Inferring Baby Sign Language Videos with an Edge Device

Alex To | Aswin Thiruvengadam | Dan Ortiz | Jeffrey Laughman

Abstract - This document provides an implementation and evaluation for an approach to advance the machine interpretation of non-verbal forms of communication from “static recognition” (images) to “motion recognition” (video) by incorporating the analysis of body motion and gestures. Baby Sign Language was selected as the gesture-based language due to its gaining popularity since the 1980s[1] and heavy use of gestures. The gesture recognition model demonstrates that a pre-trained SWIN Transformer can be applied to a new set of gesture data inputs to generate a new output layer with a strong probability to correctly infer gesture intent (with five gestures consistently achieving top 1 accuracy, 90% with independent testing set). The dataset was independently curated based on original video capture scenarios by the authors as well as published videos from YouTube (See Work Cited). Inference is demonstrated as occurring on NVIDA Jeston devices to approximate edge devices and to show potential applicability to deploy in non-controlled real-world scenarios where non-verbal communication inference could occur.

Key Terms - Vision Transformer | Gesture Analysis | Sign Language | Edge Inference

Repository - https://github.com/atox120/w251_fp

Introduction and Problem Statement

The success of convolutional neural networks for image recognition and single-frame analysis lends support for the application of analysis to multi-frame or “video” scenarios. One of these expanded applications of interest to the authors is the progression with sign language from letter recognition that mostly uses fixed hand position (and which is well covered) to gesture based sign language[2].

The specific gesture based signing being explored is baby sign language. Baby sign language is not a language, but rather it refers to a series of gestures that can be taught to children beginning as early at 8 months of age[3] to help them communicate basic needs, desires and emotional state. The goal is to introduce to the babies communication skills that can reduce the frustration of parents looking to meet what would be otherwise unknown needs.

[1] https://babysignlanguage.com/basics/history/?v=7516fd43adaa#garcia
[2] Example: Some single-letter hand recognition sign recognition analysis excludes the letters j due to it including a gesture
[3] http://parentingscience.com/baby-sign-language/

This paper explores the benefit of having baby sign language inference available on an edge device, specifically as an improvement to baby monitor tools. Observations and experiences by the authors with baby monitoring tools indicate current devices focus on providing audio and/or visual representations without context of the actions. The capability for a baby monitor tool to detect and communicate a baby sign language (BSL) gesture can help primary caregivers understand potential needs when they are not continually observing the baby directly or from a monitor. It can also assist non-primary caregivers on how to meet a baby's needs if they have no experience or training in BSL gestures. Understanding of a baby's intent supports more direct communication and reaction than a passive observance of safety.

BSL gestures are anticipated by the authors to be a strong candidate for baby monitor tools due to the gestures using a large range of motion that are beneficial to detection at greater distances, which is often the situation with the placement of baby monitor tools.

Related Works

Transformers
Transformers have their roots in NLP, and are based on an encoder decoder architecture. The transformer aims to solve the challenges associated with RNN and CNN architectures with long-range dependencies and training (*Analytics Vidhya*). It does this by avoiding recurrence and implementing an attention mechanism which acts independently from the input/output distance (ashish). As a result, a transformer architecture can train a larger model then other architectures with equal computation power. While its roots are in NLP, *Attention Is All You Need* is foundational for the works cited in this project.

Vision Transformers (ViT)
Vision Transformers were proposed by the Google Research Brain team. When announced in 2020, the Vision Transformer (ViT) became the new state of the art model for image recognition. It does this by breaking the image down into 16X16 pixels. The model then applies a linear transformation which turns the patch into a vector. This is then fed into the transformers and then is classified. While the results beat the state of the art CNN at the time, the ViT tends to be data hungry and requires larger amounts of data to train on compared to its CNN counterparts. The Google team stated they used training sets ranging from 14 million to 300 million images and "...find that large scale training trumps inductive bias" (*Dosovitskiy*).

Model Selection

Initial Considerations
Initial model considerations included both still image and video based models. These included CNN-based models such as AlexNet and GoogleLeNet and transformer based Vision Transformer. However, when reviewing BSL videos, the authors identified that the motion in the sign language gesture was important to capture, leading to the ability to train and infer on a

temporal axis as a critical consideration. An example of this can be seen in the gesture figures within the Data Collection section of this paper. The difference between the “Mommy” and “Daddy” gestures use the same hand formation, but the movements of the gestures differ. The conclusion was models that infer still images may struggle with correctly recognizing the distinction of the same hand formation having alternative starting points and ending points.

These requirements led to two potential models, Shifted Window and Multiscale Vision Transformers. When adding the hardware constraints of edge devices such as the NVIDIA Jetson Nano with 4 gigabyte of shared memory and 472 GFLOPS GPU processing power[4], The Shifted Window approach was selected due to its computational efficiency. In addition, the authors of the SWIN model provided both a pre-trained model checkpoint and github repository to reduce the training requirements for the baby sign language interpreter.

Transformer architectures have also been demonstrated to train well in parallel configurations and can achieve a magnitude increase in training time compared to a similar architecture based on recurrent neural networks (RNN). An additional advantage of transformers compared to RNN for this analysis is their handling of contextual dependencies, meaning the consideration of frame order.

Model Selection: Video Shifted Window-T

The Shifted Window Transformer (SWIN) is a class of vision transformer models influenced from the success of the Vision transformer (ViT). Both the Video SWIN and the original SWIN networks were designed and proposed by Microsoft Research and its partnered universities. SWIN was originally developed for image recognition and then was extended to video recognition by leveraging “spatiotemporal locality”, or pixels which are closer to each other in the temporal dimension are more likely to be correlated (*Liu, Ze, Video Swin Transformer*). This is then used to approximate self-attention, improving model performance over factorized models.

Shifted Window Transformers

The Vision and SWIN Transformers are heavily influenced from NLP, and as such the concept of tokenizing carries over. Instead of tokenizing a word, the model breaks the image up into small groups of pixels called a patch. These patches are then combined into non-overlapping segments called Windows to perform self-attention. These windows shift as the model progresses and the number of patches in a window is fixed leading to time complexity which is linear to image size.

The shifted window concept describes how the window relates between self-attention layers. “The shifted windows bridge the windows of the preceding layer, providing connections among them that significantly enhance modeling power (see Table 4). This strategy is also efficient in regards to real-world latency: all query patches within a window share the same key set1, which facilitates memory access in hardware.” (*figure 1)*

[4] https://developer.nvidia.com/blog/jetson-nano-ai-computing/

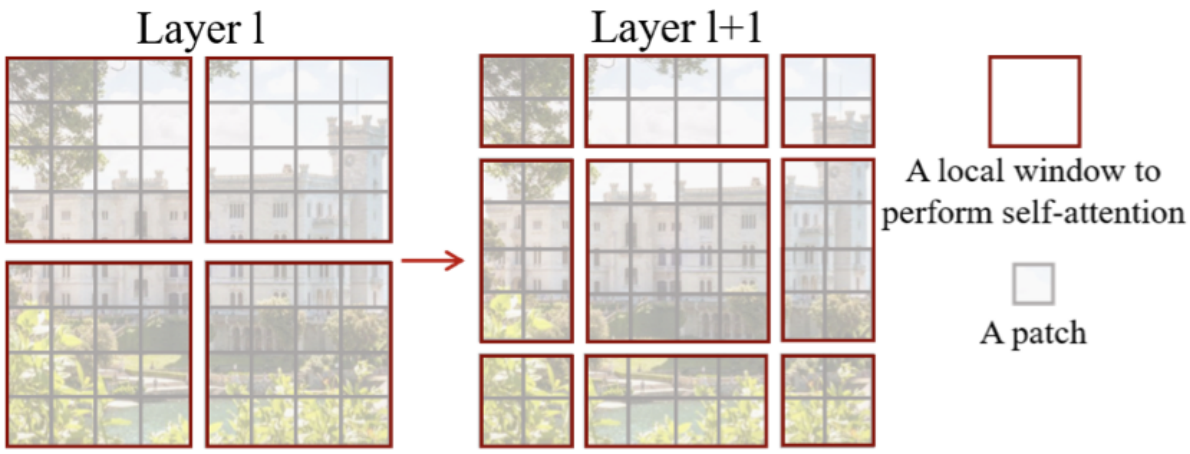


Figure 1: Depiction of how packets/tokens are combined throughout the 2D Swin layers. Note both the tokens and the windows are non-overlapping. Source: Liu, Ze

The model separates the image into non-overlapping patches (tokens). “A linear embedding layer is applied on this raw-valued feature to project it to an arbitrary dimension (denoted as C)” (Liu). The patches are then passed through back to back SWIN Transformer blocks. As the image progresses through the different stages, the model concatenates neighboring patches resulting in a 2X down sampling of the resolution and repeats for stage 3 and stage 4.

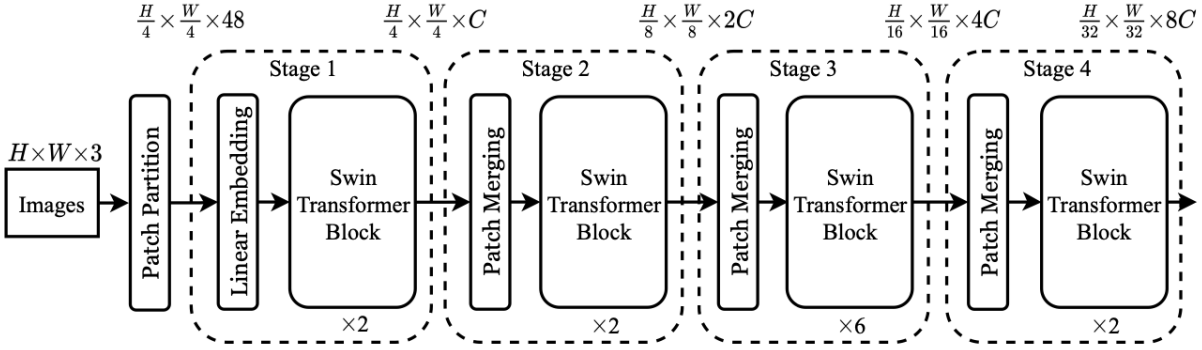


Figure 2: Diagram of the 2D SWIN Tiny (SWIN-T) Model. Source: Liu, Ze

For video processing the model adds the temporal axis as the third dimension in order to train and interpret video instead of images.

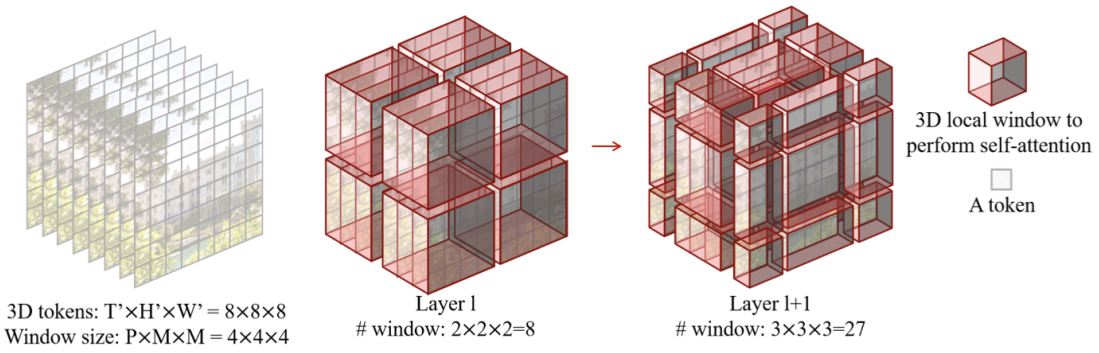


Figure 3: Depiction of how packets/tokens are combined throughout the 3D Swin layers. This takes the 2D Swin concept and extends it into a third dimension, time. Note both the tokens and the windows are non-overlapping in all dimensions. Source: Liu, Ze

Adding the time domain adds a third dimension which impacts how the model breaks the input into non-overlapping patches, or tokens, into a Time X Width X Height patch. The patches are constructed into a PXMXM window based on the layer just like the 2D variant in a non-overlapping method. The window is then shifted across all three dimensions for each layer to "introduce cross window connections while maintaining efficient computation of non-overlapping window based self-attention." Figure 4 shows just how similar the underlying architecture of the model is.

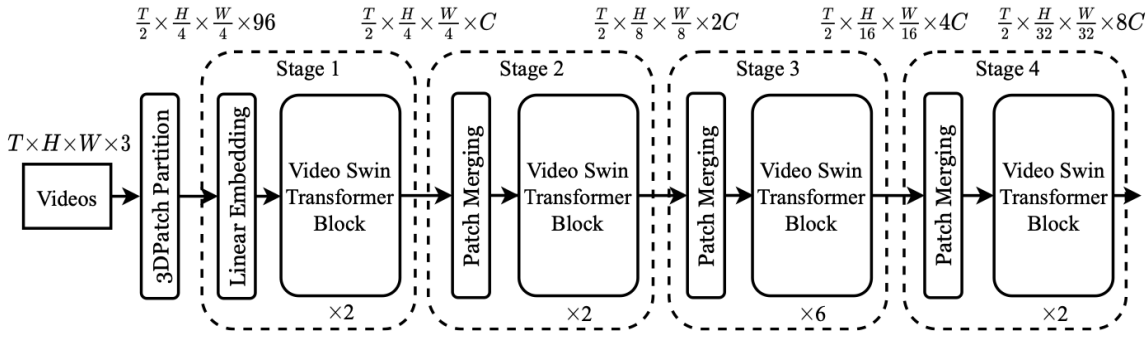


Figure 4: Diagram of the 3D SWIN Tiny (SWIN-T) Model. The architecture extends the 2D SWIN model and extends the third dimension to time. Source: Liu, Ze

One key difference between the image and video transformers relates to the self-attention mechanisms. The size and scale of video tokens makes global self-attention infeasible. According to Liu, local self-attention mechanism, in association with relative position bias, is a good approximation for the general self-attention found in traditional transformer models. The relative position bias used feeds the model the query, key, and value matrices for all real values of the patches in the window during the attention calculation in an attempt to exploit positional information. According to Liu, models which used RPB outperformed those that did not.

Data Collection for Training

Five BSL gestures (or signs) were identified for data collection. The decision to use a relatively small number of signs was purposeful so that the authors could focus on capturing a diverse set of videos for each sign in a range of scenarios. For each sign, the data set includes both original video content of multiple individuals performing the gestures in different environments (combinations of clothing and environments) and video sourced from youtube. The more straightforward use of capturing still images was not possible given the outcome intent.

The five BSL signs with data collection are “All Done”, “Water”, “Mommy”, “Daddy,” and “Poop:”

Figure 6: Signs chosen for the BSL proof of concept. Source: babysignlanguage.com

These gestures were chosen based on their distinctness for inference evaluation, but also two BSL signs (“Mommy” and “Daddy”) were included for the similarity in hand formation and motion. The gestures were also selected due the perceived value in identifying them to support a child’s immediate needs and comfort. Number of training videos across Train, Test and Validation = 280+ total videos.

For model training, the inclusion of original video content and a small set of signs also provided more control in labeling content and ensuring diversity of the training set. The ease of performing BSL signs also readily supported the ability to generate additional video data and not be limited to a public data set that may be fixed in size.

After each video is created (or collected) and labeled against the appropriate BSL signs, its original source details are modified for the training pipeline that includes both resizing and frame extraction:

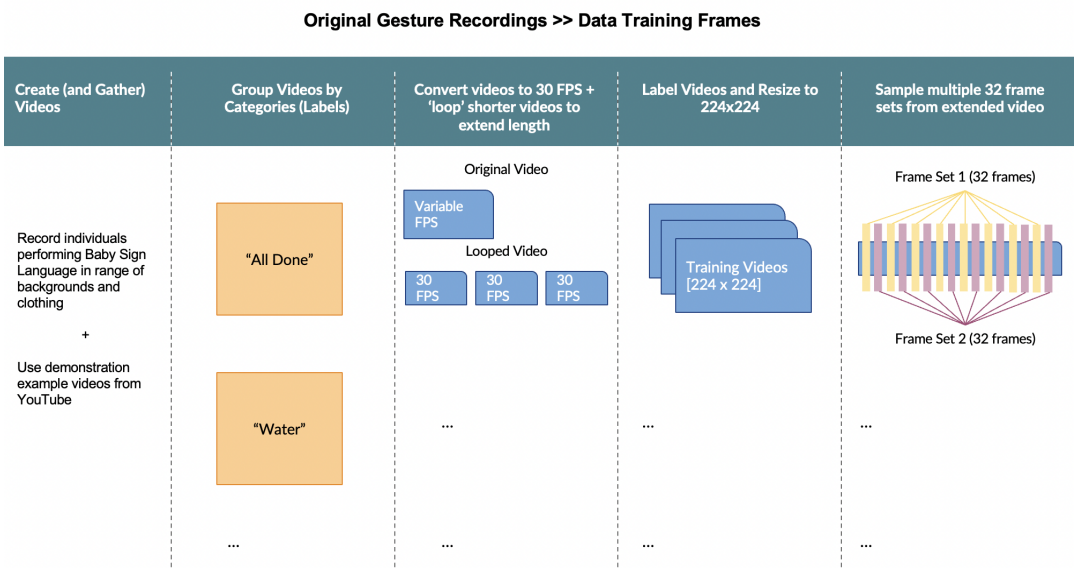


Figure 7: Not show in above image above but included in training pipeline: random flip and random crop of sampled frames

Step	Description	Tools Used
Create Videos	Generate raw videos of individuals performing Baby Sign Language	Original content: Smartphone and personal computer cameras YouTube content: VLC + iMovie
Group Videos	Annotate each raw video to one of the identified signs and store with similar	Google Drive file and folder management
Standardize FPS, loop shorter videos	Convert all video files to 30 FPS, extend shorter videos by looping content	Python
Resize and use mp4 format	Convert videos to 224 x 224 to match pre-trained architecture + use common mp4 format	Python
Create 32 frame samples from each video	Extract multiple samples from each video to generate different samples of gestures in practice	Python

Figure 8: Table outlining steps and tools used to train the BSL SWIN model. Model. Source: Liu, Ze

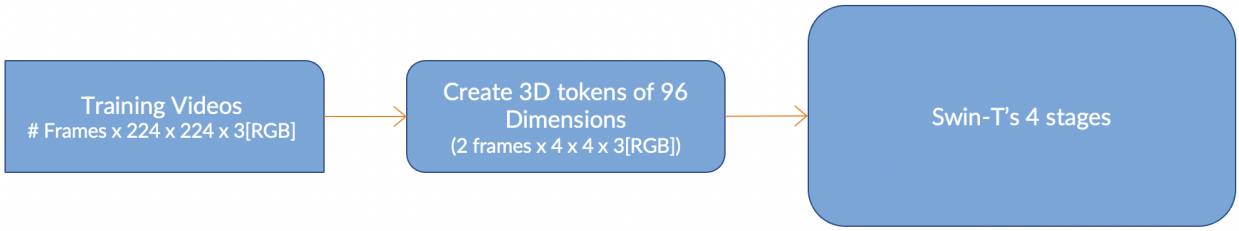
Training Baseline Model

The model in this analysis (A_SWIN) uses the parameters from an existing Video Swin Transformer[5] tiny version (Swin-T) architecture. The version of Swin-T used has previously been pre-trained (not by the authors of this document) on publicly available Kinetics-400 dataset. It demonstrated high-level accuracy in recognizing human interaction events from video

[5] https://arxiv.org/pdf/2106.13230.pdf

clips from Kinetics-400 that contains over 700 human interaction classes[6]. It should be noted that these interaction classes did not include sign language gestures.

The simplified training input process is as follows:



Initial training was performed on virtual machines to train on larger resources (ex: GPU) and to avoid adding any additional constraints tied to local device deployment for initial evaluation. Additionally, the authors recognized that local devices would only be performing inference, not training.

The employed model architecture has 28 million parameters across all layers. Training parameters for A_SWIN are modeled after Swin-T, but do not follow exclusively the same configurations. Parameters[7] for A_SWIN included the use of AdamW optimizer and a cosine annealing learning rate decay with linear warmup of 2.5, with starting learning rate $1x10^{-3}$.Training occurred for 15 epochs on Tesla T4 GPU virtual machine hardware with 8 vCPU VM. There were training calculations using floating point 32 (FP32).

Evaluation of Inference in Cloud (non-Edge Device)
Initial Evaluation of A_SWIN tuned model performance occurred in cloud-based virtual machines prior to Edge device deployment. The evaluation test submitted a pre-recorded video (no live streaming) of an individual performing BSL gestures. The individual's background and clothing were unique from what was included in the training and validation videos to better ascertain inference in other environmental states. Focusing inference results to a probabilistic selection to only instances of well-performed iterations of the five BSL gestures resulted in high accuracy across the five classes:

[6] https://deepai.org/dataset/kinetics-400
[7] https://github.com/atox120/w251_fp/blob/main/store/notebooks/bsl_train.ipynb

		Predicted label				
		all done	Water	Poop	Dad	Mom
True label	all done	2	0	0	0	0
	Water	0	2	0	0	0
	Poop	0	0	2	0	0
	Dad	0	0	0	2	0
	Mom	0	1	0	0	1

Figure 9: Table Top 1 Accuracy = 90%
Table source: https://github.com/atox120/w251_fp/blob/main/store/notebooks/bsl_inference.ipynb

The results provided confidence to proceed to model deployment at the edge device level.

Evaluation of Inference in Edge Device
At Edge Device level, all inference and demonstrations were deployed in a containerized environment with NVIDIA L4T Pytorch image as the base. NVIDIA Jetson devices were used for evaluation (both the Nano and NX series). The Jetson's native Gstreamer[8] package was leveraged to capture and stream live video from an attached USB webcam. Within Gstreamer, the streaming video was downscaled to an input of 224 x 224 pixels and an edge device's GPU compute resources set the memory to 'NVMM' when calling NvVidConv.

To increase inference speed and reduce memory overhead, NVIDIA's DeepStream SDK initialized video capture and read the video buffer directly into a Numpy array/tensor. Specifically, we instantiated a 'pipeline' object in python to initialize the video stream from a file or webcam and added a probe to read the buffer information from the sink pad. This allowed for the avoidance of using the python openCV module and its additional memory overhead during the video capture, inference and preprocessing. The captured buffer information could then be used as an input to a numpy/pytorch to map the memory address of the buffer and instantiate a new copy as a tensor, which was used for downstream, preprocessing, inference and postprocessing. This resulted in an overall performance improvement; the changes reduced memory overhead during runtime and enabled us to run inference on a live webcam stream within a containerised environment on NVIDIA Jetson devices.

After reading the buffer and instantiating the image tensor, we queued the input into a single batch of 32 frames, and then performed batched preprocessing including mean and standard-deviation normalization before loading the model onto the GPU for inference. Inference was successful and could be done at the model's original FP32 as well as reduced or quantized FP16 calculations. For comparison of speed, the time between a new inference after the prior is completed is referred to as "inference lag time." Using a NVIDIA Jetson NX with

[8]https://developer.download.nvidia.cn/embedded/L4T/r32_Release_v1.0/Docs/Accelerated_GStreamer_User_Guide.pdf

FP32, a new inference can occur approximately every 29 frames of a video. Using the same hardware at FP16, the “inference lag time” is reduced and there is a successful inference after approximately every 21 frames.

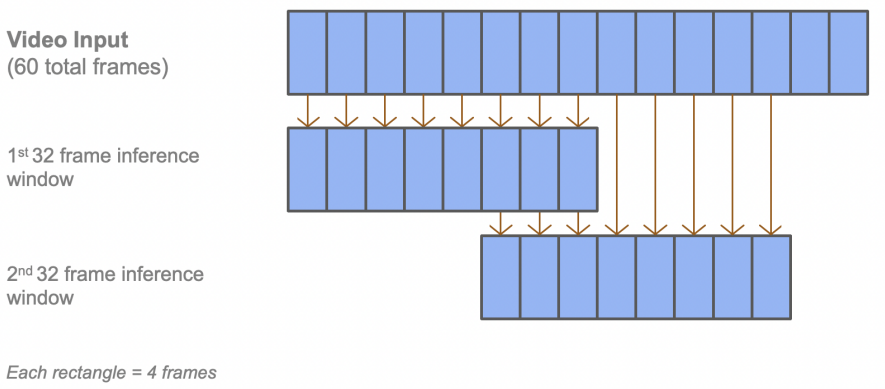


Figure 10: Illustrative example of “21 frame inference lag time” for FP16 deployment

Future Possibilities and Challenges

Successful deployment on NVIDIA Jetson devices provide an opportunity for this inference to be used in local situations while respecting family privacy and security (example: on device inference to prevent the need of streaming video of children over internet connections).

The authors consider potential use cases for this trained model’s inference to include:

- Opportunity for siblings and other family members of a child to practice BSL with each other to then teach them to a baby
- Opportunity for non-primary caregivers of a baby not versed in BSL to understand a baby’s intent when the baby may have expectations that its gestures are understood. Example of non-primary caregivers could include a sitter or visiting relative

The authors consider potential deployment scenarios to include:

- Baby monitoring cameras that can alert parents if a specific BSL gesture is performed (Example: “poop” for a diaper change while baby is in a crib)
- Free gesture-recognition focused apps for smartphones and tablets in partnership with a foundation (example: https://ai4good.org/). The authors note any deployment on a smartphone or tablet may require model adjustments to support other hardware platforms’ cpu and gpu configurations that are likely dissimilar to those used in this analysis (Example: Nvidia GPUs were used exclusively for training and testing in this analysis)

The authors consider expanding the model training to provide support for:

- General sign language
- Other visual-based communication use cases (ex: To assist in Down Syndrome, a model extension could identify or suggest visual cues, highlight if a caregiver should be using

more repetition. It could also be modified to identify if communication is occurring within proper physical closeness[9])

Conclusion

The authors have demonstrated that Video Swin Transformer architecture, specifically Swin-T implementation, is suitable for edge device or other local hardware deployment for baby sign language gesture inference. The authors' A_SWIN model completed training of five gestures ("All Done", "Water", "Mommy", "Daddy," and "Poop") and can be demonstrated to produce strong accuracy of gesture classification from a live video stream on a NVIDIA Jetson device.

Work Cited

[1] 13th, Ashley February, et al. "History of Baby Sign Language." *Baby Sign Language*, 28 Apr. 2021, https://babysignlanguage.com/basics/history/.

[2] Dewar, Gwen. "Baby Sign Language: A Guide for the Science-Minded Parent." *PARENTING SCIENCE*, PARENTING SCIENCE, 2018, https://parentingscience.com/baby-sign-language/.

[3] Deep Mind. "Kinetics-400 Dataset." Https://Deepai.org/Dataset/Kinetics-400, 29 July 2020.

[4] Nvidia. "Accelerated Gstreamer User Guide - Nvidia." *Accelerated Gstreamer User Guide*, Nvidia, 18 Mar. 2019, https://developer.download.nvidia.cn/embedded/L4T/r32_Release_v1.0/Docs/Accelerated_GStreamer_User_Guide.pdf.

[5] Nvidia. "Jetson Nano Brings AI Computing to Everyone." *NVIDIA Technical Blog*, 13 Dec. 2021, https://developer.nvidia.com/blog/jetson-nano-ai-computing/.

[6] Dosovitskiy, Alexey, Luca Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreitt, Neil Houlsby. "An Image Is Worth 16X16 Words: Transformers for Image Recognition At Scale." arxiv.org. 3 Jun 2021. 10 Apr. 2022.

[7] Liu, Ze, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, Han Hu. "Video Swin Transformer." 24 Jun. 21. 10 Apr. 2022

[8] Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, Baining Gou, Microsoft Research Asia. "Swin Transformer: Hierarchical Vision Transformer using Shifted Window." arxiv.org. 17 Aug 2021. 10 Apr 2022.

[9] Shen Sheng, Zhewei Yao, Amir Gholami, Michael W. Mahoney, Kurt Keutzer. "PowerNorm: Rethinking Batch Normalization in Transformers." Proceedings of Machine Learning Research." 2020. 10 Apr. 2022

[9] https://belovedshepherd.com/tips-communicating-with-a-person-with-downs-syndrome

[10] "Transformers in NLP: State-of-the-Art-Models." *Analytics Vidhya*, 23 July 2021, https://www.analyticsvidhya.com/blog/2019/06/understanding-transformers-nlp-state-of-the-art-models/."

[11] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. "Attention Is All You Need." arxiv.org 6 Dec 2017. 10 Apr. 2022.

[12] Savoie, Louise. "Tips: Communicating with a Person with Down's Syndrome." *Beloved Shepherd LLC*, Beloved Shepherd LLC, 8 July 2019, https://belovedshepherd.com/tips-communicating-with-a-person-with-downs-syndrome.

Training Data Work Cited

"All Done" Youtube, uploaded by Baby Sign Language, 26 Aug. 2020, https://www.youtube.com/watch?v=-ISr09a0aak

"All Done in Baby Sign Language, ASL". YouTube, uploaded by TalkBoxMom, 25 Sept. 2014, https://www.youtube.com/watch?v=DBCnRoOcsQQ

"Baby signing All Done", YouTube, uploaded by DefinitiveBabysigning.com, 14 Feb. 2013, https://www.youtube.com/watch?v=PP_1h3e8dGE

"Baby Sign Language | Baby Songs | Baby Sing Time," YouTube, uploaded by Pocket Preschool, 16 Jul. 2015, https://www.youtube.com/watch?v=0uK1YiHNFtU

"Baby Sign Language: Dad." YouTube, uploaded by PlanoPublicLibrary, 26 Apr. 2020. https://www.youtube.com/watch?v=_t-AgBtpxTU

"Baby Sign Language: Father, Dad, Daddy, Papa" YouTube, uploaded by Heather Johnson, 15 Dec. 2018, https://www.youtube.com/watch?v=vJEsWAX6F9c

"Baby Sign Language: Water." YouTube, uploaded by PlanoPublicLibrary, 2 Mar. 2016. https://www.youtube.com/watch?v=qF7h4FXrKDQ

"Baby Sign Language for Mom and Dad." YouTube, uploaded by signing4baby, 21 Nov. 2012, https://www.youtube.com/watch?v=8eSowIpE640

"Baby Sign Language: Mother, Mom, Mommy, Mama.", YouTube, uploaded by Heather Johnson, 15 Dec. 2018, https://www.youtube.com/watch?v=v2S86e8LRoI

"Baby Sign Water." YouTube, uploaded by DefinitiveBabySign.com, 14 Feb. 13, https://www.youtube.com/watch?v=8H_atWW0dRI

"Dad in Baby Sign Language, ASL". YouTube, uploaded by TalkBoxMom, 3 Oct. 2014, https://www.youtube.com/watch?v=c0PZU6t_ZeY

"Mom in Baby Sign Language, ASL" YouTube, uploaded by TalkBoxMom, 13 Oct. 2014, https://www.youtube.com/watch?v=MbaR-VVKCdM

"Finished or All Done" Youtube, uploaded by Baby Sign Language, 24 Aug. 2020, https://www.youtube.com/watch?v=sYeARHLL5sA

"Poop in Baby Sign Language, ASL". YouTube, uploaded by TalkBoxMom, 11 Nov. 2014, https://www.youtube.com/watch?v=NWwqyU8-yWo

"Poop in Baby Sign Language" Vimeo, uploaded by Baby Sign Language, 12 Dec. 2012, https://vimeo.com/55280078

"Poop" Youtube, uploaded by Baby Sign Language, 28 Aug. 10, https://www.youtube.com/watch?v=-lSr09a0aak

"Water" Youtube, uploaded by Baby Sign Language, 10 Dec. 2020, https://vimeo.com/55280078