A Novel Sign Language Translation Model: Gloss-Free Video-to-Sequence Translation Using Transformer Encoder and LLM Decoder

Mac Ya
School of Engineering
Computer Science
maaac@stanford.edu

Evy Shen
School of Humanities and Science
Symbolic Systems
evyshen@stanford.edu

Jeff Liu
School of Engineering
Computer Science
jeffliu0@stanford.edu

Abstract

We developed a gloss-free model to predict English text sequences for American Sign Language (ASL) videos. Our approach employs DINOv2, a Vision Transformer (ViT), for robust video feature extraction. These features are adjusted and normalized via linear projection for better training stability. To capture temporal dependencies between video frames, we use an LSTM decoder. For text generation, we leverage the BART tokenizer, a powerful pretrained language model, to process ground-truth captions. We experimented with combining both sentence-level and word-level ASL videos in the training dataset. This design allows our model to encode visual information from videos and generate corresponding textual descriptions. Our training datasets were: the WLASL Dataset, a collection of 11980 ASL videos representing 2000 English words as well as the How2Sign Dataset, consisting of more than 80 hours of parallel corpora of sign videos,including their respective speech, English transcripts, and depth information. Our metrics for testing were primarily Cross-entropy loss and BLEU, a standard metric for translation performance. Our best-performing model given sign video representations achieved a BLEU score of 11.35%, compared to 8.03% in another published paper's model trained and tested on How2Sign. We found that mixing the training dataset with word-level data and sentence-level did not improve the model performance and increased the variance of the dataset.

1. Introduction

Communication for deaf and hard-of-hearing communities with the rest of society is difficult for those who are unfamiliar with Sign Languages. Isolated Sign Recognition, which is the identification of isolated signs, has been extensively studied, but Sign Language Translation (SLT), the translation of continuous signs, still remains a challenging and growing field of study for computational sign language research [19]. There are over 300 sign languages globally, each consisting of manual articulations as well as non-manual elements, such as facial expressions, body poses, or mouth motions. As sign language does not have one-to-one mapping from one word to one movement and contains both spatial and visual elements, this poses unique challenges for deep learning models to segment sign movements and generalize semantic meaning.

With each of us having experienced language barriers, we were motivated to pursue making SLT as mainstream as the prevalent and advanced translation services for spoken languages. As a long-time bottleneck for research on SLT has been the lack of a parallel corpus of sign videos, real-time alignments of sign videos with their spoken translations, the advent of new datasets providing these necessary parallel corpora of sign videos and their aligned spoken transcriptions has opened up opportunities for research.

Historically, the highest performing sign-to-text models have involved gloss intermediaries [1]. Glosses are direct English transcriptions of Sign Language, containing semantic information and have one-to-one mappings with sign movements. Gloss provides value in helping the model learn the location of semantic boundaries in continuous sign language videos and understand the sign language video globally [21]. However, gloss-labeled datasets are labor-intensive to create and sparse, meaning models that are trained on those datasets are limited to the vocabulary and domain in which they were trained on. As such, gloss-free models are a growing field of exploration, and we were motivated to explore the potential of leveraging different architectures to facilitate a high translation performance for a gloss-free model.

Commonly, SLT model architecture involves transformers, the standard for sequence to sequence tasks, for visual feature extraction[15, 1]. Video inputs are typically fed into a Visual transformer (ViT) to extract visual features. Many gloss-free models competitive with gloss-based models replicate the benefits that gloss provides through utiliz-

ing knowledge obtained from pre-trained language models such as BART [9] and novel attention mechanisms. This insight motivated us to explore the potential of leveraging Visual Transformer and BART to facilitate the conversion of sign videos to spoken English text.

We explored different models, including DINOv2 and BART for text generation, with fine-tuned parameters to find the best-performing model for accuracy in English text prediction. The input to our model is videos from the How2Sign [6] and WLASL [10] datasets. The output is English text. By utilizing DINOv2 for feature extraction and BART for processing ground-truth captions and extracted features, we aimed to output a predicted English translation. Our approach involved inputting a mix of sentence-length (How2Sign) and word-length (WLASL) ASL videos along with the sentence-level English sequence and word-level Gloss sequence in the training dataset and investigating various structures involving ViT and BART, hoping to increase performance compared to previously published models.

To our knowledge, no paper has previously attempted this specific combination of DINOv2 for visual feature extraction and BART for spoken text generation in the context of Gloss-free ASL to English translation. Additionally, no prior research has mixed sentence-length and word-length ASL videos in the training dataset. Since sign language datasets are sparse at large, the conceptual benefit of this approach lies in creating a more robust and versatile model that can handle varying lengths of input data, from individual words to complete sentences. This comprehensive training strategy aims to improve the model's ability to generalize across different types of ASL input.

Our model outperformed several existing models, including the first model trained on the How2Sign dataset[15]. This demonstrates the potential of our combined use of DINOv2 and BART, as well as our innovative approach to mixing different lengths of ASL video data in training, ultimately setting a new benchmark in ASL to English translation tasks.

2. Related Works

2.1. Gloss-Based Transformer Approach

Camgoz et al. [1] introduced the Sign Language Recognition Transformer (SLRT). This state-of-the-art encoder transformer model is trained using a Connectionist Temporal Classification (CTC) loss to predict sign gloss sequences. The SLRT model extracts spatial embeddings from sign videos and learns spatio-temporal representations, which are subsequently fed into the Sign Language Translation Transformer (SLTT), an autoregressive transformer decoder model that predicts one word at a time to generate corresponding spoken language sentences.

The evaluation of their approach on the PHOENIX14T[1] [7] dataset revealed significant improvements in both sign language recognition and translation. The proposed Sign Language Transformers outperformed existing models in translating sign videos to spoken language as well as gloss to spoken language. Notably, their translation networks achieved more than double the performance in certain scenarios, evidenced by BLEU-4 [13] Scores increasing from 9.58 to 21.80. Additionally, they established new baseline translation results using transformer networks for various text-to-text sign language translation tasks, setting a new standard for future gloss-free SLT research in this domain.

2.2. Gloss-Free Approaches

Gloss-based models provide better performance [20] because gloss provides supervision in alignment information for the model to focus on important local areas. However, gloss-free models aim to replicate the benefits that gloss brings through the use of knowledge from pre-trained language models.

Yin et al.'s [21] novel gloss attention SLT network (GASLT) exemplifies this approach. Their model uses BERT to transfer knowledge of sentence-to-sentence similarity from the natural language model to an attention mechanism, helping it understand sign language videos at the sentence level. By leveraging BERT, Yin et al. were able to capture the similarity relationships between sign language videos by inputting natural language sentences into sentence BERT to calculate cosine similarity. This information was then used to aggregate all video features output by the encoder into an embedding vector representing the entire sign language video. The proposed GASLT model significantly outperformed existing gloss-free methods. More recent (2024) gloss-free approaches proposed the use of pre-trained LLM models for the translation task. Wong et al. [19] proposed an innovative pseudo-gloss mechanism within a novel framework for sign language translation. This framework leverages large-scale pre-trained vision and language models, incorporating lightweight adapters to achieve gloss-free translation. The pretraining strategy enables the encoder to learn sign representations from automatically extracted pseudo-glosses, eliminating the need for gloss order information or annotations.

3. Methodology

3.1. Design Rationale and Motivation

Motivated by the significant advancements reported by Camgoz et al. [1] and Wong et al. [19], we aim to develop a new model that leverages the power of the attention mechanism in transformer models and the advantages that large language models (LLMs) have in handling the unique

[1] RWTH-PHOENIX-Weather-2014T

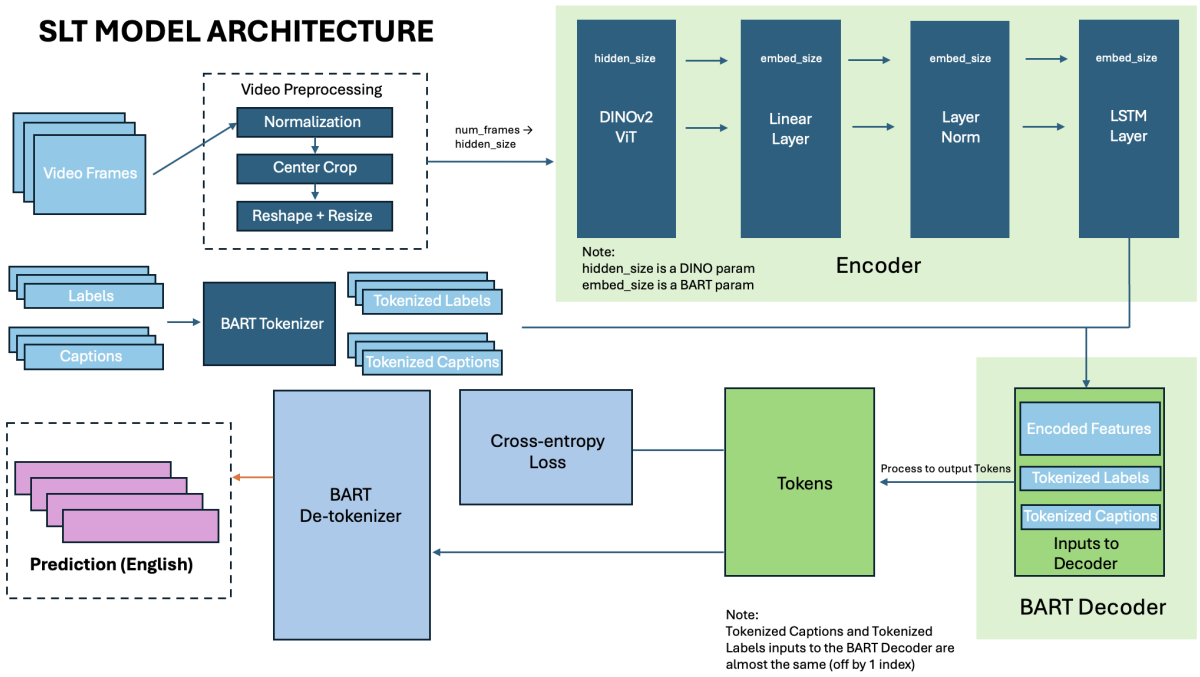

Figure 1: Proposed Custom SLT Model architecture, illustrating the various components and their interactions.

rules of glosses, as highlighted by Wong et al. Additionally, we aim to create our model using a gloss-free approach for its scalability and generality. Gloss-based models are typically limited to the specific sign language they were built for and cannot be easily adapted to other languages due to the limitations of Gloss-Spoken translators, which restricts their general applicability.

With these motivations and the various advantages discussed in prior works, this paper introduces a novel architecture for Sign Language Translation (SLT) that integrates multiple deep learning components to address the unique challenges of translating sign language videos to text. Our model [1] uses a Vision Transformer (ViT) encoder to extract features, which are then processed through an LSTM layer. This combination allows for the capturing of extensive feature information while retaining spatial-temporal features. Additionally, we employ a language model decoder to better manage the differences in order and grammatical rules between glosses and spoken language.

3.2. Overall Model Architecture

Our architecture [1] consists of an Encoder-Decoder model with DINOv2 (ViT) for image encoding, with an LSTM layer attached to the encoder for further processing and BART Decoder (LLM) for English text generation.

Note: packages used in implementation are listed in Appendix 8.1.

3.2.1 Temporal Downsampling

To aid in the efficiency of our translation model while preserving temporal information, which must process sequences comprising hundreds of frames, we employed temporal downsampling after specific layers within our encoder, which reduced the temporal dimension from T^* to $\frac{T^*}{3}$. Initially, inputs had 24 frames per second. After downsampling, it was 8 frames per second.

3.2.2 DINOv2 - Vision Transformer (ViT)

The Vision Transformer (ViT) [2] serves as the encoder for extracting powerful features from video frames. Leveraging its pre-training on large-scale image datasets, ViT excels in capturing intricate visual details crucial for understanding sign language videos.

In this study, we specifically employ DINOv2 (`facebook/dinov2-small`), a self-supervised method applied on ViT developed by Meta, known for its robust feature extraction [2] across various visual tasks. The DINO (Distillation with No Labels) model facilitates the learning of high-quality visual features without the need for labeled data, making it highly effective for extracting spatial features from video frames, aligning with our needs to preserve critical spatial-temporal features of sign

[2]DINOv2 is proven to have strong semantic segmentation and depth estimation abilities.

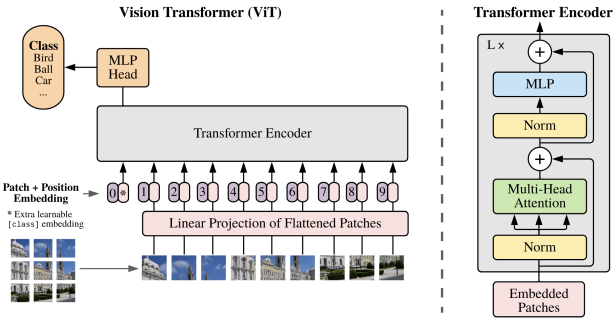

Figure 2: Vision Transformer (ViT) Architecture. Image is adapted from Dosovitskiy et al. [5]

language videos. Wong et al. [19] demonstrated in their paper that an adapted ViT model, trained with DINOv2, outperforms a ResNet18 spatial backbone in sign language recognition tasks. Their findings revealed that the adapted ViT model, when applied with LoRA (Low-Rank Adaptation), achieved superior performance with fewer than three hundred thousand trainable parameters, a significant reduction from the 11 million parameters required by the ResNet18 model.

Given these advantages, we opted to integrate DINOv2 as our ViT encoder, benefiting from its efficient and effective feature extraction capabilities. After inputting the normalized, cropped, and resized images into DINOv2, we will apply a linear projection and layer normalization to the feature vectors produced by the ViT.

3.2.3 Linear Projection and Normalization

After DINOv2, to adjust the feature embeddings to a desired size, we apply a linear projection followed by layer normalization. This step helps in stabilizing the training process and ensures the features are suitable for further processing.

$$\mathbf{E}' = \text{LayerNorm}(\mathbf{W}_{\text{linear}}\mathbf{E} + \mathbf{b}_{\text{linear}})$$

The formula for layer normalization is:

$$\text{LayerNorm}(x_i) = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}} \cdot \gamma + \beta \qquad (1)$$

Where:

- x_i *is the input vector.*
- μ *is the mean of the input vector.*
- σ^2 *is the variance of the input vector.*
- ϵ *is a small constant added for numerical stability.*
- γ *and* β *are learnable parameters for scaling and shifting.*

The output of the `LayerNorm()` will then be passed to an LSTM layer to retain it spatial-temporal feature.

3.2.4 Temporal Aggregation (LSTM)

We capture temporal dependencies between frames using a Long Short-Term Memory (LSTM) network [3.2.4], which is crucial for video data where the sequence of frames contains temporal information as mentioned.

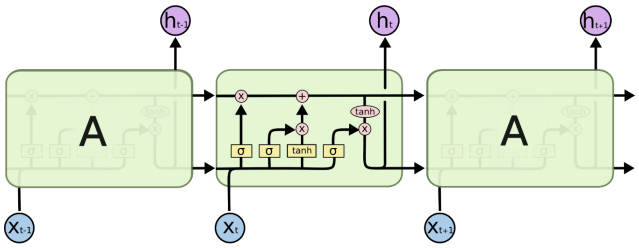

Figure 3: A Sample LSTM Cell Flow Chart. Image adapted from Olah's Blog.

The processed feature vectors are then fed into the sBART Decoder to generate English sequences in accordance with the feature vectors.

3.2.5 BART Decoder for Text Generation

For text generation, we leverage `facebook/bart-base` (BART) [9] [4], a powerful pre-trained large language model (LLM), to generate coherent and contextually appropriate text based on the encoded features from the video frames with the attention mechanism applied. The model employs a standard Transformer-based neural machine translation architecture, generalizing BERT (due to its bidirectional encoder) and GPT (with its left-to-right decoder), and is particularly effective for tasks such as dialogue, question answering, and summarization, often achieving state-of-the-art results in these areas [12] [14]. Additionally, this framework allows BART to excel in natural language generation and understanding tasks with the ability to handle the differences in the order and grammatical rules between glosses and spoken language [9]. Consequently, we utilized BART Decoder as a backbone in our architecture.

The decoder in BART operates autoregressively, meaning it generates one token at a time while considering the previously generated tokens. This left-to-right generation is crucial for tasks like text generation and translation, where the sequential order of tokens is significant. The 12 layers in the decoder includes:

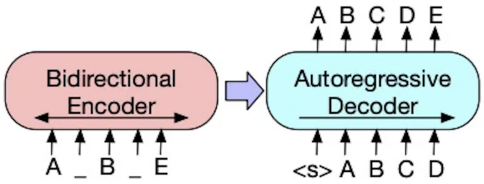

Figure 4: BART Decoder Architecture. With Encoder on the left and Decoder on the right.

- **Self-Attention Mechanism:** Restricted to only consider previous tokens to maintain causality. As mentioned in related works, attention is important to leverage in gloss-free models to create global understanding of the sign language video and attend semantically similar frames together.
- **Cross-Attention Mechanism:** This mechanism allows the decoder to attend to the encoder's outputs, effectively integrating the encoded input context with the generated sequence.
- **Feed-Forward Neural Networks:** These are used to transform the attended information before passing it to the next layer.
- **Layer Normalization and Residual Connections:** As in the encoder, these ensure stable and efficient training.

We integrate the high-quality encoded sign features from the Vision Transformer (ViT) encoder into the BART decoder with a linear layer, ensuring compatibility. The BART decoder is then employed to generate text conditionally based on these encoded features. With multiple layers, the model builds hierarchical representations of the input data, where lower layers capture basic syntactic information and higher layers model more abstract semantic relationships.

3.2.6 BART Tokenizer

The decision to use the BartTokenizer was made intentionally to leverage its advanced preprocessing capabilities, ensuring compatibility and efficiency in handling both the input sign language data and the generated text output. The BartTokenizer handles the tokenization of input sign language data, providing several key features that make it an ideal choice for our SLT task:

- **Bidirectional Processing:**
  The BartTokenizer supports bidirectional tokenization, crucially understanding the context in both forward and backward directions and aligns with our need to capture the details of sign language inputs, which may not always follow a linear structure.
- **Seq2Seq Batch Preparation:**
  The `prepare_seq2seq_batch` method of BartTokenizer efficiently prepare batches of source texts (the encoded sign language features) and labels (the expected translations), ensuring that both are appropriately tokenized and padded. This method supports padding as well as truncation strategies, ensuring that all sequences in a batch are of uniform length, which is essential for efficient batch processing in deep learning model.

3.2.7 Loss Calculation

We use cross-entropy loss to train the model, which is standard for sequence-to-sequence tasks. The loss is calculated as follows:

$$\mathcal{L} = -\sum_{t=1}^{N} y_t \log \hat{y}_t$$

where y_t is the true token and $\hat{y}_t$ is the predicted probability for the token at position t.

4. Dataset and Features

4.1. Dataset

We aggregate data from two datasets (sample video images shown in Figure 5):

WLASL Dataset: This dataset, presented at WACV 2020, is a collection of ASL videos representing 2000 English words. There are 11980 videos. Each video is processed to extract frames, which are then saved as .npy files. To facilitate efficient retrieval, we create a dictionary mapping each word to its corresponding video paths.

How2Sign Dataset: This dataset provides 4099 videos of complete ASL sentences and is a multimodal and multiview American Sign Language (ASL) dataset. It consists of more than 80 hours of parallel corpora of sign videos, including their respective speech, English transcripts, and depth information. The How2Sign dataset is divided into sentence segments, with frames extracted from each sentence and saved as .npy files.

4.2. Data Preprocessing

To preprocess our video dataset, we implemented a series of steps to ensure consistency and suitability for subsequent machine learning tasks. Initially, we defined a custom dataset class, `VideoDataset`, using PyTorch, which facilitated the handling of video data stored in .pt files and their associated labels from a CSV file. The dataset class was designed to dynamically load video frames and their corresponding labels upon request.

Figure 5: Images sampled from WLASL (top) and How2Sign (bottom) datasets.

4.2.1 Data Loading and Labeling

The `VideoDataset` class initializes by reading the CSV file containing the labels, structured with a video_id and a label for each video. The video frames are then accessed from the specified directory, ensuring each video's path is correctly constructed and the data is loaded efficiently. Our preprocessing pipeline ensures a structured and consistent format for video data, making it suitable for complex tasks such as video captioning and classification. By padding the frames and tokenizing captions, we prepared the dataset to be processed efficiently by the model, ensuring that each input in the batch maintains a uniform shape and structure. This preprocessing approach aims to enhance the model's performance and generalization capabilities by providing standardized input data.

4.2.2 Padding and Tokenization

Given the varying lengths of video frames, we employed a custom collate function, `collate_fn`, to process and collate the batches of data uniformly. The collate function performed the following steps:

- **Padding**: To standardize the number of frames across all videos in a batch, frames were padded to a fixed length (`max_frames`). This padding involved appending zero tensors to videos with fewer frames than `max_frames`. Conversely, videos exceeding `max_frames` were truncated.
- **Attention Masks**: Attention masks were created for the frames to distinguish between actual frames and padded frames. These masks contained ones for valid frames and zeros for padded frames, aiding the model in focusing only on relevant data.
- **Tokenization**: The captions associated with each video were tokenized using the BERT tokenizer. The tokenization process converted the text into fixed-length sequences (`max_len`), with padding and truncation applied as needed. This ensured all captions had uniform length, facilitating batch processing.

The final output of the preprocessing pipeline included padded frames, tokenized captions, and attention masks, all formatted for direct input into a neural network model.

4.2.3 Normalization

Data was normalized to ensure consistent input values across all videos. After preprocessing, the image dimensions were standardized to 224 x 224 pixels.

4.2.4 Dataset Splitting

Data was aggregated into 70% for training, 15% for validation, and 15% for testing. This split ensures a balanced distribution of data for model training and evaluation.

4.2.5 Time-Series Data Discretization

The time-series data in the videos is discretized by extracting frames at consistent intervals, allowing the model to process temporal information effectively. Each video input was discretized to 8 frames per second for 45 frames.

4.2.6 Examples from the Dataset

Examples from the dataset include images, video frames, and corresponding tokenized captions. These examples illustrate the data format and the preprocessing steps applied to prepare the data for model training.

4.2.7 Feature Extraction

In this setup, the features are extracted using the DinoV2 Vision Transformer (ViT) model and further processed using an LSTM for temporal aggregation.

5. Experiments and Results

5.1. Hyperparameters

The hyperparameters we experimented with are:

For the last three hyperparameters we found the best combination possible given computational limitations. Note, max frames is the most amount of frames allowed per

Parameter	Value
Learning Rate	1e-6
Optimizer	Adam
Batch Size	3
Max Frames	48
Max Sequence Length	50

Table 1: Hyperparameters

video, and max sequence length is the longest caption the model takes.

5.2. Metrics

We primarily use Cross-entropy Loss and BLEU to assess the performance of our models.

Using BLEU for our model is beneficial because it provides a standardized way to evaluate the quality of generated text by comparing it to reference texts. It measures n-gram precision, capturing how closely the model's output matches human-provided references, which is essential for assessing fluency and accuracy in text generation tasks.

5.3. Results and Interpretation

In this section, we present our results, highlighting the spoken language translations generated by our best-performing model given sign video representations (see Table 1). Note that Phoenix14T is a word-level dataset and hence models tend to perform better.

Models	Dataset	BLEU
NSLT (Camgoz et al. 2018)	Phoenix14T	9.00
TSPNet (Li et al., 2020)	Phoenix14T	13.41
CSGCR (Zhao et al., 2023)	Phoenix14T	15.18
SLTIV (Tarres et al., 2023)	How2Sign	8.03
Our Model (Mixed-data)	How2Sign	8.92
Our Model (Single-data)	How2Sign	**11.35**

Table 2: Model Performance Comparison(Gloss-free)

To interpret this performance, our best performing model and dataset setup gave a BLEU score of 11.35%. This, according to Google Cloud definitions, means the translation is "hard to get the gist", which suggests that the model could sometimes generate wrong choices of words or grammatical error that led to changes in meaning of the sentences. After analyzing the How2Sign dataset, we observed that there are sentences with nested and parallel grammatical structure such as the following

> So if you're doing a nap and you're the attackee, you know you're going to get hit and act like you got punched.

and sentences with low-frequency words such as

> The aileron is the control surface in the wing that is controlled by lateral movement right and left of the stick.

These complex structures and rare words likely contributed to the model's difficulty in generating accurate translations, highlighting areas for potential improvement in handling the ASL and its unique, specialized vocabulary.

5.4. Results of Aggregated-dataset Training

As mentioned in previous sections, our model was trained on two different data setups. Not only did we train the model on the How2Sign dataset, but also a mixture of the How2Sign dataset and the WLASL dataset. We wanted to see if the introduction of word-level videos can help the model learn to divide the input video into semantically meaningful segmants and hence improve understanding. As the testing suggests, mixing the datasets did not improve model performance (8.92 vs 11.35).

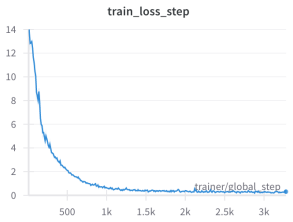

Figure 6: Training Loss

Figure 7: Validation Loss

Figure 8: Losses for Single-dataset training

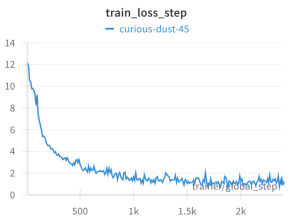

Figure 9: Training Loss

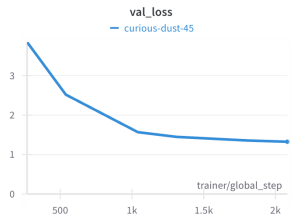

Figure 10: Validation Loss

Figure 11: Losses for Mixed-dataset training

We believe that the introduction of word-level videos increased the variance of the dataset, to which a sequence to sequence model is sensitive. Moreover, due to the dif-

ference in length between word-level videos and sentence-level videos, the encoder, constructed with Dino ViT and LSTM, might face challenges during training. As we can see, mixing the word-level video (WLASL) and sentence-level videos (How2Sign) introduces great variance during training, resulting in higher validation loss. The single dataset training has smoother curves and much lower validaton and testing loss.

5.5. Analysis on the Model

The DINO ViT model structure was greatly beneficial to extraction of visual features. To visualize the effects of the DINO encoder, we generated the following graphs to illustrate.

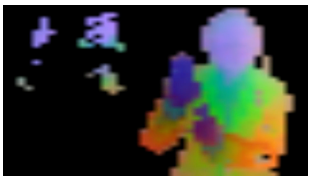

Figure 12: DINO ViT Features

Figure 13: Original Video

Figure 14: Effects of DINO ViT Encoder

As shown in Figure 14, the DINO model is very effective in terms of semantic segmentation and depth recognition. However, dividing the input image into multiple small patches to feed into Transformer, which is an essential design of DINO, causes the model to lose small details such as the fingers and the facial expressions, while having robust understanding of the main object of the image. We presume that this could be improved by having higher image definition and smaller patch sizes. Unfortunately, this is not possible for most advanced research teams due to computational resource limitation.

6. Conclusion / Future Work

In this paper, we have presented a novel approach to address the challenging problem of Sign Translation in a gloss-free setting. Our method demonstrates performance improvements over existing techniques on the How2Sign dataset. We introduced a strategy that learns from both word-level sign features and sentence level sign features, thereby allowing our sign encoder to be effectively pretrained without the use of manually annotated glosses. We investigated that the improvement this brings is limited and future work is needed.

Our analysis indicates that while our model performs well on standard datasets, it still faces challenges with complex grammatical structures and low-frequency words.

For future work, we aim to explore more sophisticated preprocessing techniques and advanced model architectures to better capture the nuances of sign language. Additionally, increasing the training data with more diverse and comprehensive sign language datasets could help improve the model's generalization capabilities. Further research will also focus on refining the temporal aggregation of visual features to enhance the performance of the sign encoder, ensuring a more robust and accurate translation system.

7. Contributions and Acknowledgements

8. Appendices

8.1. Python Packages Used in Implementation

List of Python packages used in the implementation:

Library	Reference
numpy	[2]
pandas	[18]
scikit-learn	[8]
ffmpeg-python	[22]
tqdm	[4]
torch	[16]
torchvision	[17]
opencv-python	[3]
pytorch-lightning	[11]

Table 3: List of Python libraries used and their references

References

[1] N. C. Camgoz, O. Koller, S. Hadfield, and R. Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10023–10033, 2020. 1, 2

[2] N. Developers. Numpy. Available: https://numpy.org/. 8

[3] O. Developers. Opencv-python. Available: https://opencv.org/. 8

[4] T. Developers. Tqdm. Available: https://github.com/tqdm/tqdm. 8

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[6] A. Duarte, S. Palaskar, L. Ventura, D. Ghadiyaram, K. DeHaan, F. Metze, J. Torres, and X. Giro-i Nieto. How2Sign: A Large-scale Multimodal Dataset for Continuous American Sign Language. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[7] J. Forster, C. A. Schmidt, T. Hoyoux, O. Koller, U. Zelle, J. H. Piater, and H. Ney. Rwth-phoenix-weather: A large vocabulary sign language recognition and translation corpus. In *International Conference on Language Resources and Evaluation*, 2012. 2

[8] S. learn Developers. Scikit-learn. Available: https://scikit-learn.org/. 8

[9] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. 2019. 2, 4

[10] D. Li, C. Rodriguez, X. Yu, and H. Li. 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*, pages 1459–1469, 2020. 2

[11] P. lightning Developers. Pytorch-lightning. Available: https://www.pytorchlightning.ai/. 8

[12] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020. 4

[13] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics. 2

[14] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020. 4

[15] L. Tarrés, G. I. Gállego, A. Duarte, J. Torres, and X. Giró-i Nieto. Sign language translation from instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 5625–5635, June 2023. 1, 2

[16] P. Team. Torch. Available: https://pytorch.org/. 8

[17] P. Team. Torchvision. Available: https://pytorch.org/vision/stable/. 8

[18] P. D. Team. Pandas. Available: https://pandas.pydata.org/. 8

[19] R. Wong, N. C. Camgoz, and R. Bowden. Sign2gpt: Leveraging large language models for gloss-free sign language translation. *arXiv preprint arXiv:2405.04164*, 2024. 1, 2, 4

[20] J. Ye, X. Wang, W. Jiao, J. Liang, and H. Xiong. Improving gloss-free sign language translation by reducing representation density. *arXiv preprint arXiv:2405.14312*, 2024. 2

[21] A. Yin, T. Zhong, L. Tang, W. Jin, T. Jin, and Z. Zhao. Gloss attention for gloss-free sign language translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2551–2562, 2023. 1, 2

[22] Z. Zhang. ffmpeg-python. Available: https://github.com/kkroening/ffmpeg-python. 8