Sign language translation using machine learning

Introduction

Globally, there are an estimated 70 million deaf individuals, with approximately 12 to 25 million using sign language as their primary mode of communication. Sign language has deep cultural significance and plays a role in shaping identity within deaf communities.

Sign languages are full, rich, natural languages with their own grammar, structure and expressive depth. They are as complex and capable as spoken or written languages, though they work differently. Spoken English is linear and word-based. In contrast, sign languages like British Sign Language (BSL) are visual and multidimensional. Sign languages use hand shapes, movement, facial expressions, body posture and space to convey meaning.

This structure makes sign languages complex. Often, the meaning of a sign depends on context.

Deaf and hearing people often rely on sign language interpreters to communicate. In the UK only 908 sign language interpreters are registered. Far too few to support the estimated 77,000 deaf people in the country.

The need for sign language interpreters in higher education is at an all-time high, but a shortage makes it hard to give deaf students the support they need. This is just one example of the barriers deaf individuals face to full participation in society. Despite legal protection such as the Equality Act 2010 or the Americans with Disabilities Act (ADA), the practical reality for many deaf individuals is exclusion. Access to healthcare, education, legal services and employment opportunities are often limited or entirely absent if interpreters are not provided.

AI could help in these situations. With thoughtful design, machine learning (ML) has the potential to assist, not replace, human interpreters and offer real time communication support.

Around 75% of deaf people face illiteracy. Many deaf students leave school reading well below the expected level. This is not a reflection of intelligence or capability. Rather it results from systemic barriers, including delayed or inadequate exposure to any accessible language during crucial periods of language development. Many educational systems prioritize spoken or written language. Institutional bias and a lack of inclusive strategies often result in lower expectations and reduced opportunities for deaf learners.

Finally, one of the most fundamental barriers to written language is the linguistic distance between signed and spoken languages. They differ drastically in structure and grammar. For native signers, spoken language is a second language and one that may always feel more difficult and less natural.

As a result, captions alone are not enough for full accessibility. This is why sign language production (speech-to-sign) is as important as sign language translation (sign-to-speech).

At Arm, the Developer Advocacy team is exploring how machine learning can enable a sign-to-speech translator for video conferencing. This would allow deaf users to sign naturally while the system translates their input into speech. This would make video conferencing more accessible for those hard of hearing to communicate with the hearing community.

The project originally aimed to explore sign language translation. This seemed like a logical extension of existing language translation tools which use machine learning. However, as we researched the state-of-the-art algorithms and spoke to people in the field, it became apparent just how complex, rich and underrepresented sign language is in the world of AI.

One reason is the lack of well annotated signing datasets. Given these challenges and after speaking with sign.mt (a company working in sign language processing), the project shifted focus. It now explores sign language production on mobile. Specifically translating text into SignWriting, a visual way of writing sign language, using sign.mt’s open source PyTorch model.

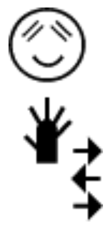


Figure 1: SignWriting of the sign for Hello.

In this blog post, I share insights from my research into sign language. Much of this blog post focuses on the challenges of sign language for machine learning and providing an overview into current technologies.

The team at sign.mt told us that the challenge of sign language translation needs more awareness and collaboration. This blog post is part of that mission. To share what we have learnt and hopefully open the door to discussion and inspire others to care about making technology more accessible.

Challenges of Sign Language Translation

Sign languages are complex, which is one reason machine learning models remain underdeveloped. Like spoken languages, sign languages have their own grammar. Sign language is multidimensional. It does not just involve hand movements, it also includes movement of the body and head, facial expressions, and body posture.

Signers often use these features at the same time to share more meaning. For example, one hand might point to a person or object. At the same time facial expressions and torso movements add context. A head shake can negate the entire sentence.

In many cases a single word does not match a single sign. The mapping between spoken words and signs is highly contextual and nonlinear, making it challenging for machine translation.

There are over 300 different sign languages used worldwide. Each can have its own idiolects. As a result, a single model for all sign languages is very challenging. Models must be either trained separately for each language or use a large multilingual dataset.

Isolated signs can be significantly easier for machine translation, where individual signs are captured and classified outside of sentence context. These models are simpler to build. But they do not capture the natural flow of real conversation. Additionally, lexical translation (word for word) simply does not work.

Directly translating spoken words to their respective signs results in ungrammatical and often confusing output. This limits its usefulness to signers because sign language has its own grammar. It often relies on the context of the conversation. Signs can even change depending on who the subject of the conversation is. Sign language, however, can translate to gloss even though this is still limited.

What are glosses?

In linguistics glosses are a brief explanation or semantic label of a word, phrase or sentence. Glosses often label signs using English words to dictate the meaning of the sign.

For example, the English sentence "Are you going to the shop?" might translate in ASL to "SHOP YOU-GO?". This structure omits helper words and uses facial expression to indicate a question. This is similar to how glossing can be used for spoken languages too.

For example, the German Es geht mir gut may be glossed as "It goes to-me good". The hyphen shows that, "to-me" refers to a single word in the original sentence. A true English translation of this expression would be something like, "I am doing fine."

Gloss is a useful tool for sign language production. However, it does not give you any description of how to make the sign.

The Data Challenge

One of the biggest challenges is the lack of publicly available datasets for continuous sign language. Most available datasets focus on isolated signs. These lack the contextual and transitional information present in continuous sign language, making it harder to train models for real-world communication.

Training AI for sign language is challenging because unlike text or audio sign language exists in physical space. This means that all training data must be video based. This greatly increases the volume and complexity of data required. To recognize a single sign reliably in one language, a model may require thousands of video examples. These must include different signers, lighting conditions, camera angles and signing styles. To cover a functional vocabulary the data requirement grows exponentially.

One of the largest datasets available is the DGS corpus which contains examples of conversational German sign language. This has meant a lot of the models trained using conversational sign language use this dataset, often translating DGS to German. There is a surprising lack of publicly available continuous ASL and BSL datasets. There is a publicly available repository to bring together many sign language datasets. However, each dataset still requires a license. Because most datasets have limited signing hours and few signers, models can develop biases. These may include racial or demographic bias, caused by a lack of diversity in the data. This needs careful consideration when creating a dataset.

Existing solutions

SLTUNet

SLTUNet is an encoder-decoder transformer architecture designed for sign-to-text translation. There are 3 different pretrained models provided based on three datasets: DGS corpus and RWTH-PHOENIX-Weather-2014 for German sign language (DGS) and CSL-daily for Chinese Sign Language. Tests on the German models, gave underwhelming results.

A simple input video of a woman signing “Hello, good morning everyone” in DGS was mistranslated into “Then he is very excited” or “in the west and northwest the clouds are loosening” using models trained using DGS corpus and RWTH-PHOENIX-Weather-2014 respectively.

The RWTH-PHOENIX-Weather-2014 model always gave weather-related phrases. This shows how domain-specific training leads to significant hallucinations when exposed to a general sign input.

This is exacerbated by the Zipfian distribution of vocabulary in conversational datasets where most words occur infrequently (fewer than 10 times in the training data). This leads to unreliable predictions. Despite the promising results on the test datasets described in the research paper, SLTUNet is not yet suitable for general sign language translation.

Sign.mt’s sign language processing project

This is a highly ambitious open-source project. It aims to support a two-way translation between multiple different spoken and signed languages. Sign.mt is developing a pipeline to produce sign language videos of an avatar. These videos can generate datasets to train models for sign- to-text translation. The team has two separate pipelines for sign language production.

1. Rule-based approach (illustrated in Figure 2)
    1. This first translates text into gloss by reducing the word to its root form and syntactic reordering and then applying grammatical rules.
    2. The glosses are then used to generate poses using look-up tables. These can then be rendered into animations.

2. Machine learning approach
    1. Translates the input text directly into Sutton SignWriting, a symbolic written system for sign languages.
    2. This SignWriting can then be converted into pose data and then synthesized into videos

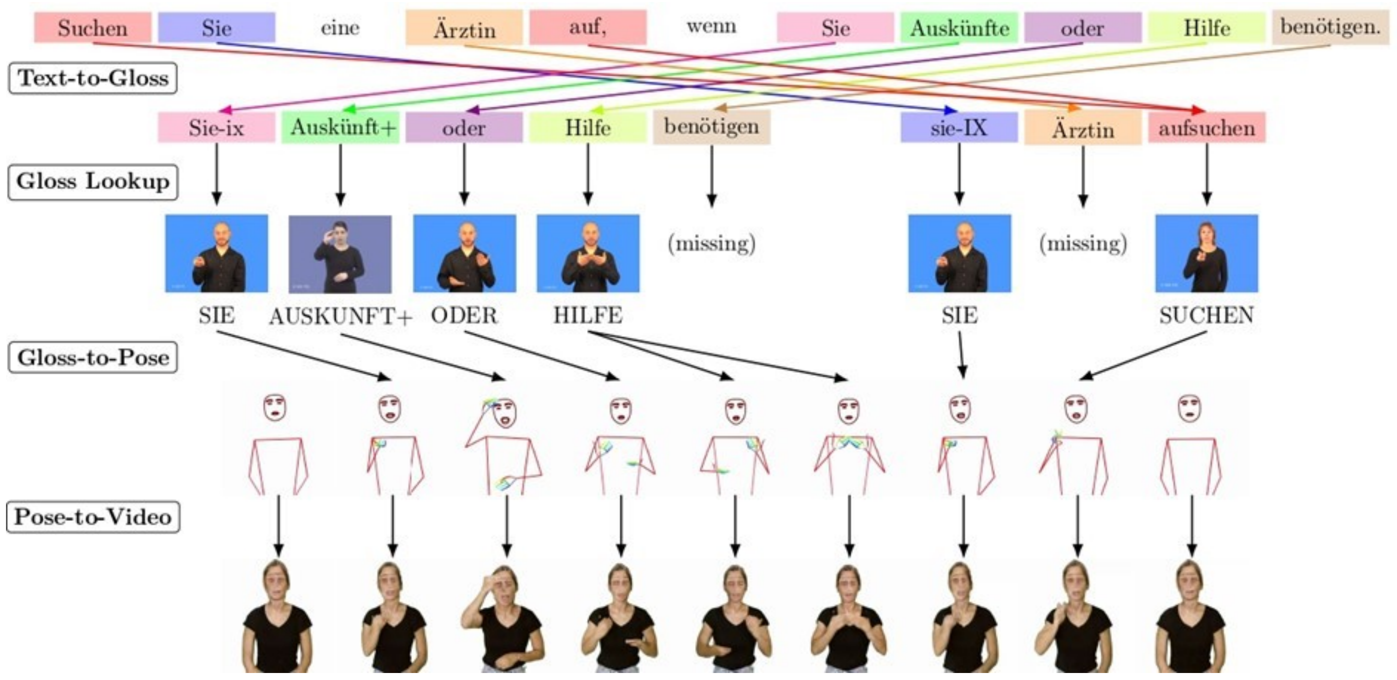


Figure 2: Pipeline of the gloss sign language translation approach.

The output sign language is low quality at the moment with only 10-40 % accuracy. Avatar-based signing is sensitive. Many deaf users find avatars inauthentic or visually unnatural, which can affect how signs are understood. Developers must approach this carefully and gather user feedback from deaf signers.

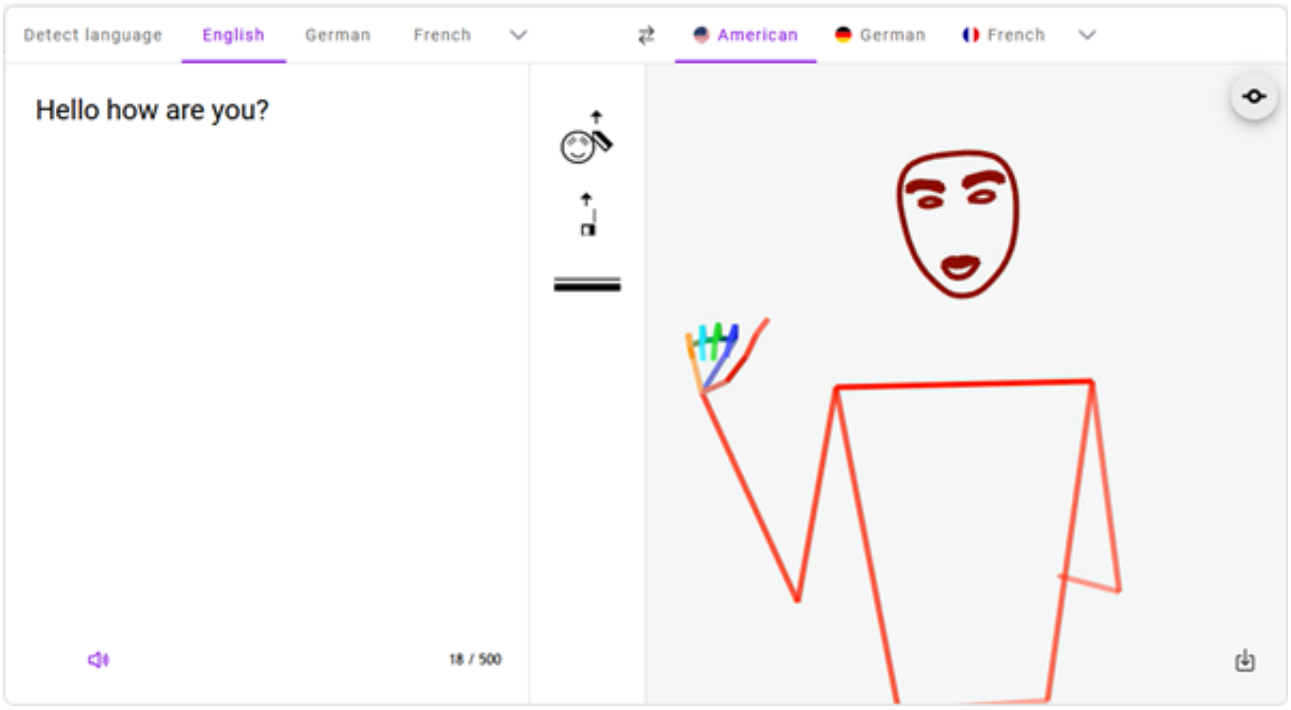


Figure 3: Screen shot of sign.mt's webpage demonstrating their sign language production technology https://sign.mt/

Sign.mt is also developing reverse translation from sign videos to text. This involves segmenting continuous signing into individual isolated signs, converting them to SignWriting, and finally translating this into text. This pipeline is still under development.

The various repositories for their comprehensive work are found here: https://github.com/sign-language-processing.

VAC_CSLR

This model introduced visual alignment constraints. These constraints guide how visual features align with the corresponding glosses. It forms the basis for several newer systems, including SLTUNet. Unfortunately, like SLTUNet, it relies on the PHOENIX-Weather dataset, which narrowly focuses on weather reports. This narrow scope makes it unsuitable for conversational use.

Handtalk

Handtalk is a polished commercial application which translates text into avatar signed videos of either Brazilian Sign Language (Libras) or ASL. It uses a gloss based system, similar to the one shown in figure 2. The avatar is expressive, and users can adjust the signing speed. The app is not open source, but its functionality provides inspiration for the potential of real-time sign language production.

Signapse

Signapse is a UK-based startup using AI to provide sign language translation in transport and public service settings. The team focuses on generating sign language videos for train stations and public announcements. Life-like signers are used over avatars. This improves authenticity and cultural acceptance. Technical details are limited but their work in this field is promising.

SignGemma

Google plans to release SignGemma later this year. It translates signed ASL into English text. A preview video shows promising results which would help individuals understand sign language. However, it has faced criticism from the deaf community, as many argue that it does not help deaf people understand the hearing community. "This still leaves the deaf community without the ability to understand the hearing person. This is built purely to assist the hearing." This highlights the need for bidirectional translation and shows why language production matters. Still, it is great to see Google involved in this technology as we can expect some promising results.

SignWriting

SignWriting is a visual system for writing sign language. Sutton SignWriting is the most widely adopted written form of sign language, however there is no consensus. It uses visual symbols to express the hand shape, palm orientation, movement, location, and facial expression. Symbols are stacked to show how different parts of the sign occur simultaneously. Formal SignWriting (FSW) is a digital representation of these which can be used in computational models.

SignWriting may be useful in machine translation because it acts as a bridge between text and motion. It is possible to use natural language processing (NLP) techniques to translate between text and sign writing. Then it is simpler to translate the SignWriting into pose and animation.

Creating the text to SignWriting App

Sign.mt's Signwriting translation model

After reviewing the available solutions, we chose to work with sign.mt's open-source machine learning model for translating text to SignWriting. We aimed to deploy this on mobile. There are many different approaches to sign language translation, including rule-based systems to translate into gloss and then look-up tables. However recent advances in ML mean the machine learning path offered the most potential.

Sign.mt's work includes many components. For our mobile app, we are using their pipeline for translating text into Formal SignWriting, a symbolic writing system for sign language. The translation model is a sequence-to-sequence neural machine translation system. It uses PyTorch and builds on the Sockeye framework developed by AWS. It features an encoder–decoder architecture with an autoregressive decoder that generates SignWriting characters.

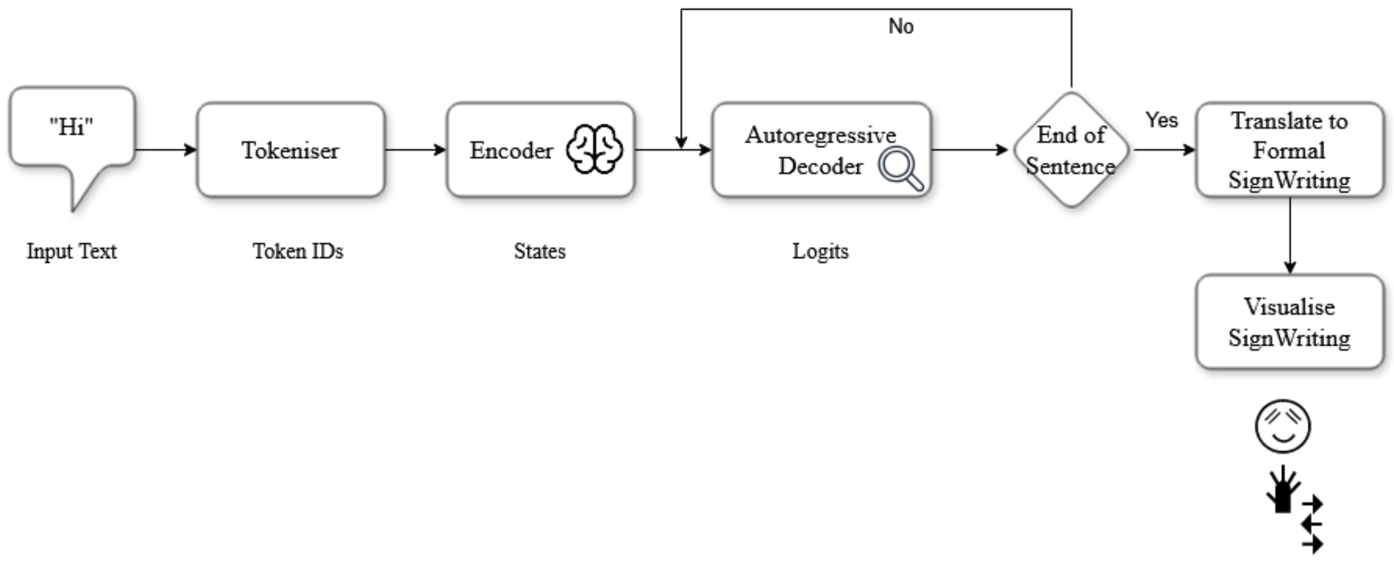


Figure 4: Diagram showing the simplified architecture of the text to Signwriting translation model

To run the full pipeline the following repository is required:

- https://github.com/sign-language-processing/signwriting-translation/tree/main

And the dependencies can be found here:

- https://github.com/sign-language-processing/sockeye
- https://github.com/sign-language-processing/

Despite its potential, the current model is limited by the size and quality of the training data. The model is trained on the SignBank+ dataset, curated by sign.mt. This dataset includes community contributions and multilingual sources. However it is small by machine learning standards. This can cause issues with translation quality, especially for complex and unfamiliar inputs. Nonetheless, there is great value in the model and this work will hopefully help future development of sign language accessibility.

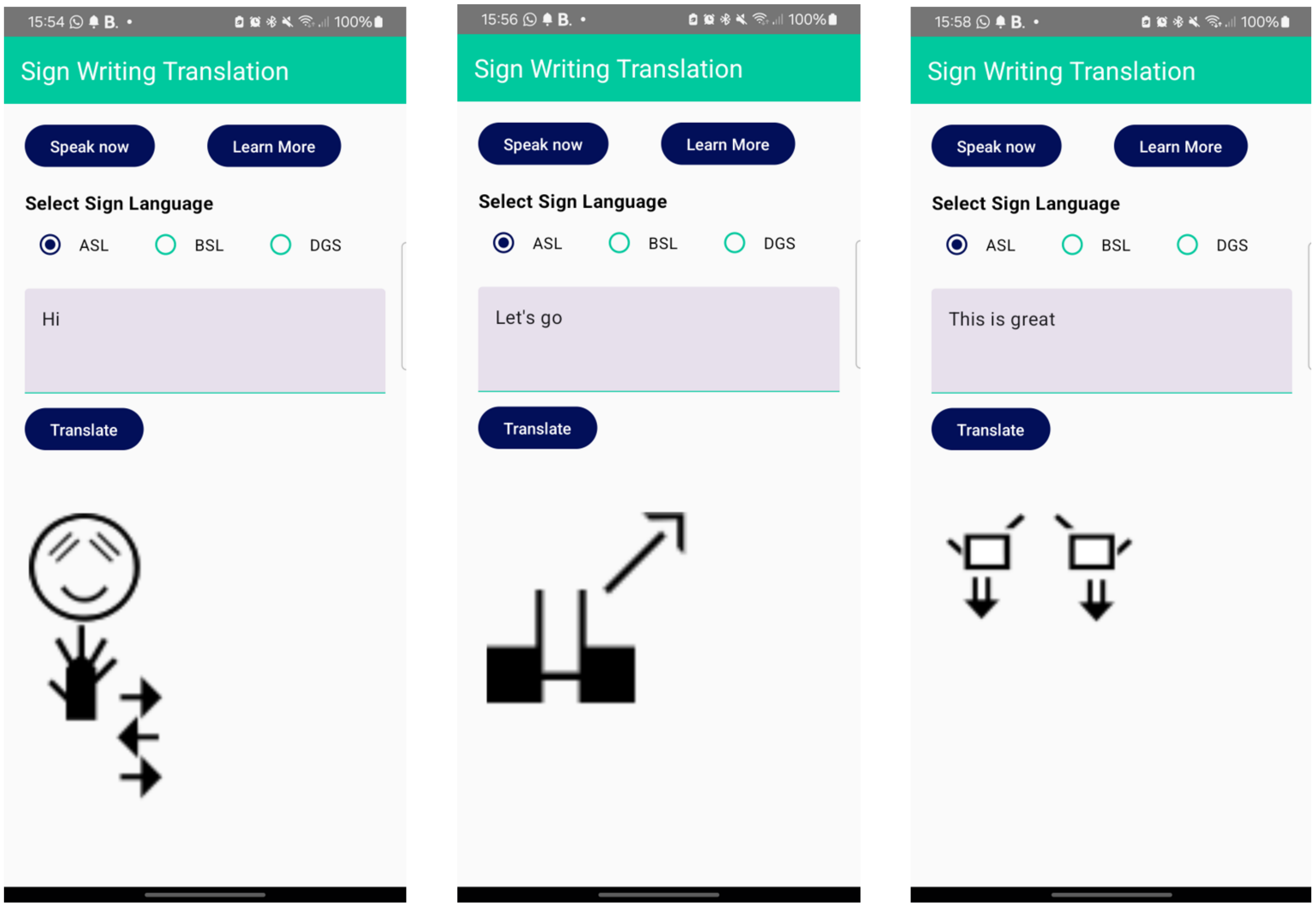
Making the PyTorch model mobile compatible

The project aims to port this model to a mobile app to improve accessibility. On-device inference enables translation on the go. The model provided by sign.mt is written in PyTorch. PyTorch is not optimized for mobile, so it uses more memory, power and time. The model needed to be converted to TFLite or ExecuTorch which are designed to leverage mobile hardware acceleration.

We used Google's ai_edge_torch to convert the PyTorch model to TFLite. Porting the model to TFLite involves many challenges. One key issues is that TFlite does not support dynamic tensor sizes. We adapted the PyTorch model to have fixed-sized tensors by padding them and then using masks to only compute the necessary part of the tensor. This does unfortunately increase computational needs. Due to the structure of the model, it was necessary to convert the encoder and decoder into separate TFLite models. This also created some challenges. The search algorithm could not be converted to TFLite, so we needed to implement greedy search or beam search in Kotlin to find the best solution for each decoder. We chose greedy search because it was simpler to implement in the given time frame, and runs faster on device. This approach reduces accuracy. With more time, implementing beam search could improve the output quality particularly for longer sentences. PyTorch's jit.trace operator is not supported by TFLite, so we replaced it throughout the model.

To build the Android app, we ported all inference logic, tokenization, and result visualization from python to Kotlin. Speech-to-text using Android's built-in API was also added and provided an easy input method. Visualizing the SignWriting is done by creating bitmap images using the Sutton SignWriting fonts. These images appear in the user interface.

The project produced a lightweight mobile app that translates speech or text into visual sign writing. This is an early step forward to building a real-time sign language assistant.



Select Sign Language: [✓ ASL] BSL DGS

Hi

Let's go

This is great

Figure 5: Example screenshots of the SignWriting translation app

Call to Action

Sign language translation is not only a technical challenge, it is a human-centered AI problem. At its core this project is about making communication more inclusive and accessible for deaf and hard-of-hearing communities around the world.

Despite recent advances in machine learning, one thing remains clear to me after undertaking this project: technology alone is not enough. Current open-source machine learning models do not allow deaf and hard-of-hearing people to easily communicate with hearing individuals. The main bottleneck in sign language translation is not model architecture or compute power, but the lack of large, diverse, and annotated sign language datasets. Collaboration with the deaf community is essential when working on sign language technology. Meaningful progress requires collaboration among developers, researchers, linguists, and deaf signers.

The potential for this technology is enormous. If tools like the one developed in this project were improved and deployed at scale the impact could be transformative. Just some of my thoughts are:

- Instant SignWriting and video translations could be added to online documents, helping signers understand written content in their own visual language.
- Live captioning could be added to media sources and video conferences with Signing avatars, making it more accessible than with plain subtitles.

Whether you are a developer, researcher, educator, or designer, you have a role to play:

- Contribute to open-source projects like sign.mt and the sign-language-processing open source project.
- Help collect and annotate sign language data, especially from underrepresented sign languages.
- Build tools that center deaf users, not just technical feasibility.
- Advocate for inclusive features in apps, media, and public services.

By investing time and care into this space, we are not just solving a machine learning problem. We are amplifying voices that have too often been left out of the conversation.

Conclusion

This project was both a technical and human exploration. It began as an attempt to build a sign language translator for video conferencing. It evolved into a broader study into the state of sign language technology, its challenges, and potential. This project revealed how complex and under resourced sign language translation remains. Sign languages are visual, contextual, and expressive. They require a different approach to machine learning. Despite the technical barriers, this is a space with the potential to make a real-world difference.

We extend a sincere thank you to the team at sign.mt. Their open-source tools and research provided the foundation of this work. After speaking with them my eyes were opened to just how complex this problem is. Sign.mt's open-source contributions have pushed the field forward and made it more accessible to others building inclusive tools.

The goal of this project was not just to build an app. It is to raise awareness about the importance of sign language accessibility in this digital age. Whether it is adding SignWriting overlays to documents, building smarter captions for TV, or designing avatars that communicate authentically in sign language, this technology has the power to transform lives. Technology must include everyone. Together, we can ensure that the next generation of AI truly speaks to, and signs for, everyone.

If you are interested in other translation work our team has done, please have a look at the following blog posts:

Sing this song in another language, part 1: Creating an ML Pipeline Sing this song in another language, part 2: translating Machine Learning Pipelines to Android