

A Novel Hybrid Framework for Mobile Sign Language Translation, Local LLM Text Generation, and Analytics*

Zachary Eanes, Scott Barlowe, Alex Charlot, and Andrew Scott
Department of Mathematics and Computer Science
Western Carolina University
Cullowhee, NC 28723

{zteanes1, ajcharlot1}@catamount.wcu.edu
{sabarlowe, andrewscott}@email.wcu.edu

Abstract

Hearing impairment impedes critical verbal communication in education, business, entertainment, and interpersonal relationships. Severe hearing impairment often affects speech development and increases reliance on sign language to both convey and receive information. Sign language is complex, nuanced, and often difficult to learn. There has been much effort in applying advances in machine learning and application development to ease the interpretation of sign language. In this paper, we present a novel platform for facilitating the interpretation of sign language in an intuitive mobile environment. Specifically, the system for sign language translation presented here combines an intuitive user interface for gesture recording and platform navigation, a pretrained machine learning model for translation, a tunable local large-language model for sentence construction, and presentation of translation analytics created from remote user information. Functionality is implemented with an innovative client-server architecture that balances on-device and off-device resources.

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

1 Introduction

Communication is a fundamental part of the human experience and takes a multitude of forms that most of us use without much consideration. Verbal communication, a primary way of conveying information, is not possible for many people. In the United States alone, recent census data reported that more than 12 million people have difficulty hearing [5]. The hearing impaired often communicate through one of many developed sign languages where manual signals (e.g., hand position, finger position, and motion) are supplemented by other nonverbal signals (e.g., eye gaze, body movement, and head orientation) to provide a comprehensive and accurate method of communication [2]. American Sign Language (ASL) is one of the most common sign languages used in the United States, and a 2006 study reported that ASL is the primary language of up to 500,000 people [21]. However, that study is now almost 20 years old and may no longer accurately reflect sign language usage in the United States. A 2014 study estimated that over 6 million people in the United States use some type of sign language [20] and in 2018 the Modern Language Association reported that ASL is the third most commonly taught language in colleges and universities [4].

The hearing impaired often develop altered speech patterns [25], [26] and sign language provides a common baseline for communicating with those without hearing impairment. However, many activities in education, business, entertainment, and personal relationships critical to both hearing and hearing impaired individuals can be hindered if one party does not know how to interpret sign language. Our work aims to fill this communication gap by providing a system that facilitates mobile ASL translation with a rich set of supporting interactions and functionality. The system presented here integrates mobile application development, machine learning, a tunable local large-language model, and translation analytics to facilitate communication between those who rely on sign language (signers) and those who do not know sign language (nonsigners). Specifically, we present a novel platform that provides the following in a mobile application:

- A user-friendly and intuitive interface
- Capture of motion-based, word-level ASL gestures
- Gesture translation with machine learning
- Sentence construction with a local large-language model
- Presentation of translation analytics

Our platform balances on-device and off-device resources with an innovative client-server architecture. The mobile application manages the user interface and gesture capture. A server executes language recognition, manages the local large-language model, and returns information to the mobile application. A remote database stores usage information and is queried when the user requests translation analytics.

The remainder of the paper is organized as follows. We begin by presenting related work and explaining how our approach differs from previous attempts. Next, the platform interface and functionality available to the user are described. We then present the novel implementation that provides the foundation for the user experience. The advantages and shortcomings of the system are discussed next. Finally, we conclude with future work.

2 Related Work

Previous work most relevant to our approach can be divided into two related but distinct areas. The first area consists of efforts to improve the effectiveness of machine learning techniques employed for sign language recognition and generation. The second area consists of platforms that facilitate real-time (or close to real-time) translation.

2.1 Machine Learning

Machine learning has been widely applied to sign language. Alaghband et al. [2] provides an extensive review of machine learning efforts for manual recognition, facial recognition, and translation tasks. The techniques are too numerous to list here, but the surveyed methods include convolutional neural networks (CNNs), long short-term memory, gated recurrent units, generative adversarial networks, and hybrid machine learning architectures. Alaghband et al. also surveyed hardware to capture sign language gestures such as specialized cameras and gloves. Recently, large-language models (LLMs) have been applied to sign language recognition and generation. For example, Fang et al. [6] developed an LLM to generate sign language poses from textual input, and Hwang et al. [12] proposed an architecture that provides spatial features, motion features, and a language prompt for an LLM that returns sign language translations.

2.2 Translation Applications

Translation platforms are available as mobile or web applications. *Hand Talk* [27] is a mobile application that provides learning resources and also translates text and audio into ASL and Brazilian Sign Language. *SignVision* [22] is a

web application available for mobile devices and provides real-time translation of Singapore Sign Language. *SignVision* sends strings of translated words to ChatGPT [23] for sentence construction. *Sign Language AI* [14] is a mobile application that claims to have both real-time translation of over 2600 signs to multiple languages, English to sign language translation, and search capabilities. However, no information about the implementation is provided. Numerous sign language translation implementations have recently chosen MediaPipe [11] to facilitate real-time gesture recognition. MediaPipe is a suite of libraries that relies on landmark detection to provide the machine learning infrastructure for gesture recognition across a range of platform environments, including mobile devices.

Our approach differs from those mentioned above. Efforts focused on machine learning approaches seek model development or refinement. Integration into applications that can be widely distributed is often a secondary concern or postponed to future work. For example, LLMs employed in approaches seeking to improve or expand translation capabilities need to be first custom-tailored for sign language translation before integration. There are other differences. Architectures for the mobile and web translation applications are either unknown, use a remote and/or general-purpose LLM outside of the developer’s control, or do not provide a framework for providing analytics that may be useful when first learning to communicate with sign language. Furthermore, our framework avoids the need for the developer to have direct knowledge of landmark detection techniques to recognize gestures. The system presented here provides a user-friendly mobile application that integrates a pretrained machine learning model, a tunable large-language model that executes locally, and access to usage analytics.

3 System Design

We now present a novel architecture for a mobile application that translates word-level ASL and addresses the shortcomings of previous approaches. Our approach (Figure 1) consists of a client-server architecture. The mobile application is the client and provides an intuitive user interface built with Flutter [10]. The mobile application accesses our server that executes a pretrained three-dimensional CNN to recognize gestures and a tunable local LLM to construct a sentence consisting of translated words. The mobile application also stores usage information in a remote database and queries that database to present translation analytics to the user. The user experience is described next and then implementation details follow.

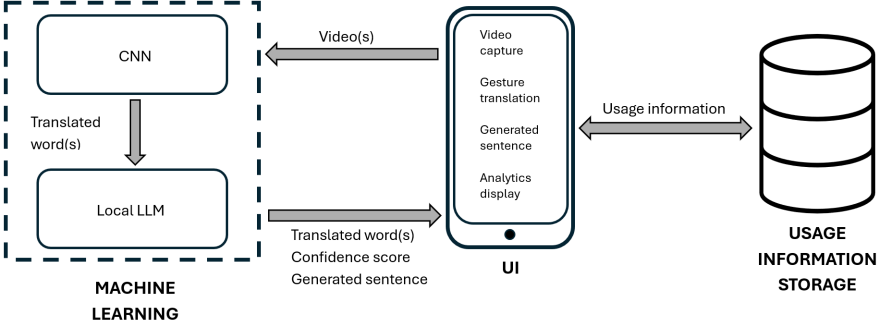


Figure 1: **System Architecture.** A mobile application records gestures and sends videos representing individual gestures to the server where the recognition model and local LLM reside. The model assigns a translation for each gesture and the translated words are input into the local LLM to generate a sentence. The translated words, overall confidence score, and sentence constructed by the LLM are returned to the mobile application. Usage information is stored in a remote database where they can be retrieved by the mobile application and used to generate translation analytics.

3.1 User Experience

Account Creation and Settings. After installing the application on a mobile device, the user must create an account before using any of the functionality. An account is created by providing a first name, last name, email, and password. A settings screen (Figure 2(a)) provides several options. Our system uses a green-yellow-red color scheme to report confidence scores from gesture translations, where green indicates high confidence, yellow indicates medium confidence, and red indicates low confidence. However, we recognize that users may have red-green colorblindness and provide alternate color schemes for two main types. Alternate color schemes for users with protanopia and deuteranopia can be chosen by selecting *RCB* and *GCB*, respectively. The settings screen also allows the list of translated words and the sentences generated by the LLM to be exported to pdf format.

Video Capture. The first step in translating sign language with our system is to capture a sequence of one or more videos where each video represents an individual ASL gesture (Figure 2(b)). The nonsigner aims the mobile phone’s camera at the signer. The user may record themselves by selecting the *camera-flip* button in the top right corner of the interface (Figure 2(b)-1). Gesture capture is initiated by selecting the *record* button (Figure 2(b)-2). Additional gestures needed to convey a thought or sentence can be recorded

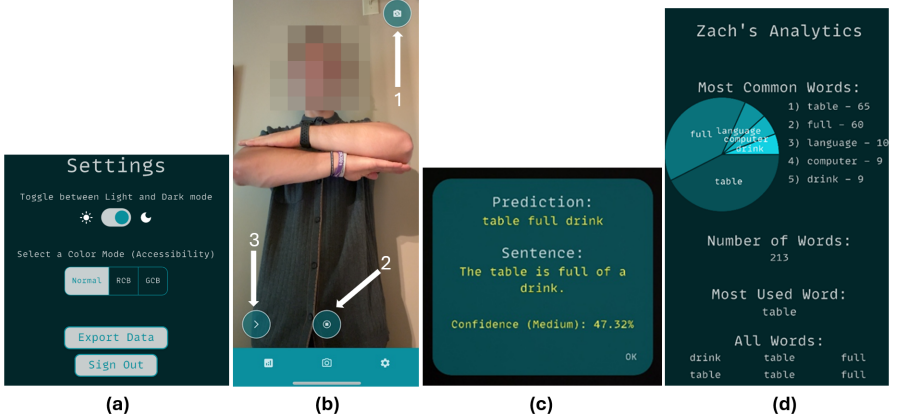


Figure 2: **The user interface.** (a) A settings screen allows users to choose color schemes appropriate for two types of red-green colorblindness and to toggle between light and dark mode. (b) Users can flip the camera view (see 1). The capture process is started and finished by selecting the *record* button (see 2). Gestures can be grouped together by selecting the *next* button before each additional video (see 3). In this example, ASL gestures for *table*, *full*, and *drink* are recorded and grouped together for translation. (c) The translated words from the grouped ASL gestures, the sentence constructed by the local LLM, and the overall confidence score are displayed. (d) The analytics screen displays the most commonly translated words, the number of translated words, and a list of all words translated.

and grouped with the first video in the sequence by selecting the *next* button (Figure 2(b)-3). Before adding a video to the sequence, the signer needs to be notified that the system is ready to record additional gestures. The notification must be 1) convenient for the user who is already using one hand to aim the camera and 2) clear to the signer who may be hearing impaired. Our solution is to utilize the flash on the camera, which is activated when the user initiates the recording of additional videos by choosing the *next* button. Selecting the *record* button again ends the capture process for the sequence of gestures to be grouped.

Translation and Sentence Construction. The server receives the videos and translates the gestures represented in each video into American English words. The mobile application receives the translation and the overall confidence score of the model’s translation (ranging from 0 to 1). The overall confidence score is currently the average of the confidence scores of each translated word in the sequence captured by the user. All of the translated words

presented to the user are color-coded according to categories of model confidence. Translations with an overall confidence score > 0.70 are presented in green, translations with an overall confidence score ≥ 0.35 and ≤ 0.70 are presented in yellow, and translations with an overall confidence score < 0.35 are presented in red. In the future, we plan to experiment with other definitions of overall confidence and to allow the user to adjust the category boundaries.

Before sending the translations and confidence score to the mobile application, the translated words are fed into a local LLM, which forms an American English sentence based on the input. The sentence generated by the local LLM is sent to the mobile application along with the translated words and is presented with the same color as the translated words. Figure 2(c) shows the translated words *table*, *full*, and *drink* and the resulting sentence (*The table is full of a drink.*) were returned with a medium confidence score (as determined by our implementation).

Analytics. Although our application does not currently offer the specialized functionality offered by platforms primarily focused on learning sign language (see [17], [3], [24], [15] for examples), we recognize the importance of providing a nonsigner guidance on where to begin learning activities. To help inexperienced signers concentrate on the words that are frequently encountered, our system provides an analytics screen accessed by selecting the leftmost icon at the bottom of Figure 2(b). The most commonly translated words, the number of translated words, and a list of all translated words (including duplicates) for a user are presented with text and a pie chart (Figure 2(d)). In the future, we plan to expand both the amount of information stored and the analytics presented.

3.2 System Implementation

The implementation that facilitates the functionality of the user experience is now described. Specifically, we provide details for video capture, video translation, sentence generation, and translation analytics.

Video Capture. Our system gives users the ability to group as many gestures as desired and to let the user decide when to initiate translation. The client (the on-device mobile application) captures the video representing a single gesture using the device’s camera. Each recorded video is immediately sent to the server with a flag that indicates if the video is the last in the sequence.

Video Translation. The server is structured with FastAPI [7] and uses PyTorch [8] to translate each video into a word recognized by the system. Each video is segmented into individual frames that are then resized for consistency and upsampled. Our system uses the Inception-v1 I3D model [16], a three-dimensional CNN, pretrained on word-level ASL videos. Recognition of

a gesture is performed by sorting the confidence scores returned by the model for each possible gesture and then mapping the confidence scores to a list of preloaded words accessed by the system. This is performed for each video in the group of videos, resulting in a list of translated words that is returned to the mobile application and presented to the user.

Sentence Generation. Before sending the translated words to the mobile application, our system passes them to an LLM for sentence construction. Instead of using a closed (e.g., hidden model weights) and remote LLM such as ChatGPT [23], we use an open-weight, tunable, and local LLM. Specifically, we use GPT4All [1] to call a quantized, 8 billion-parameter version of Meta’s Llama 3 [19] model. Several constraints are enforced via hard-coded prompts to the LLM. For each constructed sentence, the LLM is reminded of its purpose and the desired format: *Translate provided American Sign Language (ASL) into English text and summarize the message into a single, coherent sentence.* The system also provides prompts that request the LLM to exclude additional context or information. The sentence constructed by Llama 3 is returned to the mobile application, along with the translated word for each gesture and the overall confidence score.

Translation Analytics. The mobile application receives the translated labels, the constructed sentence, and the overall confidence score from the server. Google’s Cloud Firestore [9], a NoSQL database, is used to store account information, translated words, and constructed sentences. When the user visits the analytics screen, the mobile application accesses the remote database and then generates the statistics presented to the user.

4 Results and Discussion

The advantages and shortcomings of our platform are described below. We organize the discussion into sections for architecture design, model usage, and user experience.

4.1 Architecture Design

The system uses a novel client-server architecture that balances on-device and off-device resources. Separating machine learning from the mobile application allows any model updates to be performed without users needing to update the application. Placing machine learning on a remote server also allows access to increased resources and eliminates the need for model pruning [18], [13] and other optimizations that often need to occur before executing machine learning algorithms on mobile devices. Usage information is also stored remotely, which could potentially allow for analytics across multiple devices and/or users. We utilize the limited resources on the mobile device for execution of the user

interface, video recording, and calculating statistics instead of machine learning or storage.

The choice of a local LLM allows gesture recognition and large-language model management to be implemented at the same location. In our system, this eliminates the need to access a remote model for sentence construction. The Llama 3 [19] model specifically provides the additional benefit of a tunable baseline LLM. This provides a suitable balance by 1) eliminating the need to implement a specialized LLM but 2) allowing future customization in the training process and resource utilization.

4.2 Model Usage

Our system currently uses the Inception-v1 I3D model from Li et al. [16] trained on over 119 signers in over 20,000 videos for four different categories: 100 words, 300 words, 1000 words, and 2000 words. We chose the version trained on 100 words since Li et al. report a higher accuracy than models trained on other word counts. This higher accuracy aided us during testing and required a smaller number of words to be preloaded as labels when initially building the system. Expanding the platform to include model versions with a higher word count would be relatively straightforward and would consist mainly of swapping out the current model and increasing our list of words that the system checks before assigning a label.

During experimentation with our system, we noted several issues related to model performance. We suspect that at least some of the variation in confidence scores can be attributed to differences in motion. Words that have unique motion patterns as a large part of their expression seem to be distinguished more easily than those that have similar motion patterns but rely primarily on nuanced differences in hand formation. Another issue we found was the importance of video length. During initial testing, we attempted to increase model performance with environmental variations during video capture (e.g., environment light and contrast). However, our model improved the most after video lengths were cropped as much as possible without altering what was being signed. We plan to investigate both of these issues further in the context of both prior work in video recognition and our own implementation.

Figure 2 shows our attempt to produce a translation for a simple sentence: *The table is full of drinks*. Separate gestures for *table*, *full*, and *drink* were recorded and translated. The result produced by the LLM (*The table is full of a drink.*) appears close to our intention. However, our experience in ASL is limited to what was necessary to build this system and to perform preliminary tests. We plan to collaborate with an ASL expert to further investigate the accuracy of the sentence constructed by the LLM, including gesture recognition refinements and LLM tuning.

4.3 User Experience

Our system provides a novel, intuitive user experience, but there are areas of interaction that we would like to improve. For example, a nonsigner indicates readiness to record another gesture with a flashing light initiated by selecting the *next* button. This method is convenient for the user holding the phone and clear to the signer. However, this approach also requires the nonsigner to know when a gesture starts and finishes which may shift more responsibility to the signer to insert gaps between gestures or to otherwise notify the nonsigner where words begin and end. *Sign Language AI* [14] claims to provide real-time performance of ASL gesture recognition and we would like to investigate ways to eliminate the need to manually capture video and still retain the balance of on-device and off-device balance achieved by our system.

Another area of improvement that we would like to investigate is reducing system delays. Initial testing of our system with clock utilities natively provided by the frameworks we chose for implementation resulted in approximately 2.5 seconds to translate a single word (not including recording or constructing a sentence) and approximately 7.5 seconds for the LLM to construct a sentence. Increasing the number of words input to the LLM had a negligible effect on the time for sentence construction. The server was equipped with only a single Nvidia GeForce RTX 3070 graphics card and we plan to experiment with additional resources.

5 Conclusion and Future Work

We have presented a novel architecture for bridging the gap between those who rely on sign language to communicate and those who wish to understand the hearing impaired but do not know sign language. Our system uses a pretrained CNN to recognize gestures recorded by the user. A tunable local LLM is used to construct sentences for the translated words. Our system tracks translation information, stores the information in a remote database, and retrieves the information when needed. Gesture recognition and translation, sentence construction, and translation analytics are tied together with an intuitive user interface. In the immediate future, we plan to expand the number of gestures that can be translated, investigate ways to improve video capture, and experiment with model tuning in collaboration with an ASL expert.

References

- [1] Nomic AI. *GPT4All*. <https://www.nomic.ai/gpt4all/>.

- [2] Marie Alaghband, Hamid Reza Maghroor, and Ivan Garibay. “A survey on sign language literature”. In: *Machine Learning with Applications* 14 (2023), p. 100504. ISSN: 2666-8270. DOI: <https://doi.org/10.1016/j.mlwa.2023.100504>. URL: <https://www.sciencedirect.com/science/article/pii/S2666827023000579>.
- [3] The ASL App. <https://theaslapp.com/>.
- [4] Modern Language Association. *Enrollments in Languages Other Than English in United States Institutions of Higher Education, Summer 2016 and Fall 2016: Final Report*. <https://www.mla.org/content/download/110154/file/2016-Enrollments-Final-Report.pdf>.
- [5] U.S Census Bureau. *2023 American Community Survey*. <https://data.census.gov/table/ACSST1Y2023.S1810>.
- [6] Sen Fang et al. “SignLLM: Sign Languages Production Large Language Models”. In: *CoRR* abs/2405.10718 (2024). URL: <https://doi.org/10.48550/arXiv.2405.10718>.
- [7] *FastAPI*. <https://fastapi.tiangolo.com/>.
- [8] PyTorch Foundation. *PyTorch*. <https://pytorch.org/>.
- [9] Google. *Firestore*. <https://firebase.google.com/products/firestore>.
- [10] Google. *Flutter*. <https://flutter.dev/>.
- [11] Google. *MediaPipe*. <https://ai.google.dev/edge/mediapipe/solutions/guide>.
- [12] Eui Jun Hwang et al. “An Efficient Gloss-Free Sign Language Translation Using Spatial Configurations and Motion Dynamics with LLMs”. In: *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by Luis Chiruzzo, Alan Ritter, and Lu Wang. Albuquerque, New Mexico: Association for Computational Linguistics, Apr. 2025, pp. 3901–3920. ISBN: 979-8-89176-189-6. URL: <https://aclanthology.org/2025.naacl-long.197/>.
- [13] Yuyang Jin et al. “Efficient Inference for Pruned CNN Models on Mobile Devices With Holistic Sparsity Alignment”. In: *IEEE Transactions on Parallel and Distributed Systems* 35.11 (2024), pp. 2208–2223. DOI: 10.1109/TPDS.2024.3462092.
- [14] Paul Kelly. *Sign Language AI*. https://play.google.com/store/apps/details?id=ai.terp.www.twa&hl=en_US&pli=1.

- [15] Vaclav Knapp and Matyas Bohacek. “Pose-aware Large Language Model Interface for Providing Feedback to Sign Language Learners”. In: *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*. ASSETS ’24. St. John’s, NL, Canada: Association for Computing Machinery, 2024. ISBN: 9798400706776. DOI: 10.1145/3663548.3688515. URL: <https://doi.org/10.1145/3663548.3688515>.
- [16] Dongxu Li et al. “Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods Comparison”. In: *The IEEE Winter Conference on Applications of Computer Vision*. 2020, pp. 1459–1469.
- [17] Lingvano. <https://www.lingvano.com/asl/>.
- [18] Jiachen Mao et al. “TPRune: Efficient Transformer Pruning for Mobile Devices”. In: *ACM Trans. Cyber-Phys. Syst.* 5.3 (Apr. 2021). ISSN: 2378-962X. DOI: 10.1145/3446640. URL: <https://doi.org/10.1145/3446640>.
- [19] Meta. *Llama 3*. <https://www.llama.com/models/llama-3/>.
- [20] Ross E Mitchell and Travas A Young. “How Many People Use Sign Language? A National Health Survey-Based Estimate”. In: *The Journal of Deaf Studies and Deaf Education* 28.1 (Jan. 2023), pp. 1–6. DOI: <https://doi.org/10.1093/deafed/enac031>.
- [21] Ross E. Mitchell et al. “How Many People Use ASL in the United States?: Why Estimates Need Updating”. In: *Sign Language Studies* 6.3 (2006), pp. 306–335. ISSN: 03021475, 15336263. URL: <http://www.jstor.org/stable/26190621> (visited on 05/17/2025).
- [22] Nasrullah Nazaruddin. “SignVision — A Real-time Mobile Sign Language Recognition Application built using ChatGPT”. In: *Medium* (Apr. 2024). URL: <https://medium.com/@nesruler/signvision-a-real-time-mobile-sign-language-recognition-application-using-chatgpt-1ce73a0c6a55>.
- [23] OpenAI. *ChatGPT*. chatgpt.com.
- [24] Hope Orovwode, Oduntan Ibukun, and John Amanesi Abubakar. “A machine learning-driven web application for sign language learning”. In: *Frontiers in Artificial Intelligence* 7 (June 2024). DOI: 10.3389/frai.2024.1297347.
- [25] Nadia Porcar-Gozalbo et al. “Impact of Hearing Loss Type on Linguistic Development in Children: A Cross-Sectional Study”. In: *Audiology Research* 14.6 (2024), pp. 1014–1027. ISSN: 2039-4349. DOI: 10.3390/audiolres14060084. URL: <https://www.mdpi.com/2039-4349/14/6/84>.

- [26] Jessica A. Scott and Hannah M. Dostal. “Language Development and Deaf/Hard of Hearing Children”. In: *Education Sciences* 9.2 (2019). ISSN: 2227-7102. DOI: 10.3390/educsci9020135. URL: <https://www.mdpi.com/2227-7102/9/2/135>.
- [27] Hand Talk. *Hand Talk Translator*. https://play.google.com/store/apps/details?id=br.com.handtalk&hl=en_US.