or quantized **YOLO (You Only Look Once)** can be run in the browser via **ONNX Runtime Web**.

- **Efficiency:** These models are significantly faster than OCR. They output bounding boxes for regions containing text.
- **Implementation:** The extension runs this detector on the sampled frame. If no text is found, the pipeline stops, saving battery. If text is found, the bounding boxes are cropped from the main image.[24]

3.2.2 Optical Character Recognition (reading the "what")

The cropped text regions are passed to the OCR engine.

- **Tesseract.js:** This is the industry standard for in-browser OCR. It runs a C++ codebase compiled to WASM.[25]
- **WASM Multithreading:** Tesseract.js supports Web Workers, allowing the heavy processing to happen on a background thread, preventing the browser UI from freezing.[27]
- **Indic Specifics:** Standard Tesseract data models (hin.traineddata, tam.traineddata) are often trained on document scans. For screen translation, the extension should ideally use models fine-tuned on "wild" data (low resolution, varied fonts). This is where the user's requirement for "Indianization" becomes critical—using generic models often leads to poor segmentation of Indic ligatures.[5]

3.3 The Translation Layer: Neural Machine Translation (NMT)

The extracted text string must now be translated. This is the most computationally intensive step.

- **Transformers.js:** This library is the key enabler. It allows state-of-the-art Transformer models (like those from Hugging Face) to run directly in the browser.[28]
- **Model Selection (IndicTrans2):** The user specifically needs Indic language support. **IndicTrans2**, developed by AI4Bharat, is currently the superior open-source model for this purpose, outperforming Meta's NLLB on Indic benchmarks.[29]
- **Quantization:** The full IndicTrans2 model (1B parameters) is too large for a browser extension (multiple gigabytes). To make this feasible, the model must be **quantized** (compressed) to 8-bit or 4-bit integer precision. This reduces the size to approx. 100-300MB with minimal loss in accuracy.[31]
- **ONNX Runtime:** The quantized model is executed using ONNX Runtime Web, which can leverage the CPU via WASM (using SIMD instructions) or, crucially, the GPU via WebGPU for massive acceleration.[32]

3.4 The Overlay Layer: Visual replacement

The final step is presenting the translation.

- **Canvas Overlay:** The extension injects a transparent HTML Canvas or SVG layer over the video/image.

- **In-painting:** For a seamless "Google Lens-like" experience, the tool ideally performs "inpainting"—erasing the original text and rendering the translation in a similar style. However, purely browser-based inpainting is computationally expensive. A simpler approach, used by Project Naptha and Translumo, is to render a high-contrast background box behind the translated text to obscure the original.[4]

4. The Challenge of "Proper Indianization"

The user asks if existing tools are "properly Indianized." This inquiry touches upon the specific difficulties of processing Indian languages, which belong predominantly to the Indo-Aryan and Dravidian families. These languages use **Abugida** scripts, where the concept of a "character" is fluid, involving base consonants, inherent vowels, and complex diacritic modifiers (Matras).

4.1 The Segmentation Problem in OCR

In Latin scripts, characters are distinct and separated by small gaps. In many North Indian scripts (Devanagari, Bengali, Gurmukhi), a continuous horizontal line called the **Shirorekha** connects the letters of a word.

- **Failure Mode:** Generic OCR engines often fail to separate connected characters, or they interpret the headline as noise. This results in "garbage" output where characters are split incorrectly.[5]
- **Solution:** A "properly Indianized" tool cannot rely on the default Tesseract English models. It must bundle specific traineddata files that have been trained on Indic datasets. AI4Bharat has released OCR models specifically optimized for these scripts, which handle the headline and vertical conjuncts significantly better than generic models.[6]

4.2 Agglutination and Morphology

Dravidian languages like Tamil and Malayalam are agglutinative—words are formed by stringing together morphemes. A single word in Tamil can be equivalent to a whole sentence in English.

- **Translation Implication:** A word-for-word translation approach (often used in simple tools) fails catastrophically here. The NMT model must contextually understand the entire agglutinated string. **IndicTrans2** is specifically architected to handle this morphology, whereas older models often truncated or mistranslated these long compound words.[29]

4.3 Code-Mixing (Hinglish/Tanglish)

Indian digital content is heavily code-mixed. A tweet or chat message might read "Bas 5 min mein pahunch raha hoon" (Hindi written in Latin script) or mix Devanagari and English ("फिल्म bahut achi thi").

- **Model Requirement:** The translation model must be robust to code-switching. Standard translation models often treat Romanized Hindi as "English" and fail to translate it, or output nonsense.[36] AI4Bharat's datasets explicitly include code-mixed data, making their models uniquely suited for this "Indianized" requirement compared to Euro-centric models like Bergamot's.[6]

5. Detailed Analysis of Existing Ecosystem

The user asks: "Is this thing already there?" To provide a definitive answer, we must dissect the specific capabilities of the leading tools in this space.

5.1 Firefox Translations (Project Bergamot)

- **Overview:** This is the most prominent "Offline Browser Translator." It uses the Bergamot engine (WASM) to translate locally.[18]
- **Why it falls short:** It is strictly a DOM (Document Object Model) translator. It translates text defined in HTML tags (<p>, <div>). It ignores text inside <img>, <canvas>, or <video> tags. Therefore, it cannot "read the screen" or translate memes, game UI, or scanned PDFs. Furthermore, its Indic language support is currently limited compared to its European support.[20]

5.2 Linguist Extension

- **Overview:** An open-source extension that aggregates various translation engines (Bing, Google, DeepL) and allows for custom configurations.[38]
- **Why it falls short:** While it has an "offline" mode, this usually refers to using a local server (like LibreTranslate) rather than being self-contained in the extension. More importantly, its OCR capabilities are not "real-time overlay." It requires the user to select an area manually, similar to a snipping tool, breaking the seamless experience the user desires.[40]

5.3 Translumo (The Desktop Competitor)

- **Overview:** An open-source desktop tool designed for gamers.[4]
- **Strengths:** It excels at the "overlay" aspect. It can detect text in a game window and overlay the translation. It uses Tesseract and Windows OCR.
- **Why it falls short:** It is a Windows executable, not a browser extension. It requires installation,.NET runtimes, and Python environments, making it heavy and complex for casual users. It is not "integrated" into the browsing experience (e.g., it doesn't scroll with the webpage).[4]

5.4 Crow Translate

- **Overview:** A Qt-based desktop translator.[10]
- **Strengths:** Robust, supports many languages.
- **Why it falls short:** Its primary utility is a "system tray" translator. While it has OCR, it is static (capture-and-translate), not dynamic (live video translation). Its offline mode requires significant user configuration (downloading Tessdata, setting up local servers), failing the "ease of use" test for a mass-market extension.[11]

5.5 ShareX + Custom Scripts

- **Overview:** Power users often combine ShareX (for OCR) with scripts to send text to translators.[41]

- **Why it falls short:** This is a DIY hack, not a product. It is disjointed and requires manual triggering for every translation event. It is not automated.[42]

6. Future Roadmap: Browser-Native AI

A critical insight for the user is the impending release of **Browser-Native AI APIs**. Both Google Chrome and other Chromium browsers are in the process of integrating Large Language Models (LLMs) and translation models directly into the browser executable.

6.1 Chrome's Built-in AI (Gemini Nano)

Google is rolling out the "Prompt API" and "Translation API" for Chrome.[43]

- **Significance:** This changes the feasibility landscape entirely. Instead of an extension developer needing to bundle a 200MB transformers.js model, they will soon be able to write code like await window.ai.translator.translate(text).
- **Impact on Proposed Idea:** This solves the "Download Size" and "RAM usage" problems. The extension can focus solely on the OCR (WASM) part, and offload the heavy translation to the browser's native, optimized model. This makes the user's idea much more viable in the 2025-2026 timeframe.[44]
- **Privacy:** Since the model is built-in to Chrome and runs on-device (Gemini Nano), it preserves the privacy requirement the user is concerned about.[46]

7. Feasibility and Implementation Strategy

7.1 Feasibility Verdict

The project is **Feasible**. All necessary components exist in the open-source ecosystem.

- **OCR:** Tesseract.js (WASM) is mature.

- **Translation:** Transformers.js + IndicTrans2 (Quantized ONNX) is functional.
- **Language Support:** AI4Bharat provides the necessary data.
- **Integration:** Browser APIs (desktopCapture, offscreen documents) allow the plumbing.

7.2 Recommended Stack for "The Novel Extension"

To build the tool the user envisions, the following stack is recommended:

1. **Core:** Manifest V3 Chrome Extension.
2. **Capture:** chrome.tabCapture (lower latency than desktopCapture for tab-specific needs).
3. **OCR Engine: Tesseract.js** utilizing the **Simd** (Single Instruction Multiple Data) WASM build for speed. It should load **AI4Bharat's Indic-finetuned traineddata** instead of standard Tesseract data.[26]
4. **Translation Engine: Transformers.js** loading a 4-bit quantized version of **IndicTrans2**. This should be loaded in a Web Worker to keep the main thread responsive.[47]
5. **Storage:** Use IndexedDB to store the large language models (100MB+) so they are only downloaded once.[32]
6. **UI/UX:** A React-based overlay injected into the Shadow DOM of the active tab to prevent CSS conflicts with the host page.

8. Conclusion

The concept of a local, open-source, real-time screen translator for Indian languages is a **novel and technically viable** proposition that addresses a significant gap in the current market. While generic tools like Google Translate and DeepL dominate the online space, and utilities like Translumo serve niche desktop needs, there is no unified browser extension that combines **offline privacy**, **Indic-centric optimization**, and **real-time overlay** capabilities.

By leveraging the convergence of **WebAssembly**, **Transformers.js**, and **AI4Bharat's open-source Indic models**, a developer can build this tool today. Furthermore, the impending arrival of **Chrome's built-in AI APIs** offers a future-proof path to reduce technical overhead and enhance performance. This tool would not only solve the user's specific friction points with Google Lens (uploading, online dependency) but also serve as a vital accessibility tool for millions of Indian language users navigating an English-dominant web.

8.1 Summary of Key Differentiators

Feature	Generic Cloud Extension	Desktop Utility (Crow/Translumo)	Proposed Indic Extension
Privacy	Low (Data sent to cloud)	High (Local processing)	**High (Local processing)**
Integration	Browser Toolbar	Separate Window	**Seamless Page Overlay**
Indic Quality	Variable (Generic models)	Variable (Generic Tesseract)	**High (AI4Bharat Models)**
Connectivity	Required	Optional (Complex setup)	**Not Required**
UX Friction	High (Select & Send)	High (Capture Hotkeys)	**Low (Automated)**

The proposed solution is not merely a "wrapper" for existing tools but a new architectural synthesis that specifically targets the linguistic and technical needs of the Indian user base.

9. References and Data Sources

The analysis in this report is underpinned by data from the following key sources:

- **Browser Translation APIs:** [1]
- **Desktop OCR Tools:** [3]
- **Indic Language Models (AI4Bharat):** [6]
- **Technical Libraries (Tesseract.js, Transformers.js):** [25]
- **Offline Browser Translation (Bergamot):** [18]
- **Indic Script Challenges:** [5]

This combination of sources confirms that while the *parts* exist, the *product* does not, validating the user's idea as a novel and valuable contribution to the open-source ecosystem.

Works cited

1. Download & use Google Translate - Computer, accessed November 22, 2025, https://support.google.com/translate/answer/6350850?hl=en&co=GENIE.Platform%3DDesktop
2. 7 Best Translation Chrome Extensions 2025 - WPS Office, accessed November 22, 2025, https://www.wps.com/blog/7-best-translation-chrome-extensions/
3. Crow Translate - Download, accessed November 22, 2025, https://crow-translate.updatestar.com/
4. ramjke/Translumo: Advanced real-time screen translator for games, hardcoded subtitles in videos, static text and etc. - GitHub, accessed November 22, 2025, https://github.com/ramjke/Translumo
5. Tesseract Models for Indian Languages - Indic-OCR, accessed November 22, 2025, https://indic-ocr.github.io/tessdata/
6. AI4Bharat, accessed November 22, 2025, https://ai4bharat.iitm.ac.in/
7. Google Translate - Chrome Web Store, accessed November 22, 2025, https://chromewebstore.google.com/detail/google-translate/aapbdbdomjkkjkaonfhkkikfgjllcleb?hl=en
8. Google Translate for Chrome - Download, accessed November 22, 2025, https://google-translate.en.softonic.com/
9. A Personal Interpreter on Your Phone or Computer - Google Translate, accessed November 22, 2025, https://translate.google.com/about/
10. Office / Crow Translate · GitLab - KDE Invent, accessed November 22, 2025, https://invent.kde.org/office/crow-translate
11. crow-translate/crow-translate: A simple and lightweight translator that allows you to translate and speak text using Google, Yandex Bing, LibreTranslate and Lingva. - GitHub, accessed November 22, 2025, https://github.com/crow-translate/crow-translate
12. Windows snipping tools text extractor vs Powertoys text extractor : r/Windows11 - Reddit, accessed November 22, 2025, https://www.reddit.com/r/Windows11/comments/1m6zkl1/windows_snipping_tools_text_extractor_vs/
13. Text Extractor can usually be relied on? : r/PowerToys - Reddit, accessed November 22, 2025, https://www.reddit.com/r/PowerToys/comments/1f46rot/text_extractor_can_usually_be_relied_on/
14. PowerToys Text Extractor Utility for Windows - Microsoft Learn, accessed November 22, 2025, https://learn.microsoft.com/en-us/windows/powertoys/text-extractor
15. Project Naptha, accessed November 22, 2025, https://projectnaptha.com/
16. Pengetahuan.md - GitHub Gist, accessed November 22, 2025, https://gist.github.com/kawainime/8d878d64a4563a540e2968df1c2ab055

17. Project Naptha for Google Chrome - Extension Download, accessed November 22, 2025, https://project-naptha.en.softonic.com/chrome/extension
18. Mozilla Firefox 117 beta brings an automatic language translator for websites - gHacks.net, accessed November 22, 2025, https://www.ghacks.net/2023/08/02/mozilla-firefox-117-beta-brings-an-automatic-language-translator-for-websites-and-it-works-offline/
19. Mozilla releases local machine translation tools as part of Project Bergamot, accessed November 22, 2025, https://blog.mozilla.org/en/mozilla/local-translation-add-on-project-bergamot/
20. Firefox 117 introduces local translations - TUXEDO Computers, accessed November 22, 2025, https://www.tuxedocomputers.com/en/firefox-117-introduces-local-translations.tuxedo
21. Firefox Translation - Mozilla Support, accessed November 22, 2025, https://support.mozilla.org/en-US/kb/website-translation
22. chrome.desktopCapture | API - Chrome for Developers, accessed November 22, 2025, https://developer.chrome.com/docs/extensions/reference/api/desktopCapture
23. 22 Understanding Chrome Extensions Desktop Capture | by M2K Developments - Medium, accessed November 22, 2025, https://m2kdevelopments.medium.com/22-understanding-chrome-extensions-desktop-capture-31edfd2f512d
24. ONNX Runtime Web—running your machine learning model in browser, accessed November 22, 2025, https://opensource.microsoft.com/blog/2021/09/02/onnx-runtime-web-running-your-machine-learning-model-in-browser
25. Image to Text OCR - Extract Text from Images Online Free - WuTools, accessed November 22, 2025, https://wutools.com/image/image-to-text
26. Tesseract.js | Pure Javascript OCR for 100 Languages!, accessed November 22, 2025, https://tesseract.projectnaptha.com/
27. Tesseract.js: Pure JavaScript OCR for 100 Languages | Hacker News, accessed November 22, 2025, https://news.ycombinator.com/item?id=21843713
28. Transformers.js - Hugging Face, accessed November 22, 2025, https://huggingface.co/docs/transformers.js/en/index
29. IndicTrans2 - AI4Bharat, accessed November 22, 2025, https://ai4bharat.iitm.ac.in/areas/model/NMT/IndicTrans2
30. AI4Bharat/IndicTrans2: Translation models for 22 scheduled languages of India - GitHub, accessed November 22, 2025, https://github.com/AI4Bharat/IndicTrans2
31. How to convert NLLB models to ONNX format and quantify them in INT4 #76 - GitHub, accessed November 22, 2025, https://github.com/niedev/RTranslator/issues/76
32. Deploying ONNX Runtime Web | onnxruntime, accessed November 22, 2025, https://onnxruntime.ai/docs/tutorials/web/deploy.html
33. Run AI Models Entirely in the Browser Using WebAssembly + ONNX Runtime (No Backend Required) - DEV Community, accessed November 22, 2025,

https://dev.to/hexshift/run-ai-models-entirely-in-the-browser-using-webassembly-onnx-runtime-no-backend-required-4lag
34. Comparative Analysis of Outcomes of Tesseract OCR for Different Languages | Request PDF - ResearchGate, accessed November 22, 2025, https://www.researchgate.net/publication/380416121_Comparative_Analysis_of_Outcomes_of_Tesseract_OCR_for_Different_Languages
35. \textls[0]Decoding the Diversity: A Review of the Indic AI Research Landscape - arXiv, accessed November 22, 2025, https://arxiv.org/html/2406.09559v1
36. Firefox is getting language translation | Hacker News, accessed November 22, 2025, https://news.ycombinator.com/item?id=21321430
37. Translate to and from 400+ languages locally with MADLAD-400 : r/LocalLLaMA - Reddit, accessed November 22, 2025, https://www.reddit.com/r/LocalLLaMA/comments/17qt6m4/translate_to_and_from_400_languages_locally_with/
38. Linguist Translate, accessed November 22, 2025, https://linguister.io/
39. Best Open Source Google Translate Alternatives (2025), accessed November 22, 2025, https://openalternative.co/alternatives/google-translate
40. open source offline image translator? : r/opensource - Reddit, accessed November 22, 2025, https://www.reddit.com/r/opensource/comments/x51qh4/open_source_offline_image_translator/
41. Offline OCR · Issue #5656 - GitHub, accessed November 22, 2025, https://github.com/ShareX/ShareX/issues/5656
42. A question about OCR language · Issue #6855 · ShareX/ShareX - GitHub, accessed November 22, 2025, https://github.com/ShareX/ShareX/issues/6855
43. How Policybazaar and JioHotstar use the Translator and Language Detector APIs to build multilingual experiences | Blog | Chrome for Developers, accessed November 22, 2025, https://developer.chrome.com/blog/pb-jiohotstar-translation-ai
44. Chrome Built-in Offline AI Chat/Translation, accessed November 22, 2025, https://chromewebstore.google.com/detail/chrome-ai-chrome-built-in/emlpbjnkjcbepocijpjggfmmloipamia
45. Google AI for Developers: Gemini Developer API | Gemma open models, accessed November 22, 2025, https://ai.google.dev/
46. WebAssembly and WebGPU enhancements for faster Web AI, part 1 | Blog | Chrome for Developers, accessed November 22, 2025, https://developer.chrome.com/blog/io24-webassembly-webgpu-1
47. Building a React application - Hugging Face, accessed November 22, 2025, https://huggingface.co/docs/transformers.js/en/tutorials/react
48. @codethicket/react-ai-translator - npm, accessed November 22, 2025, https://www.npmjs.com/package/@codethicket/react-ai-translator?activeTab=readme
49. How to Run Tesseract OCR for Hindi-English Language: Full Setup, Best Config & Sample Result - Aditya Mangal, accessed November 22, 2025, https://adityamangal98.medium.com/how-to-run-tesseract-ocr-for-hindi-english

-language-full-setup-best-config-sample-result-8af763d68f73