Deep Learning Methods for Sign Language Translation

TEJASWINI ANANTHANARAYANA, PRIYANSHU SRIVASTAVA, AKASH CHINTHA, AKHIL SANTHA, BRIAN LANDY, JOSEPH PANARO, ANDRE WEBSTER, NIKUNJ KOTECHA, SHAGAN SAH, THOMASTINE SARCHET, RAYMOND PTUCHA, and IFEOMA NWOGU, Rochester Institute of Technology

Many sign languages are bona fide natural languages with grammatical rules and lexicons hence can benefit from machine translation methods. Similarly, since sign language is a visual-spatial language, it can also benefit from computer vision methods for encoding it. With the advent of deep learning methods in recent years, significant advances have been made in natural language processing (specifically neural machine translation) and in computer vision methods (specifically image and video captioning). Researchers have therefore begun expanding these learning methods to sign language understanding. Sign language interpretation is especially challenging, because it involves a continuous visual-spatial modality where meaning is often derived based on context.

The focus of this article, therefore, is to examine various deep learning–based methods for encoding sign language as inputs, and to analyze the efficacy of several machine translation methods, over three different sign language datasets. The goal is to determine which combinations are sufficiently robust for sign language translation *without* any gloss-based information.

To understand the role of the different input features, we perform ablation studies over the model architectures (input features + neural translation models) for improved continuous sign language translation. These input features include body and finger joints, facial points, as well as vector representations/embeddings from convolutional neural networks. The machine translation models explored include several baseline sequence-to-sequence approaches, more complex and challenging networks using attention, reinforcement learning, and the transformer model. We implement the translation methods over multiple sign languages—German (GSL), American (ASL), and Chinese sign languages (CSL). From our analysis, the transformer model combined with input embeddings from ResNet50 or pose-based landmark features outperformed all the other sequence-to-sequence models by achieving higher BLEU2-BLEU4 scores when applied to the controlled and constrained GSL benchmark dataset. These combinations also showed significant promise on the other less controlled ASL and CSL datasets.

CCS Concepts: • **Human-centered computing** → *Gestural input*; • **Computing methodologies** → **Language resources**; *Image representations;*

Additional Key Words and Phrases: Sign language translation, Deaf and Hard-of-Hearing, accessibility, deep learning, sequence modeling, transformer, attention

This material is based upon work partially supported by the National Science Foundation under Grant No. 1846076. We thank Divyansh Gupta for his assistance in providing some of the statistical information for the manuscript.

Authors' address: T. Ananthanarayana, P. Srivastava, A. Chintha, A. Santha, B. Landy, J. Panaro, A. Webster, N. Kotecha, S. Sah, T. Sarchet, R. Ptucha, and I. Nwogu, Rochester Institute of technology, Rochester, New York, 14623; emails: {ta2184, ps6808, ac1864, as2958, bxl1703, jgp6466, agw1754, nrk1787, sxs4337, tasbka, rwpeec, ionvcs}@rit.edu.

1936-7228/2021/10-ART22 $15.00
https://doi.org/10.1145/3477498

ACM Reference format:
Tejaswini Ananthanarayana, Priyanshu Srivastava, Akash Chintha, Akhil Santha, Brian Landy, Joseph Panaro, Andre Webster, Nikunj Kotecha, Shagan Sah, Thomastine Sarchet, Raymond Ptucha, and Ifeoma Nwogu. 2021. Deep Learning Methods for Sign Language Translation. *ACM Trans. Access. Comput.* 14, 4, Article 22 (October 2021), 30 pages.
https://doi.org/10.1145/3477498

1 INTRODUCTION

According to the World Health Organization, approximately 430 million people have hearing loss and require some rehabilitation assistance [66]. Sign language is a natural visual-spatial language used for communication among many **Deaf and Hard-of-Hearing (DHH)** people. It is estimated that there are over 200 [34] different sign languages used around the world and sign languages are not necessarily mutually intelligible even if their spoken counterparts are similar [69].

Unfortunately, the majority of hearing people do not understand sign language, and this may result in missed opportunities for some members of the DHH communities, in areas such as education, employment, sports, and other social activities. Sign language interpretation can address many of these challenges by facilitating the communication between signers and non-signers. But the use of human interpreters for sign language translation can be inconvenient, difficult to schedule, and costly, hence more research groups are attempting to automate this translation process using machine learning techniques. It is important to note that an artificial intelligence tool such as we propose facilitates the language system support required by both DHH signers and non-signers, but is not a replacement for sign language interpreters or direct signer to signer communication.

Many sign languages are bona fide languages with their own grammars and lexicons [81], and hence there are rules governing the order of signed words along with the forms of signed words to use when communicating. Because of these underlying rules, we can implement machine translation methods for sign languages. In recent times, text-to-text–based machine translations have enjoyed much success using deep learning–based sequence modeling methods, and in this work, we visit how these methods are applied to sign language translation.

Unlike regular hand gestures, sign languages typically have five parameters (grammatical features): handshape, location (use of space), palm orientation, body movement, and facial grammar (non-manual signals) [36, 71]. Several of these five parameters such as leaning the body forward/backward, head turns and shakes, eyebrow movements, nose wrinkling, mouth movements, and so on, do not have equivalents in written or spoken language grammar. For this reason, unlike their spoken/written language counterparts, for machine analysis, sign languages need to be encoded visually and spatially. Deep learning–based computer vision techniques, provide a framework to support this.

In this article, therefore, we evaluate various deep learning–based methods for *encoding* sign language at the front-end and combine these with various deep learning–based *machine translation* techniques at the back-end. Specifically, for machine translation, we implement sequence-to-sequence models with and without attention, a **reinforcement learning– (RL)** based model, and a transformer model for sign language to text translations. We perform this comprehensive evaluation using three different established sign languages. To the best of our knowledge, this is the first work to perform continuous sign language translation, *without* the intermediary gloss[1]-based recognition step.

[1]More about gloss is discussed in Section 2.

The rest of the article is organized as follows: a brief explanation of glossing for the **American Sign Language (ASL)** is presented in Section 2; the related works are described in Section 3; Section 4 describes the sign language translation methodologies implemented in this work; the dataset statistics and feature vectors used are described in Sections 5 and 6, respectively; popular evaluation metrics are described in Section 7.1; Section 7.2 discusses training details and results obtained from different ablations and a brief analysis of the results are given in Section 7.3; and, finally, the conclusion is presented in Section 8.

2 SIGN LANGUAGE TRANSCRIPTION (GLOSSING)

Glossing is a written form of sign language [37, 38], though it does not have the same structure as the spoken language equivalent. When a person signs a phrase, glossing refers to the written form of the sign-for-sign transcription (not translation) of that phrase. For example, when signing the phrase "My name is Albert," the signer might finger-spell "Albert" and conceptually place that spelled construct at some location in a three-dimensional (3D) space in the vicinity of the signer. This way, any ensuing references to the name "Albert" will no longer require full fingerspelling; rather, the signer can point to that abstract location in space where the construct was stored. While the gloss can accurately capture this process, there is no equivalent representation in the spoken/written equivalent language. Sign language translation aims more to preserve the overall meaning of the phrase while following the rules of the language grammar. As another example, ASL makes use of special signs known as *classifiers*. A classifier categorizes a bunch of phrases to a single sign that then can be used as a reference while signing. An example is the "3-handshape" used to represent a vehicle. But the sign can also include the orientation, speed, location, and direction of travel of the vehicle. The written form or gloss of a sign includes information about these classifiers and also includes information relating to facial and body movements.

Some glossing terminologies include information like `LOC` referring to a location, `PO` referring to palm orientation, `LCL` referring to locative classifiers, and so on. Glossing also includes information based on the number of times a sign is repeated. For example, the plus sign `++` at the end of a gloss indicates a number of repetitions of an ASL word; e.g., `again++` (signing the word "again" two more times) means "again" and "again." Another example, `HELP+++` could mean "help many times" or "help from time to time," depending on the context, the duration of the movement, and spatial reference to convey different meanings.

Some glosses along with their meanings and spoken sentence equivalent are shown in Table 1.

The main difference between a sign and a spoken language is the mode of communication, where sign languages are visual-gestural, whereas spoken languages are based on speaking and hearing [19]. For this reason, it is difficult to capture sign language fully in writing and is best done via a video recording. For continuous sign language translation, researchers are presented with the challenge of interpreting a language based on a complex visual-gestural grammar involving handshapes, spatial locations, facial expressions, palm orientations, movement, and non-manual signals from video recordings into the written word. This brief overview of glossing sheds some light on the significant challenges involved in interpreting signs to spoken language.

In this work, we address some of these sign language translation challenges by (i) exploring different visual features that can be extracted from signing videos, (ii) performing neural machine translation using various sequence-to-sequence deep learning models ranging in complexity, and (iii) presenting our findings from applying the features and models to three different sign languages. In the real-world scenario, because it is challenging to obtain gloss information, we address sign language translation in this article, by investigating the best model and input features that are robust enough without using any gloss-based information.

Table 1. Glosses (Sign-for-Sign Transcriptions) and Their Spoken Sentence Translations from the ASL Dataset [21]

No.	Gloss	Spoken Sentence	Selected gloss meaning
1.	IX-loc:j OVER/AFTER EXAGGERATE IX-3P:j LATER_2 USE COAT RAIN COAT USE UMBRELLA BOOT PANTS_3	People go too far: they use umbrellas, wear rain coats, and put boots and pants on.	IX: index, IX-loc: points to a location.
2.	IX-1p SAY part:indef UP-TO-NOW IX-3p:I 5"looking for words" IX-3p:I fs-PAUL KNOW IX-3p:I POSS-1p (1h) GOOD/ THANK-YOU FRIEND	I said to Paul in the email, "you know, you have been a good friend of mine"	fs: fingerspell, POSS-1: Possessive pronouns (my, mine, etc,).
3.	IX-1p LOOK:p 5"resignation" FINE IX-1p WAIT++	Thought "ugh" and ended up waiting again.	++: repeat sign.

3 RELATED WORKS

Sign language translation borrows concepts from various **natural language processing (NLP)** tasks where sentences from one language are translated to the other. In the following subsections, we discuss some of these methods and highlight existing research being done that contribute in one way or the other toward sign language interpretation. In the following subsections, we discuss different sequence modeling techniques related to sign language understanding including text-to-text translation, video captioning, gesture recognition, reinforcement learning, and video sign language to text translation.

3.1 Sequence Modeling for Text-to-Text Translation

One application of sequence modeling is machine or language translation. Neural machine translation is the task of translating a text from one language to the other. Machine translation tasks started to gain recognition from the work proposed by Kalchbrenner and Blunsom [40] where a probabilistic continuous translation model was introduced. Following this work, Sutskever et al. [89] introduced a multilayered **Long Short-Term Memory (LSTM)** [33] to take in a variable number of encoded words of the input sentence and translate it to a target language using another LSTM in the decoder. The performance of these networks was limited by the length of the input sequence that was highlighted by Cho et al. [12]. Bahdanau et al. [5] extended LSTMs by introducing attention mechanism. This involves taking a linear combination of encoder hidden states when predicting the next target word in the decoder. There are many other works using attention mechanism for the neural machine learning tasks [55, 105].

Vaswani et al. [94] introduced a text-to-text–based model completely based on attention and feed-forward networks without the use of **Recurrent Neural Networks (RNNs)**. These models, called transformer models, process all input words in parallel using the query, key, and value word embeddings. Other works such as Generative Pre-Traning models, GPT-1 [76], GPT-2 [77], and GPT-3 [7] have extended the work by Vaswani et al. These models are designed for text-based tasks like text classification, sentence similarity, question answering, next-word prediction, and text summarization. These GPT models initially train the transformer model with large unlabeled data in an

unsupervised fashion and later perform supervised learning to address the above-mentioned tasks. GPT-3 [7] is a very big language model with 175 billion parameters. **Bidirectional Encoder Representations from Transformers (BERT)** [24] model, also inspired by the transformer model, trains on unlabeled data in an unsupervised manner and these pre-trained weights are used to fine-tune on labeled data for downstream tasks like question and answering.

3.2 Video Captioning

Sequence modeling has also been extended toward other variable length tasks like video description, visual question and answering, scene description, and action or gesture recognition. Venugopalan et al. [97] introduced a sequence-to-sequence video to text model with two stacked LSTMs. Each frame is passed one at a time during the encoding stage. The hidden representation from the encoding stage is passed onto the decoding stage where the model predicts one word at a time. Hierarchical Recurrent Neural Networks proposed by Yu et al. [108] not only include attention mechanisms but also use sentence and paragraph generators. Researchers have explored different variations of LSTMs such as leveraging attention mechanism with semantic consistency [28, 67], and LSTMs with temporal modifications based on the discontinuities between the frames [6]. Olivastri [63] perform end-to-end captioning of videos with an encoder-decoder architecture using LSTMs with attention. Aafaq et al. [3] performed comprehensive ablation studies for video captioning across several datasets and compare benchmark results based on the size of the dataset and number of classes. They evaluate results on different evaluation metrics to highlight the advantages and disadvantages of different metrics. Reinforcement learning-based video captioning is also gaining popularity with a teacher–student module [50, 101]. To enable processing of longer sequences, transformer-based models [11, 87, 113] and BERT-based models [54, 88] are also contributing toward improving the performance of the video captioning tasks. Vision-and-Language BERT [54] introduced co-attentional transformer layers over two parallel pipelines, one for text and the other for visual data. The co-attentional layer helps in providing information from the video pipeline to text pipeline and vice versa. Transformer-XL [18] enhanced the NLP tasks by learning even longer-term dependencies in a computationally contractible fashion.

3.3 Gesture Recognition

Gesture and action recognition is a variant of sequence modeling. These are often treated as a categorical classification problem. In these types of models, the temporal features of the video and specifically the motion of the body, hand, and face play a major role. Karpathy et al. [41] introduced several early, late, and mid-fusion temporal variations of **convolutional neural networks (CNN)** for video classification. Pigou et al. [73] and Nishida et al. [61] used recurrent architectures for gesture recognition. To facilitate the gesture recognition process, several works have studied ways to extract 2D and 3D locations of body, hands, and face. OpenPose [9, 10, 86, 102] consolidated many of these works into a single framework by providing 70 facial keypoints, 25 body keypoints, and 21 hand keypoints for each hand. OpenFace [65] extracts facial landmark and action units. Artrack [39] also provides key joints in the body. While gesture and action recognition analyze a sequence of frames for discrete classification, continuous sign language translation analyzes a stream of frames, continuously outputting textual content that takes into consideration prior video context.

3.4 Reinforcement Learning

Reinforcement learning has been very popular in robotics research [43, 47, 70, 95]. Reinforcement learning gained a lot of fame from its successes in playing the well-known Atari game [58] and AlphaGo game [27, 85]. However, only recently has it started to contribute to the field of sequence

modeling. A typical sequence-to-sequence model like LSTM is trained such that it predicts the next word based on the previous ground-truth word. This method is also sometimes referred to as *"teacher forcing"* where instead of passing the previous predicted word to obtain the next word, the previous ground-truth word is passed. *Teacher forcing* is applied only during training. During test time the predicted previous word is passed on to guess the next word. This leads to a performance hit, known as *exposure bias*, as the model is not able to generalize well. *Exposure bias* ([78, 111]) leads to error during test time. In addition to exposure bias, there can be issues because there is no back-propagation of the evaluation metric during testing, i.e., we can see *non-differentiable* issues during testing. These two issues are addressed largely by using RL.

Rennie et al. [80] implemented image captioning using **self-critical sequence training (SCST)**, which is based on the *REINFORCE* algorithm [103]. SCST with REINFORCE baselines its experiences from the test-time inference algorithm output to normalize the rewards. A positive weight is assigned to the samples that performed well during test-time inference and those that do not perform well are suppressed. In this work, SCST with REINFORCE was performed after the system was trained with cross-entropy loss for a minimum of 20 epochs. To improve the test-time inference, the **Consensus-based Image Description Evaluation (CIDEr)** [96] metric is used with SCST along with greedy decoding. CIDEr score is evaluated by matching predicted sentence with the consensus of a set of human-annotated ground-truth sentences. Shi et al. [84] use two networks, a "policy network" and "value network," which perform joint learning to predict the next word in an image captioning task. While the policy network evaluates the confidence of predicting the next word, the value network evaluates rewards based on the predictions. The policy network, in short, is a supervised network trained with cross-entropy loss. The value network is trained to minimize the mean squared loss. These two networks are then jointly trained to maximize the rewards. Li et al. [51] and Zhang et al. [111] extend the concept from Reference [80] by using the REINFORCE algorithm toward video captioning and continuous sign language recognition tasks, respectively. While Reference [51] uses a sequence-to-sequence LSTM-based network with the CIDEr score, Reference [111] uses the transformer model and **Word Error Rate (WER)** to minimize the loss and maximize the reward.

3.5 Video Sign Language Translation

Word-level and sentence-level datasets are widely used for sign language translation. Word-level sign language translation tasks can be classified under image classification or gesture/action recognition. References [45, 46, 52, 59] made contributions toward tasks like translating discrete signs, mouth, and hand shapes while Ye et al. [106] used a combination of RGB, optical flow, and depth information for ASL translation using a 3D CNN to classify words using its respective temporal information from the continuous sign language video. Residual architectures with temporal convolutions and framewise classification were used by Pigou et al. [74] to classify 100 signs. Word-level sign language interpretation is a simpler task than continuous sign language as the former involves classifying the input clip between a given number of classes. Whereas for continuous sign language, the input clip itself is long enough needing to take care of long-term dependencies and the output is a series of words where every current word depends on the previously predicted words.

Continuous sign language utilizes temporal information and can be considered a sub-set of video captioning. Fang et al. [26] introduced DeepASL, which uses a hierarchical bidirectional deep recurrent neural network for both word-level and continuous sign language translation. DeepASL is an extensive work showcasing sign language translation and speech recognition for real-time two-way communication. Wang et al. [100] introduced a hybrid model where C3D-ResNet features from the input frames are passed into a **temporal convolution (TCOV)** and **bidirectional GRU**

(BGRU) block. TCOV and BGRU are responsible for short-term and long-term learning, respectively. A fusion layer is also used with inputs from both the TCOV and BGRU blocks to learn the complementary relationship between them. **Connectionist temporal classification (CTC)** [30] loss combines losses from these three blocks to predict the words of the sign language sentence. Pu et al. [75] used 3D temporal residual CNNs with CTC post-processing for continuous sign language recognition, whereas Reference [16] use CTC loss to train end-to-end models with custom networks targeting certain specific problems known as SubUNets. Camgoz et al. [17] used a multilayer RNN and an encoder-decoder structure for sign language to text translation. This model incorporates encoder-decoder attention to improve the learning process. Yuan et al. [110] introduced a **Chinese sign language dataset (CSLD)** and performed Chinese sign language translation by modeling a two-layer LSTM encoder-decoder–based architecture using different body, hand, and facial features from input frames. Ko et al. [42] studied different methods for sign language translation using various joint locations and CNN features as input on the Korean sign language dataset. In this article, we study sign language translation by not only considering different models but by also looking at different datasets to understand how the organization of the dataset affects the performance.

4 SPECIFIC MODELS FOR SIGN LANGUAGE TRANSLATION

In this section, we review different deep learning captioning methods specifically tailored to perform sign language to text translation. We start by briefly explaining a basic Recurrent Neural Network and then sequence-to-sequence models, transformer models, and inclusion of RL.

4.1 Recurrent Neural Networks

Neural networks are well suited for handling tasks that have fixed size input and output vectors. For example, a CNN takes as input an image of predetermined resolution and outputs a vector of probabilities for each output class it has been trained on. For temporal tasks, such as videos, this concept can be extended to take in a fixed number of frames. RNN extend this concept further by using hidden states. Hidden states allow the output to be a function of the current input and the output from the previous timestep. By passing the previous hidden state information, the RNN can make predictions based upon all prior inputs. RNNs are powerful architectures for NLP-based tasks like language translation, question and answering, image captioning, and video captioning. Vanilla or basic RNNs can take in an arbitrary number of timesteps, but unfortunately, the passing of information over time is subject to vanishing gradients, thus limiting its usefulness to only a few timesteps. To remember longer-term dependencies of dozens to even over 100 timesteps, a special type of RNN called LSTM [33] can be used. Using a series of gates and a second hidden state called a memory cell, LSTMs can be used as a replacement for the vanilla RNN. **Gated recurrent unit (GRU)** [13, 14] is a slight variation of LSTM with fewer gates and no memory cells. The LSTM and GRU RNN methodologies have been used extensively in deep learning applications like image captioning, video captioning, speech recognition, and language translation. Even LSTMs and GRUs are, however, subject to vanishing and exploding gradient problems on long sequences. Transformer model and RL address these issues and are described in Sections 4.4 and 4.5, respectively.

4.2 Sequence-to-Sequence Modeling for Sign Language to Text

A basic sequence-to-sequence model for sign language video to text translation inspired by References [89, 97, 98] is described in this subsection. The sign language translation task is modeled to take the features derived from the input frames such as OpenPose [9], extracted features from a Deep Neural Network, or k-means centroids [53]. These features are fed one frame at a time to the encoder. The hidden state from the encoder carries semantically rich features of the signing

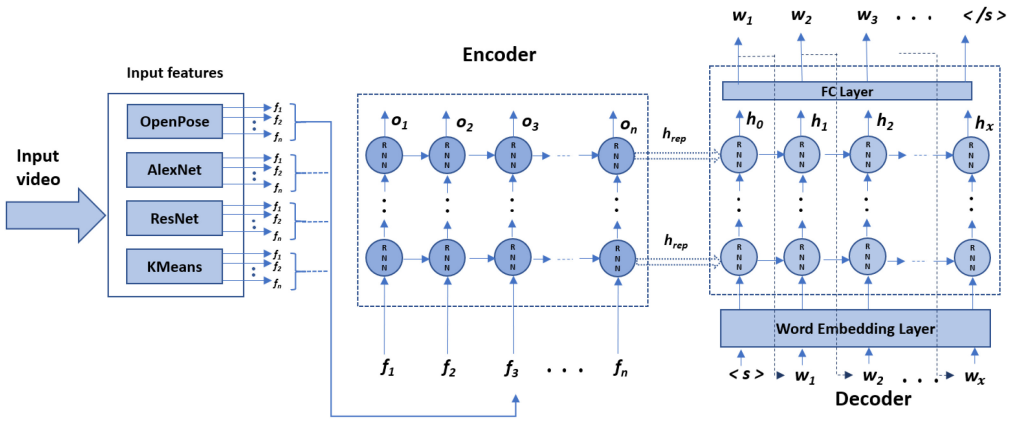

Fig. 1. Sequence-to-sequence model without attention.

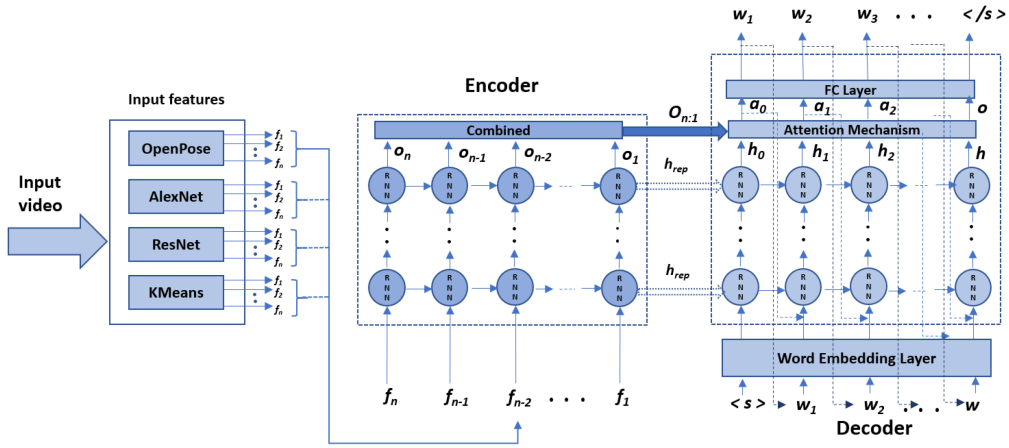

Fig. 2. Sequence-to-sequence model with attention mechanism.

in the visual domain. The hidden states are propagated to the decoder that employs those features in producing one word at a time. During inference time the decoder starts predicting words upon receiving the "start of sentence" token <s> and predicts one word at a time. The word predicted at timestep t is fed as input to the next layer at timestep $t+1$. The encoder and the decoder layers are typically stacked with LSTM units. As this serves as a baseline model for the sign language translation task, experiments using single and multiple stacked recurrent layers on both the recurrent channels have been carried out. The sequence-to-sequence model is shown in Figure 1 with respective notations in Table 2.

4.3 Sequence-to-Sequence Modeling with Attention

Inspired by sequence-to-sequence models [17, 97] and attention mechanisms [5, 55], an architecture is assembled to perform sign language to text translation as shown in the Figure 2. Inputs to the model are identical as to the basic sequence-to-sequence model shown in Figure 1. One

Table 2. Notations for Sequence-to-Sequence Models

Notation	Meaning
n	Input sequence length
x	Output sequence length
$f_1, f_2, f_3 \ldots . f_n$	Feature vectors for each frame of the video
$o_1, o_2, o_3 \ldots . o_n$	Output vectors from encoder for each frame of the video
h_{rep}	Latent embedding from encoder
$w_1, w_2, w_3 \ldots . w_x$	Predicted words
$a_1, a_2, a_3 \ldots . a_x$	Encoder-decoder attention vectors for each word
$< s >$	Start of sentence token
$< /s >$	End of sentence token

frame feature at a time is passed into the encoder. Reversing the sequence order has proven to be beneficial to the model [17, 89], as it helps mitigate long-term dependency and vanishing gradients. With respect to Figure 2, $o_{n:1}$ a linear combination of the output of all input timesteps and is passed onto the decoder in addition to the latent embedding h_{rep}. The decoder takes in previous word embedding, previous hidden state, and previous attention weights as input. The decoder starts decoding words after receiving the "start of sentence" token ($< s >$) and predicts one word at a time by feeding the previous predicted word as input to the next timestep as explained in Section 4.2. The attention mechanism block incorporates encoder-decoder attention by taking in the output vector from the encoder and previous hidden representation from the output of the decoder. The attention vector is calculated as shown in Equation (1),

$$a_x = tanh(W_c c_x + W_h h_x + W_b b), \tag{1}$$

where c_x is the context vector, h_x is the hidden state, b is the bias, and W_c, W_h, and W_b are the learned weights for the context vector, hidden state, and bias, respectively. The context vector as shown in (2) is a weighted sum of encoder outputs. γ_n^x represents the attention weights,

$$c_x = \sum_{n=1}^{N} \gamma_n^x o_n. \tag{2}$$

The attention weights highlight the importance of encoder input with the generated word while predicting the next word. The attention weights are typically normalized using the probabilistic softmax equation. Attention weights and mechanism are shown in Equation (3),

$$\gamma_n^x = softmax(f_{att}(h_x, o_n)), \tag{3}$$

where f_{att} represents the attention mechanism used. We give two options for function f_{att}, (Equation (4a)) Luong attention [55] based on multiplication and (Equation (4b)) Bahdanau attention [5] based on concatenation,

$$X = h_x^T W o_n, \tag{4a}$$

$$X = V^T tanh(W[h_x; o_n]). \tag{4b}$$

Here, W and V are learned weights. Combining (1:4), the decoder can be expressed as Equation (5) as follows:

$$y_x, h_x = Decoder(word\_embedding_{x-1}, h_{x-1}, a_{x-1}). \tag{5}$$

The decoder stops decoding once it receives the "end of sentence" token ($< /s >$).

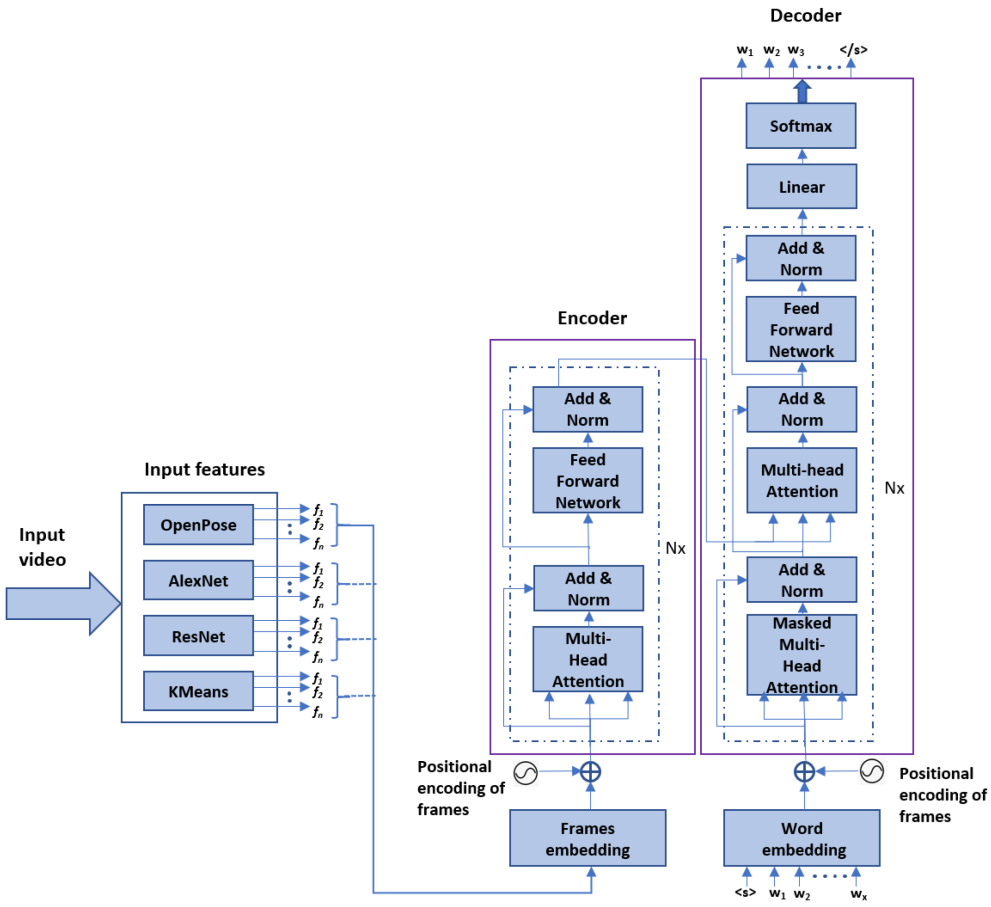

Fig. 3. The baseline transformer model.

We implement sequence-to-sequence model using LSTM as the RNN by sending in a reversed input frame sequence and incorporating encoder-decoder attention based on Luong attention [55] to predict the words one at a time in the decoder.

4.4 Transformer Model

Transformer models have demonstrated remarkable capabilities lately for language translation tasks [24, 76, 77, 94]. This idea was leveraged for video captioning [11, 18, 113]. Recently, Camgoz et al. [8] modified the transformer model for sign language translation using CTC loss. The features used in this model were trained on a CNN-LSTM-HMM architecture [44] where gloss labels are used in a weakly supervised setup followed by a HMM model that performs the alignment. Yin [107] experimented with the transformer model by using different number of layers in the transformer model and perform Sign to Gloss and Gloss to Text translation. We focus on the transformer model shown in Figure 3, which is a state-of-the-art model introduced by Vaswani et al. [94] for the task of sign language to text translation. While an LSTM has been shown to produce good results with over a 100 output tokens long, transformers can produce good results with over a 1,000 output tokens long. The transformer encoder and decoder layers are repeated N times.

Each layer in an encoder consists of two sublayers and the decoder consists of three sublayers. Unlike a typical RNN where sequence of inputs are fed one at a time, the transformer takes all the inputs together. To help the model understand the order of input, a positional encoding parameter is introduced. The positional encoding provides the order information in the sequence of inputs and this is added to the input embedding before passing it to the encoder. Positional encoding is given in Equation (6) [94],

$$\begin{aligned} PE_{(pos,2i)} &= sin(pos/10000^{2i/d_{model}}) \\ PE_{(pos,2i+1)} &= cos(pos/10000^{2i/d_{model}}), \end{aligned} \tag{6}$$

where *pos* is the position of the input in the sequence of inputs and i represents embedding dimension. The first sub-layer in the encoder is the "Multi-head Attention." The customized positional encoding with input embedding is now used to calculate the keys (K), queries (Q), and values (V) by learning three distinct linear layers as shown in Equation (7),

$$\begin{aligned} K &= linear\_key(input) \\ Q &= linear\_query(input) \\ V &= linear\_value(input). \end{aligned} \tag{7}$$

In Equation (7), input is a combination of input embedding and positional encoding. *linear_key*, *linear_query*, and *linear_value* are three distinct linear neural network layers for K, Q, and V, respectively. These three linear layers can also be considered as a typical **feedforward neural network (FFN)** layers that are learned separately with three different weights. The attention is then calculated as shown in Equation (8) [94],

$$Attention(K, Q, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{8}$$

where d_k is the dimension of the key vector. Equation (8) is scaled by a factor of $\sqrt{d_k}$ to avoid the larger values of dot product that will leawd to smaller gradients due to softmax. This attention mechanism is called *Scaled Dot-Product Attention*. The authors of Reference [94] introduced the concept of *Multi-Head Attention* where the linearly learnt queries, keys, and values are projected separately h times where h represents the number of heads. For each of these projections, scaled dot-product attention vectors are calculated as per Equation (8), and the results for all the heads are concatenated together in a final linear layer to obtain the multi-head attention layer output. This multi-head attention output is passed through a FFN with two linear layers and a ReLU activation as shown in Equation (9), where x is the output from multi-head attention block,

$$FFN(x) = max(0, xW1 + b1)W2 + b2, \tag{9}$$

where $W1$ and $b1$ are the weights and biases, respectively, of the first linear layer; $W2$ and biases are weights and biases of the second linear layer.

The first sub-layer in the decoder is a masked multi-head attention layer. The masking mechanism is used to mask the future words. The second sub-layer in the decoder is the encoder-decoder multi-head attention where the query comes from the previous decoder layer and key and value parameters come from the encoder layer. The output is then passed through the FFN as per Equation (9) followed by a linear layer and softmax to obtain the output probabilities. Residual connections are used across all sub-layers in the encoder and decoder. After every sub-layer, a layer normalization is implemented where the inputs are normalized across the features.

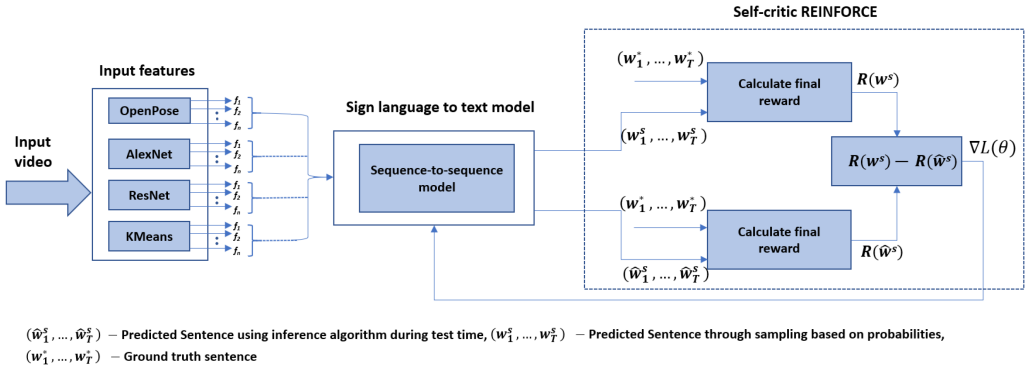

Fig. 4. Sign language to text model using reinforcement learning.

4.5 Sequence-to-Sequence Modeling with Reinforcement Learning

Inspired by References [80, 111] we investigate the performance of a sign language to text model using RL to update the weights and contrast and compare its behavior with different input feature vectors and different models. Reinforcement learning learns what actions need to be taken to maximize a certain reward. Actions are not initially defined but it is something that the model discovers while obtaining either positive or negative rewards [90]. Most of the sequence-to-sequence methods use "teacher-forcing" while training leading to exposure bias issues during test time. Reinforcement learning mitigates this exposure bias problem. Casting our model in the RL archetype, the type of model used (sequence-to-sequence, or transformer) will act as an agent with a policy p_θ. The environment will be a video from a sign language dataset, and the state or observation will be frames from the video. The action for this task would be predicting the next word. Initially, the reward is set to 0 until *EOS* token is received. At the end of all the tokens, the final reward is denoted as R. The goal is to minimize the negative expected reward as shown in Equation (10). The architecture is as shown in Figure 4,

$$L(\theta) = -\mathbb{E}_{w^s \sim p_\theta}[R(w^s)], \tag{10}$$

where $w^s = (w_1^s, \ldots, w_T^s)$ are the words sampled from the model from timesteps $1 - T$. Adapting from References [90, 103], we utilize REINFORCE [103] with a baseline that is computed as shown in Equation (11),

$$\nabla L(\theta) = -\mathbb{E}_{w^s \sim p_\theta}[R(w^s)\nabla_\theta log p_\theta(w^s)]. \tag{11}$$

The self-critic REINFORCE algorithm is calculated by using inference algorithm at test time for calculating the reward,

$$\begin{aligned} R(w^s) &= Metric((w_1^s, \ldots, w_T^s), (w_1^*, \ldots, w_T^*)) \\ R(\hat{w}^s) &= Metric((\hat{w}_1^s), \ldots, (\hat{w}_T^s), (w_1^*, \ldots, w_T^*)), \end{aligned} \tag{12}$$

where $(w_1^s, \ldots, w_T^s)$ is the sentence generated through sampling-based probabilities and $((\hat{w}_1^s), \ldots, (\hat{w}_T^s))$ is the sentence generated using the inference algorithm during test time and $(w_1^*, \ldots, w_T^*)$ is the ground-truth sentence. The *Metric* based on which the reward in Equation (12) is calculated can be any evaluation metric like CIDEr or WER.

The word sampled at timestep t, $\hat{w}_t$ can be obtained by taking the max probability following the greedy decoding technique as shown in Equation (13),

$$\hat{w}_t = \arg\max_{w_t} p(w_t). \tag{13}$$

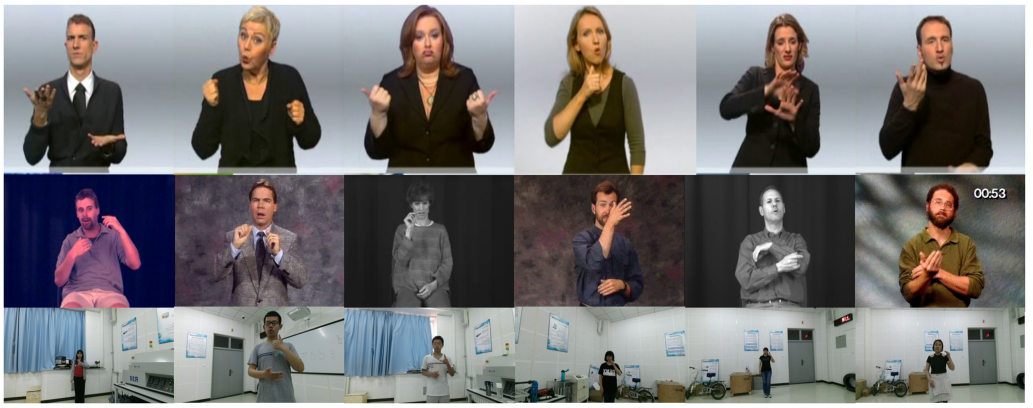

Fig. 5. Random samples from the datasets: (top) GSL dataset [60], (middle) ASL dataset [21], and (bottom) CSL dataset [110].

The Self-critic loss function can now be written as follows:

$$\nabla L(\theta) \approx -(R(w^s) - R(\hat{w}))\nabla_\theta log p_\theta(w^s). \tag{14}$$

The final part of Equation (14), $\nabla_\theta log p_\theta(w^s)$, is effectively equivalent to $\frac{\nabla p_\theta(w^s)}{p_\theta(w^s)}$, which is the column vector of the partial derivatives of $p_\theta(w^s)$ divided by the policy evaluation for that particular w^s. The form in the equation is used specifically to make it possible to update the loss function when doing gradient descent. In Figure 4, the symbol $\nabla L(\theta)$ is intended to represent the final gradient expression of the RL algorithm. This allows for the generation of a scalar loss value that the sign language to text models can use to backpropagate and learn. This can be contrasted to a more traditional loss function such as cross entropy.

5 DATASET INFORMATION

We evaluate these methods on a **German Sign Language (GSL)** dataset [60] and apply the best performing model and features on both a **Chinese Sign Language (CSL)** dataset [110] and an ASL dataset [21]. Some examples of frames from these datasets are shown in the Figure 5 to illustrate the nature of the input data we are working with.

5.1 German Sign Language

This data are collected from weather forecast airings from the RWTH-PHOENIX-Weather dataset [60]. The videos were recorded at 25 frames per second with 210×260 pixels as the frame size. Figure 6(a) highlights the word-level frequency for unique utterances. We use the same train/dev/test split as the original creators of the GSL dataset [60]. The GSL dataset signed by nine signers collected 7,096 training videos and ground-truth captions of which 6,811 are unique utterances as seen from the sentence repetition frequency in Figure 6(b). There are 1, 078 unique words in the dataset with 1, 042 repeated 2–10 times and other words spread out with frequencies greater than 11. Figures 7 and 8 shows that majority of the annotated sentences are signed by signer 1, signer 4, and signer 5 with an unequal distribution of sentences between the nine signers. The GSL dataset was collected in a controlled environment with similarly illuminated background for almost all the videos with color contrasts in the clothes worn by the signers as seen in the top row of Figure 5. The dataset is also confined to only weather-related sentences, thus spanning a confined subject area as shown in the word cloud in the Figures 7 and 8.

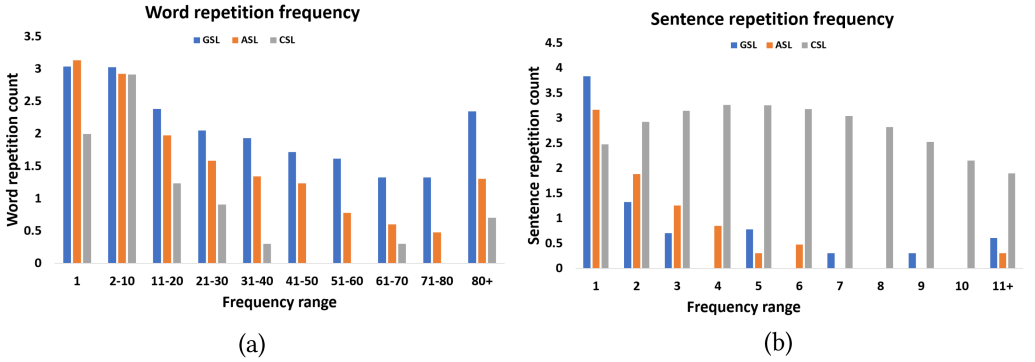

Fig. 6. (a) Word repetition frequency for unique utterances. (b) Sentence repetition frequency. The Y axis represents the log10 scale.

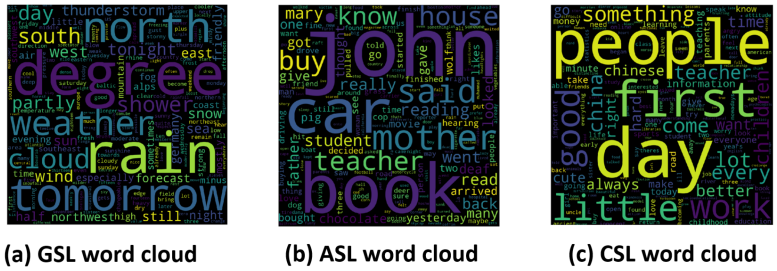

Fig. 7. Word clouds for GSL, CSL, and ASL datasets.

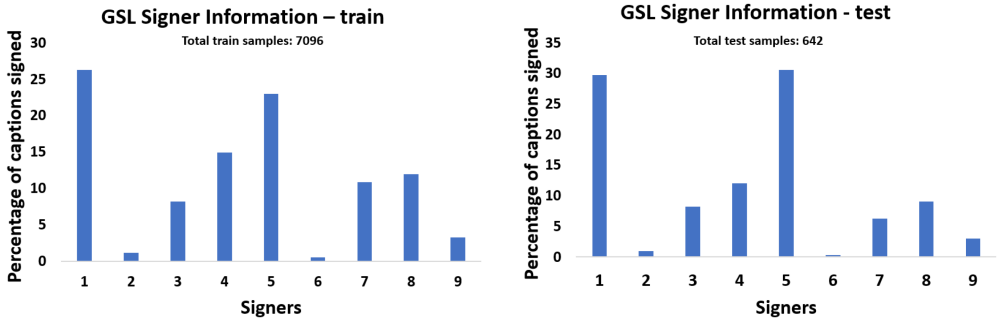

Fig. 8. Graph representing number of signs annotated by each signer in the GSL dataset.

5.2 American Sign Language

The ASL dataset [21] consists of videos narrating 38 different topics in American sign language. The frames were extracted at 25 frames per second with the resolution of videos ranging from 216×218 to 312×324. The extracted dataset has videos signed by 7 signers. The distribution between the signers is shown in Figure 9. The dataset consists of 38 stories. As the videos are approximately 1.5 to 2 minutes long on average, each video is divided into multiple sub-videos based on individual utterances. Due to this, some of the videos and utterances do not match one-to-one and the quality of the sub-videos are also affected poorly. The ASL dataset has around 1457 unique utterances. The word and sentence repetition frequency are shown in Figure 6(a) and

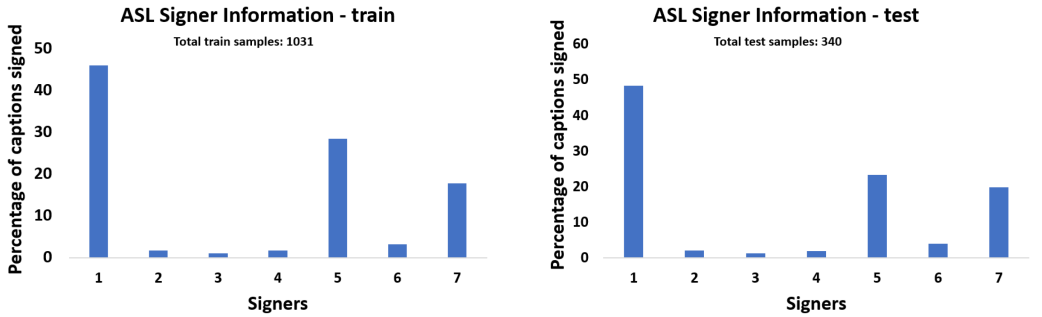

Fig. 9. Graph representing number of signs annotated by each signer in the ASL dataset.

Figure 6(b), respectively. From Figure 5, middle row, it can be seen that, very similarly to the GSL dataset, the ASL dataset also has a controlled background with contrast in cloth colors. The signers appear to be signing professionally with ease.

5.3 Chinese Sign Language

The Chinese Sign Language Dataset [110] consists of approximately 50,000 Chinese sign language videos; 1,000 utterances were assigned to 50 Deaf signers (25 male and 25 female). All signers have a fair distribution of utterances assigned for signing as shown in Figure 10, and 10,000 utterances have been signed in the CSL dataset. The videos were recorded at 30 frames per second. The color image frames are 1,920 × 1,080 pixels in size. The segmentation of words and characters specific to the Mandarin language were done using the Chinese word segmentation tool, Jieba.[2] Figure 6(a) shows the word-level frequency among unique utterances. Figure 6(b) shows the repetition frequency of the 10,000 utterances. This CSL dataset was based on a recently standardized sign language in China [110], and hence the signers used in this dataset are not native signers. They were provided training videos showing experts performing these new standardized signs, and the signers mimicked the videos by experts. These videos are taken in different settings and backgrounds and at different positions from the camera.

6 INPUT FEATURES

We extract various features from each frame from the sign language datasets and pass them in as inputs to the different models under consideration. These features have been meticulously chosen to obtain good performance. We extract OpenPose [10] features obtaining body, hand, and face joint locations, and AlexNet [48], and ResNet [32] CNN feature from the video. The CNN feature extraction models are pre-trained on ImageNet [23] dataset to extract a multi-dimensional feature vector from the visual frames. We also evaluate the use of k-means cluster IDs based on OpenPose joint locations. The details on each of the feature extractors are described in the following subsections.

6.0.1 OpenPose Features. Using OpenPose [10] we extract 25 body-joint keypoints, 21 keypoints for each hand, and 70 facial landmark keypoints. x, y and *confidence* are obtained for each of these 137 keypoints. Of these, we only choose the x, y coordinates of the joints. So for each frame, we have an input vector of 274 points. We keep the order of the joints the same as the original authors [10]. Because CSL has full-body information (see Figure 5) we use all 274 points, but for the GSL and ASL datasets, only the upper body landmarks are present. The other landmarks are zeroed

[2]Jieba Chinese text segmentation. https://github.com/fxsjy/jieba/.

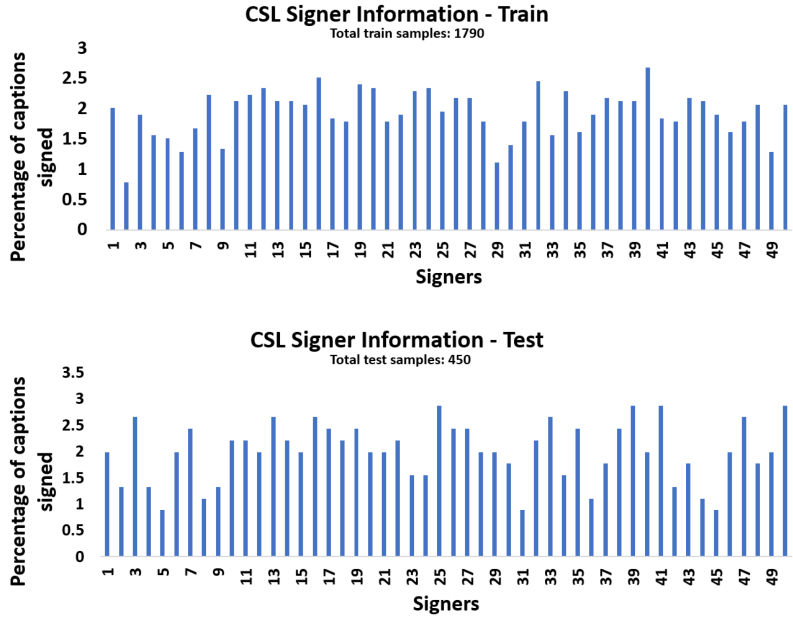

Fig. 10. Graph representing number of signs annotated by each signer in the CSL dataset.

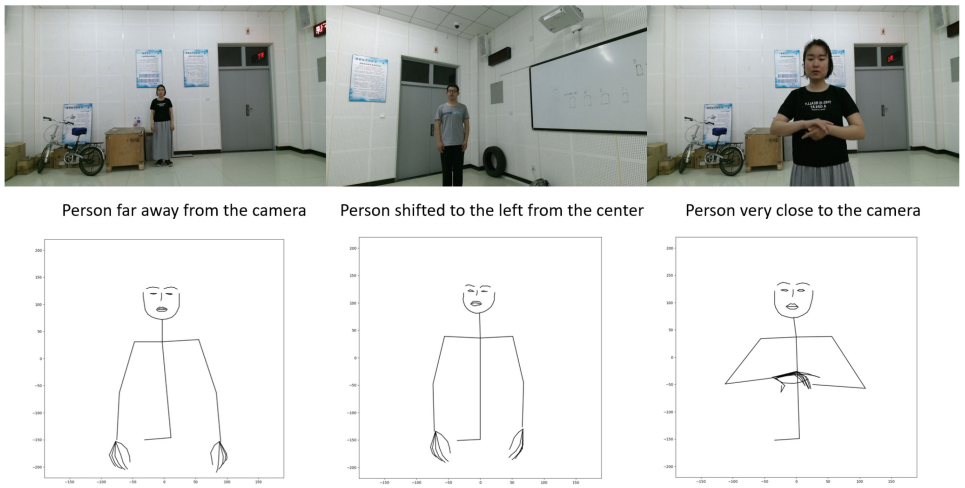

Fig. 11. Top figure shows sample frames from the CSLD, bottom figure shows the respective canonical representations of the frames.

out. From Figure 11 it can be seen that there are variations in the dataset based on where the person is standing relative to the camera position. To take care of this variation, We obtain a canonical and normalized representation based on the center body points. After mapping to canonical form, all the body points are centered to the origin and scaled to the same size. Figure 11 demonstrates the value of canonical representations. We additionally perform frame-to-frame smoothing of the OpenPose points. The OpenPose points are, first, median filtered to remove any noise in the data and, second, temporally smoothed using Savitzky-Golay [82] filtering mechanism.

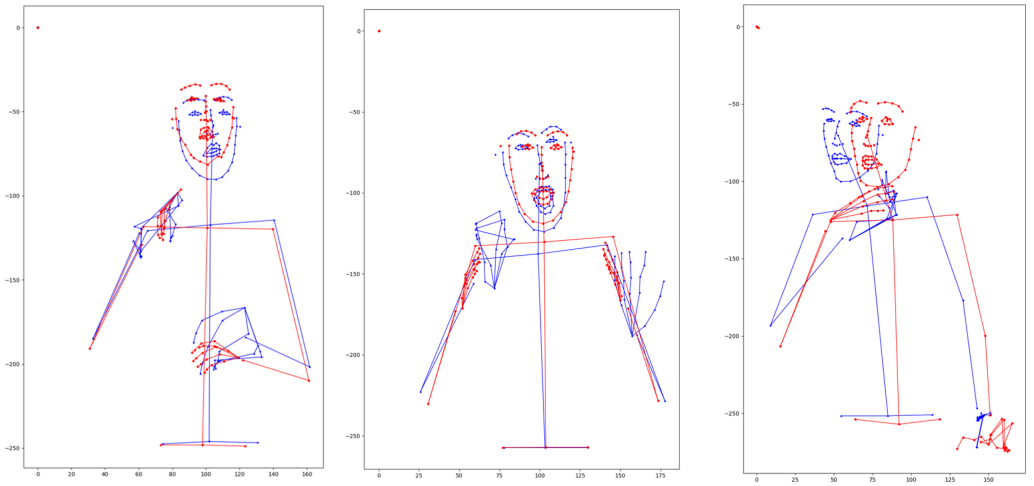

Fig. 12. The k-means cluster points (red) and original OpenPose points (blue) are plotted for a few random frames from the GSL dataset. The x and y axis represent the body joint locations.

6.0.2 CNN Features. We explore different CNN features to see which visual features stand out. The features include pretrained AlexNet (4,096 dimensional) [48], pretrained ResNet50 (2,4048 dimensional) [32], pretrained EfficientNet-B7 (2,560 dimensional) [93], and pretrained InceptionV3 (2,048 dimensional) [92]. These visual extractors are pretrained on ImageNet [23]. We also perform end-to-end training using InceptionV1 (1,024 dimensional) [91] and ResNet50 and use the trained network to extract the corresponding features.

6.0.3 Kmeans from OpenPose Features. We developed a pose dictionary where each frame belonged to a cluster based on the similarity of the keypoints and cluster center. Hence the entire video was represented as a vector of cluster IDs; 9,521 clusters gave the best performance.

Figure 12 shows how k-means groups similar poses together. Here we plot some frames belonging to particular cluster IDs.

7 ANALYSIS AND RESULTS

In this section, we will discuss our analysis and results obtained, involving different types of input features, performance metrics, and performance evaluation.

7.1 Performance Metrics

In the following subsections we briefly discuss the different performance metrics used in this work.

7.1.1 BLEU Scores. **BiLingual Evaluation Understudy (BLEU)** [68] BLEU score is a popular metric that compares a predicted caption with one or more ground-truth captions. The metric is based on a calculation that compares the n-grams (number of words per group) from the predicted caption to n-grams in the ground-truth caption. Examples of n-grams are unigrams (comparing single words), bigrams (comparing two words), trigrams (comparing three words), and so on. The unigram precision is calculated by counting the number of times a word occurs in the predicted sentence, this is then clipped to the maximum number of times that particular word has occurred in the ground-truth sentence. This result is divided by the total number of words in the predicted sentence. However, this method of measuring the BLEU score fails whenever the length of the predicted sentence is less than the ground-truth sentence. To overcome this problem, the predicted

sentence is penalized whenever its length is less than that of the ground-truth sentence. This penalty is called Brevity Penalty [68].

7.2 Results

This section contrasts different sign language to text models trained and evaluated on the GSL, ASL, and CSL datasets. Different ablations are performed by taking various kinds of feature vectors as input. For GSL, the models are trained on 7,096 videos and validated and tested on 519 and 642 videos, respectively. For CSL, 2240 videos have been chosen based on sentence repetition. The models are trained on 1,790 of these 2,240 videos and tested on 450 videos. After splitting the main story videos from the ASL dataset, we obtain 1, 031 training and 340 test videos. Other training details are mentioned according to the specific model under consideration in the subsections below. Unless otherwise mentioned, for GSL the maximum frame length and maximum caption length are set to 300 and 30, whereas for CSL it is set to 240 and 20 and for ASL it is set to 225 and 10, respectively. Each batch requires the same input and output length for processing. For this reason, most of our sequence models are designed in a way that can handle a fixed input length and a fixed output length. To avoid a lot of padding, the maximum frame is chosen to be the average frame length and the caption length is chosen to be the average caption length in the respective datasets. For training, we use NVIDIA RTX 2080 Ti graphics processor.

7.2.1 Training Details.

(1) Sequence-to-sequence model: The model is trained using Adam optimization with a learning rate and a decay of $1e^{-04}$ and $1e^{-05}$, respectively. We set the dimensionality of the hidden space to 256 for all the datasets the model is trained on. Results are reported on a bi-directional LSTM model with four layers. The model is trained with a batch size of 32. On average, the model takes 14 hours or 150 epochs to train on the GPU.
(2) Sequence-to-sequence model with attention: The model is trained using Adam optimization with a learning rate of $1e^{-05}$. Training is performed for 150,000 steps. We use Luong [55] attention for this sign language translation model. Training takes approximately four days on one GPU. We use a batch size of one as it proved to give the best results in Reference [17]. The model is trained with 1,000-dimensional hidden units and four layers of bi-directional LSTM.
(3) Transformer Model: While training the transformer model, for each batch the maximum length of the sequence in the batch is chosen as the number of frames and likewise for the number of captions. Batch normalization and softsign [62] activation are used on input features before feeding to the network as it gives us good results. The model is trained for 100 epochs with a batch size of 32 and Adam optimizer.
(4) Reinforcement Learning: The model is trained using Adam optimization with a learning rate and a decay of $5e^{-04}$ and $5e^{-05}$, respectively. We set the dimensionality of the hidden space to 256. Results are reported on a bi-directional LSTM model with two layers. The model is trained with a batch size of 32. On average, the model takes 15 hours or 280 epochs to train on the GPU. The model is pre-trained using cross-entropy for 56 epochs and then trained using self-critical sequence training for the remaining 224 epochs.

7.3 Analysis of Results

We begin our analysis by considering the GSL dataset mainly due to two reasons, one, the dataset consists of samples portraying a controlled environment, and two, because the dataset has highest training samples and a good distribution of samples in the validation and test sets as compared to the other datasets.

Table 3. OpenPose Ablation Results on Sequence-to-Sequence Model without Attention on the GSL Dataset

OpenPose Features	Set	BLEU 1	BLEU 2	BLEU 3	BLEU 4
Hands	Validation	7.31	1.36	0.34	0.07
	Test	12.14	1.71	0.55	0.12
Body	Validation	4.63	0.88	0.18	0.01
	Test	6.29	0.82	0.14	0.01
Face	Validation	2.54	0.46	0.04	0.01
	Test	2.23	0.40	0.04	0.01
Hands + Body	Validation	18.11	6.50	4.42	4.06
	Test	20.68	7.01	4.65	4.33
Hands + Face	Validation	18.13	5.92	4.51	4.06
	Test	17.68	5.76	4.47	4.04
Hands + Body + Face	Validation	**23.31**	**8.73**	**6.49**	**5.68**
	Test	**23.49**	**8.52**	**6.36**	**5.55**

Table 4. Ablations on Basic Sequence-to-Sequence Models without Attention on the GSL Dataset Using Openpose Points without Frame-to-Frame Smoothing

Model	Set	BLEU 1	BLEU 2	BLEU 3	BLEU 4
Vanilla RNN	Validation	12.67	4.09	3.17	2.79
	Test	14.31	4.68	3.43	2.92
GRU	Validation	20.16	7.41	5.29	4.66
	Test	20.40	7.56	5.63	4.98
LSTM	Validation	**23.62**	**8.89**	**6.54**	**5.71**
	Test	**22.89**	**8.43**	**6.20**	**5.57**

7.3.1 Variants of OpenPose for Sign Language Translation. OpenPose joints keypoints gained popularity from gesture and action recognition tasks [9, 10, 86, 102] as already see in the Section 3. We want to explore its performance for sign language translation. In addition, OpenPose joints are very human interpretable. It highlights the importance of joints for particular signs. We want to investigate how different OpenPose joints contribute toward the performance of sign language translation. The ablations with OpenPose features on the GSL dataset are shown in the Table 3.

Observations. From our observation, hands are the most important keypoints from OpenPose that is useful for translation. Even though hands alone does not yield high scores, it improves when combined with face and body, whereas it jumps significantly when all three are combined.

7.3.2 Choosing among Vanilla RNN, GRU, and LSTM. We perform sign language to text translation on the GSL dataset using Vanilla RNN, GRU, and LSTM bi-directional models by feeding OpenPose key points as input. From the results shown in Table 4, we can see that Vanilla RNN performs the worst when compared to LSTM and GRU.

Observations. We can thus infer that LSTM and GRU are capable of handling long-term dependencies when compared to Vanilla RNN. For all the sequence-to-sequence modeling used in our experiments henceforth, we will be using LSTM as the RNN model.

7.3.3 Ablations with Different CNN Features. To choose an appropriate CNN architecture, we perform ablations using the embedding features extracted from the architectures. These CNNs (ResNet50, AlexNet, EfficientNet-B7, InceptionV1, V3) are pretrained on ImageNet and used as

Table 5. Ablation Results on Different CNN Features Using the Transformer Model on the GSL Dataset

CNN Features	Set	BLEU 1	BLEU 2	BLEU 3	BLEU 4
Pretrained ResNet50	Validation	**24.75**	**16.04**	**11.56**	**8.95**
	Test	**23.67**	**14.58**	**10.29**	**8.00**
Pretrained AlexNet	Validation	23.81	14.71	10.65	8.32
	Test	23.00	13.86	9.75	7.5
Pretrained EfficienetNet-B7	Validation	20.84	12.4	8.78	6.9
	Test	18.84	11.36	8.06	6.31
Pretrained Inceptionv3	Validation	19.90	11.75	8.44	6.67
	Test	18.71	11.08	7.93	6.19
Pretrained Inceptionv1	Validation	17.7	10.26	7.28	5.72
	Test	18.18	10.63	7.39	5.67

feature extractors, whereas we performed end-to-end training by freezing the first three layers of InceptionV1 and ResNet50 in a sequence-to-sequence setup. As the end-to-end training needs a lot of memory and takes many days to converge, our models did not have the opportunity to learn as well. All the features were evaluated using the transformer model. The results of the ablations are presented in Table 5.

Observations. As seen in Table 5, pretrained ResNet50 features provide the best results. Going forward for all the ablations with CNN features we will be using ResNet50 extracted features.

From our analysis, we have observed that SLT datasets benefit from features extracted using deep networks. When comparing with shallow networks (AlexNet), deep networks (ResNet50) provides better features giving better performance in SLT. Empirically, from our studies, we have seen that wider networks do not perform as well. So Inception v1, Inception v3, EfficientNet-B7 (even though is not only wide but also deep) do not seem to perform as well as a purely deep network. The deep network architecture and skip-connections in ResNet50 prove to be the most beneficial for SLT yielding the best results.

7.3.4 Experiments and Ablations on GSL Dataset. From the experiments and ablations in Table 6, we study how different feature inputs contribute toward improving the model and how attention plays an important role in sign language to text translation. Visual features from ResNet50 and location-based features from OpenPose smoothed version perform better than other features. Between the sequence-to-sequence models, attention mechanism predicts *bi-grams* and longer *n-grams* better than the sequence-to-sequence model without attention. The transformer model performs further better than the sequence-to-sequence model with attention in predicting *bi-grams* and more due to its extensive multi-head attention and self-attention mechanism. The results of the ablations are presented in Table 6.

Observations. Compared to the other models evaluated, the transformer can learn more weights and parameters that contribute toward its good performance on the GSL dataset as shown in Table 6. The multi-head self-attention mechanism in the transformer model can learn complex representations for every frame. The self-attention in the transformer model helps to learn interdependencies between different frames depending upon the position of the frames and the ground-truth caption. The self-attentions are calculated in parallel, eight times, referred to as eight heads, and concatenated together forming multi-head attention. This parallelization has an advantage in the transformer model as against the sequential sequence-to-sequence models. The advanced architecture of the transformer model helps in learning long-term dependencies when compared to the

Table 6. Ablation Results on Different Feature Inputs Using the GSL Dataset

Input Features	Model	Set	BLEU 1	BLEU 2	BLEU 3	BLEU 4
OpenPose (NS)	s2s	Validation	23.62	8.89	6.54	5.71
		Test	**22.89**	8.43	6.20	5.57
	s2s with att	Validation	17.64	11.42	8.52	6.78
		Test	18.50	11.92	8.73	6.95
	Transformer model	Validation	**24.2**	**15.28**	**11.08**	**8.69**
		Test	22.69	**14.61**	**10.38**	**8.09**
OpenPose (WS)	s2s	Validation	23.31	8.73	6.49	5.68
		Test	23.49	8.52	6.36	5.55
	s2s with att	Validation	20.65	13.39	9.80	7.68
		Test	19.65	12.36	8.97	7.00
	Transformer model	Validation	**24.82**	**15.87**	**11.53**	**9.07**
		Test	**23.52**	**15.24**	**11.00**	**8.67**
ResNet50	s2s	Validation	21.55	7.97	5.76	4.96
		Test	20.90	7.81	5.08	4.22
	s2s with att	Validation	23.01	14.48	10.20	7.79
		Test	21.71	13.89	10.07	7.87
	Transformer model	Validation	**24.75**	**16.04**	**11.56**	**8.95**
		Test	**23.67**	**14.58**	**10.29**	**8.00**
k-means	s2s	Validation	**15.68**	3.67	3.12	2.74
		Test	**16.93**	3.90	3.32	2.95
	s2s with att	Validation	12.75	6.92	5.06	4.07
		Test	12.78	7.30	5.51	**4.54**
	Transformer model	Validation	14.02	**7.47**	**5.44**	**4.41**
		Test	13.65	**7.46**	**5.57**	**4.54**

NS - No Smoothing, WS - With Smoothing,
s2s - basic sequence-to-sequence model without attention,
s2s with att - sequence-to-sequence model with attention.

sequence-to-sequence models. Although ResNet50 and OpenPose input features perform very similar with the highest BLEU scores on the transformer model, OpenPose features are less expensive to get than ResNet50 features.

7.3.5 Experiments and Ablations on CSL Dataset. Preliminary results on the CSL dataset [110] are shown in Table 7. The CSL dataset chosen for training is a very small subset of the whole dataset. Due to the language complexity and abnormalities in the dataset, there is still room for improvement in the Chinese Sign Language recognition models. BLEU 1 scores are considerably high when compared to BLEU 2 scores, because a few characters like "我 (I)" occur 2,103 times in the dataset followed by "是 (is)," "他 (he)," "你 (you)," repeating around 1,000 times. For the CSL dataset, OpenPose feature inputs work better than CNN features because of the structure of the dataset. As explained in Section 6 and Figure 11 the OpenPose canonical form obtain a scaled and center version of points for all the frames, whereas the CNN features still take in as input the original frames where the person signing may not be at the center and the same distance from the camera. For such a real-life scenario dataset consisting of different abnormalities and noise, OpenPose proves to be beneficial while performing sign language to text translation.

The attention mechanisms (sequence-to-sequence with attention and transformer model) perform very well with OpenPose input features. The transformer model, however, performs the best with high BLEU 1 to BLEU 4 scores for the pose-based inputs. Whereas, the CNN feature input overall does not do very well due to abnormalities in the dataset.

Table 7. Ablation Results on Different Feature Inputs Using the CSL Dataset

Input Features	Model	Set	BLEU 1	BLEU 2	BLEU 3	BLEU 4
OpenPose (NS)	s2s	Test	14.36	1.60	0.04	0.01
	s2s with att	Test	39.45	34.44	32.55	31.43
	Transformer model	Test	**52.04**	**48.94**	**47.86**	**47.25**
OpenPose (WS)	s2s	Test	19.68	2.32	0.12	0.01
	s2s with att	Test	36.13	31.11	29.48	28.63
	Transformer model	Test	**43.03**	**38.93**	**37.52**	**36.76**
ResNet50	s2s	Test	**17.41**	1.19	0.12	0.01
	s2s with att	Test	14.01	5.16	3.55	2.93
	Transformer model	Test	13.2	**6.04**	**4.58**	**4.02**

NS - No Smoothing, WS - With Smoothing,
s2s - basic sequence-to-sequence model without attention
s2s with att - sequence-to-sequence model with attention.

Observations. In situations where the CNN features are not as good due to the structure of the dataset, OpenPose feature inputs help in elevating the results.

7.3.6 Experiments and Ablations on ASL Dataset. Preliminary results with the ASL dataset are shown in Table 8. The sequence-to-sequence model with attention follows a similar trend like GSL and CSL in predicting longer *n-grams* better for different feature inputs. It is evident from Tables 6, 7, and 8 that the organization of the dataset highly influences the results. While ASL is very similar to the GSL dataset, it has fewer samples and covers a widespread topic rather than just weather information (like GSL). It is very different from the CSL dataset with respect to the number of signers, the number of samples, and the type of samples. The OpenPose and CNN feature inputs perform comparatively the same for the ASL dataset using the attention models. The sequence-to-sequence model with attention performs the best when compared with the transformer model and the sequence-to-sequence model without attention. The transformer model does not provide a boost in BLEU scores as seen in the other datasets.

Observations. The organization of the ASL dataset is somewhere between the GSL and CSL dataset. Because of the input frames having a controlled background, it performs better than CSL with CNN features but due to the poor quality of frames when compared to CSL, it incurs a performance hit when using OpenPose points. The poor performance of the transformer model may be due to the ASL being a smaller dataset with fewer repetitions as seen in Figure 6 and with 10% of sentences in the test set seen during training. The CSL dataset is chosen such that the dataset has sentences that are repeated due to this there is a high overlap of the captions between test and train but these are all signed by different signers. GSL, however, being a huge dataset, performs fairly better than ASL even with less number of test samples seen during training. To further understand the ASL dataset, we perform experiments to see how different signers interpret the same sign, which is further explained in Section 7.3.8.

7.3.7 Addition Experiments and Analysis Using RL. Reinforcement learning is also used as one of the models to analyze how it tackles the *exposure bias* problem when compared to other methods. Using a rewards mechanism, the final gradient is calculated, which is then backpropagated to the model under consideration. We report preliminary results with RL on the sequence-to-sequence model without attention using ResNet features on the GSL dataset. The results are shown in Table 9.

Table 8. Ablation Results on Different Feature Inputs Using the ASL Dataset

Input Features	Model	Set	BLEU 1	BLEU 2	BLEU 3	BLEU 4
OpenPose (NS)	s2s	Test	14.68	3.81	2.76	2.31
	s2s with att	Test	**19.83**	**8.45**	**4.79**	2.93
	Transformer model	Test	13.58	6.7	4.35	**3.17**
OpenPose (WS)	s2s	Test	16.85	4.17	3.29	3.02
	s2s with att	Test	**20.20**	**8.62**	**5.21**	**3.43**
	Transformer model	Test	13.57	6.83	4.46	3.34
ResNet50	s2s	Test	**21.66**	4.95	4.43	**4.26**
	s2s with att	Test	19.55	**10.02**	**6.2**	4.25
	Transformer model	test	12.78	5.7	3.38	2.43

NS - No Smoothing, WS - With Smoothing,
s2s - basic sequence-to-sequence model without attention,
s2s with att - sequence-to-sequence model with attention.

Table 9. Results on the GSL Dataset Using RL-based Sequence-to-Sequence (s2s) Model without Attention with ResNet Input Features

Model	Set	BLEU 1	BLEU 2	BLEU 3	BLEU 4
s2s without RL	Validation	21.55	7.97	5.76	4.96
	Test	20.90	7.81	5.08	4.22
s2s with RL	Validation	26.97	8.94	7.08	6.48
	Test	25.70	7.87	6.31	5.64

Table 10. Human as an Oracle Experiment on ASL Dataset for Original (Sign Language RGB Video) and OpenPose Generated Videos

Video type	BLEU 1	BLEU 2	BLEU 3	BLEU 4
Original videos	24.86	12.17	6.87	3.81
OpenPose videos	6.32	1.790	0.59	0.14

Observations. The RL model performs better than the sequence-to-sequence model without attention. Thus the RL model without teacher-forcing can perform well when compared to the model that incorporates teacher-forcing thus reducing the *exposure-bias* problem. This model is also able to predict *one-gram* words better than the sequence-to-sequence model with attention. In the future, it would be interesting to see how the RL model performs if attention is introduced.

7.3.8 Human as an Oracle. We performed an experiment with a human as an oracle (an expert signer) to compare how a human is able to predict the captions on a sign language video versus OpenPose joints visualization for each frame rendered as a video. The OpenPose joints are mainly used for this experiment, because the OpenPose annotated body, hands, and face joints are highly interpretable by humans. The results on 340 ASL videos from the test set and OpenPose videos are shown in Table 10.

The low BLEU 4 scores in Table 10 for original videos may be attributed to the fact that the ground-truth captions were taken out of context from a story telling scenario. Therefore, the signs and captions may not necessarily match up. Ideally, 50% BLEU 4 score from human evaluation would indicate that the ground truth is collected efficiently. To test the efficacy of the dataset

Table 11. Comparison between Ground Truth and Predicted Captions between Two Human Annotators

No.	Ground-truth caption	ASL Signer 1	ASL Signer 2
1.	He called and asked if they had the right kind of dog, and the answer was yes!	You can have	they shouted "you have it?", "yes".
2.	He keeps drinking coffee	Still making out	coffee continued still.
3.	It must be a head trip or something	I was tripping	I'm strict/hardheaded.
4.	As I was driving, I was really bored. So I started thinking about what I could do.	The car ride was boring	Driving's boring.
5.	Did the teacher buy a house yesterday?	Did the teacher buy a house yesterday?	Did the teacher buy a house yesterday?

The yellow cells show agreement while green shows disagreement (Best viewed in color).

collection further, we asked another ASL signer to annotate five videos and compared the results of the two signers with the ground truth.

Observations. From Table 11 we can see that the predicted sentences have non-matching *grams* thus leading to low BLEU scores in Table 10.

8 CONCLUSION

There is still much to be done to facilitate smooth communication between hearing and non-hearing communities. In this article, we have discussed how methodologies evolved from text-to-text translation to sign-to-text translation. We selected four models to evaluate: the sequence-to-sequence model, the sequence-to-sequence model with attention, the transformer model, and the reinforcement-learning–based model; to better understand how well they perform for sign language translation. We also explored different input features such as the joints of the body and hands, along with facial landmarks all using OpenPose, CNN features, and k-means cluster IDs for each frame. To better understand which sets and subsets of features aid best in improving performance, we performed ablation studies using different combinations of body, hands, and face keypoints as inputs to our models. In addition to ablations with different input features, we also examined different sign language datasets to better understand how the performance varied, based on the complexity (controlled collection and constrained vocabulary) of the sign language videos. We evaluated our models on different sign languages, ASL, GSL, and CSL. Our transformer model outperformed all the other sequence-to-sequence models on the GSL and CSL datasets using OpenPose features. In some cases, ResNet features provided similar results as OpenPose but as seen it is highly dataset dependent. ASL dataset could not leverage the attention mechanisms involved in the transformer model due to being a smaller and noisy dataset. In general, the attention mechanisms involved in the transformer model learned more weights than the sequence-to-sequence models with and without attention, thus leading to significantly better learning capability.

To provide the reader with a comprehensive study for sign language translation, we performed a preliminary experiment using ResNet features with the sequence-to-sequence model without attention and RL. This RL-based model helped in mitigating the exposure bias issue usually seen due to teacher-forcing when sequence-to-sequence models are used. Our RL-based sequence-to-sequence model yielded approximately a 1.5- to 5-point increase in BLEU 1–BLEU 4 scores. Our human-as-an-oracle experiment completed our extensive study for this article by looking at how

a human expert signer would interpret the ASL videos and their corresponding OpenPose-based stick videos. We used OpenPose for comparison as it is visually more understandable for predicting signs when compared to other features.

In conclusion, this article has provided a comprehensive overview of the representations and models used for continuous sign language translation without gloss. We have shown that the transformer model performed the best on the the controlled and constrained GSL dataset, especially when combined with either OpenPose or ResNet50 as input features. But for the other less consistent datasets, the results were more varied. Also, the human experiment demonstrated that our ASL dataset is very noisy (with several incorrect captions), resulting in low BLEU scores being obtained even by a human expert and low agreement between two different signers. In the future, we plan to collect a larger and more consistent ASL dataset, to make it more on par with the GSL and CSL datasets.

APPENDIX

A SAMPLE VIDEOS FROM DIFFERENT DATASETS FOR SAME UTTERANCE

This section shows sign language videos of the same utterance signed by different signers. The videos can be downloaded from here. These videos can be best played using a VLC player. From the videos, we can see that GSL and ASL utterances signed by different signers look very similar in terms of signs and speed, whereas the CSL videos show the difference in speed and slight variation in the signs between the two signers. These examples mimic the entire dataset very closely.

REFERENCES

[1] Biyi Fang, Jillian Co, and Mi Zhang. 2017. DeepASL: Enabling ubiquitous and non-intrusive word and sentence-level sign language translation. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. 1–13.

[2] Openpose 2D. 2018. Retrieved January 4, 2020 from https://github.com/CMU-Perceptual-Computing-Lab/openpose.

[3] Nayyer Aafaq, Syed Zulqarnain Gilani, Wei Liu, and Ajmal Mian. 2018. Video description: A survey of methods, datasets and evaluation metrics. *ACM Comput. Surv.* 52, 6, Article 115 (Oct. 2019), 37 pages. https://doi.org/10.1145/3355390

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR'15)*, San Diego, CA, USA, May 7-9, 2015, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1409.0473

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1409.0473

[6] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. 2017. Hierarchical boundary-aware neural encoder for video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*.

[7] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 1877–1901. https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[8] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. 2020. Sign language transformers: Joint end-to-end sign language recognition and translation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 10020–10030. https://doi.org/10.1109/CVPR42600.2020.01004

[9] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. 2019. OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Trans. Pattern Anal. Mach. Intell.* (2019).

[10] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2017. Realtime multi-person 2D pose estimation using part affinity fields. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'17)*.

[11] Ming Chen, Yingming Li, Zhongfei Zhang, and Siyu Huang. 2018. TVT: Two-view transformer network for video captioning. In *Proceedings of the 10th Asian Conference on Machine Learning*, Jun Zhu and Ichiro Takeuchi (Eds.). PMLR, 847–862. http://proceedings.mlr.press/v95/chen18b.html.

[12] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Association for Computational Linguistics, Doha, Qatar, 103–111. https://doi.org/10.3115/v1/W14-4012

[13] KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Association for Computational Linguistics, Doha, Qatar, 103–111. https://doi.org/10.3115/v1/W14-4012

[14] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555. Retrieved from https://arxiv.org/abs/1412.3555.

[15] Joon Son Chung, Andrew Senior, Oriol Vinyals, and Andrew Zisserman. 2017. Lip reading sentences in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. IEEE, 3444–3453.

[16] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, and Richard Bowden. 2017. SubUNets: End-to-end hand shape and continuous sign language recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

[17] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. 2018. Neural sign language translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 7784–7793.

[18] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 2978–2988. https://doi.org/10.18653/v1/P19-1285

[19] Simona Damian. 2011. Spoken vs. sign languages—What's the difference? *Cogn. Brain Behav.* 15, 2 (2011), 251.

[20] Abhishek Das, Satwik Kottur, Jose M. F. Moura, Stefan Lee, and Dhruv Batra. 2017. Learning cooperative visual dialog agents with deep reinforcement learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'17)*.

[21] American Sign Language Dataset. Retrieved May 9, 2020 from http://www.bu.edu/asllrp/.

[22] American Sign Language Dataset. Retrieved May 9, 2020 from http://www.bu.edu/asllrp/.

[23] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 248–255.

[24] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[25] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2625–2634.

[26] Biyi Fang, Jillian Co, and Mi Zhang. 2017. DeepASL: Enabling ubiquitous and non-intrusive word and sentence-level sign language translation. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. 1–13.

[27] AlphaGo Zero: Starting from scratch. Retrieved May 1, 2020 from https://deepmind.com/blog/article/alphago-zero-starting-scratch.

[28] L. Gao, Z. Guo, H. Zhang, X. Xu, and H. T. Shen. 2017. Video captioning with attention-based LSTM and semantic consistency. *IEEE Trans. Multimedia* 19, 9 (2017), 2045–2055.

[29] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. 1243–1252.

[30] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*. 369–376.

[31] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. 2017. Learning spatio-temporal features with 3D residual networks for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 3154–3160.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

[33] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9, 8 (Nov. 1997), 17351780. https://doi.org/10.1162/neco.1997.9.8.1735

[34] Markus Hosemann and Jana Steinbach. [n.d.]. Atlas of Sign Language Structures.
[35] Shiliang Huang, Chensi Mao, Jinxu Tao, and Zhongfu Ye. 2018. A novel chinese sign language recognition method based on keyframe-centered clips. *IEEE Sign. Process. Lett.* 25, 3 (2018), 442–446.
[36] Five Paramters in Sign Language. Retrieved March 24, 2021 from https://www.lifeprint.com/asl101/pages-layout/parameters.htm/.
[37] Glossing in Sign Language. Retrieved April 23, 2020 from https://www.lifeprint.com/asl101/topics/gloss.htm.
[38] Glossing in Sign Language. Retrieved April 23, 2020 from https://www.startasl.com/sign-language-symbols/.
[39] Eldar Insafutdinov, Mykhaylo Andriluka, Leonid Pishchulin, Siyu Tang, Evgeny Levinkov, Bjoern Andres, and Bernt Schiele. 2016. Articulated multi-person tracking in the wild. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1293–1301.
[40] Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1700–1709.
[41] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1725–1732.
[42] Sang-Ki Ko, Chang Jo Kim, Hyedong Jung, and Choongsang Cho. 2019. Neural sign language translation based on human keypoint estimation. *Appl. Sci.* 9, 13 (2019), 2683.
[43] Jens Kober, J. Andrew Bagnell, and Jan Peters. 2013. Reinforcement learning in robotics: A survey. *Int. J. Robot. Res.* 32, 11 (2013), 1238–1274.
[44] O. Koller, N. C. Camgoz, H. Ney, and R. Bowden. 2020. Weakly supervised learning with multi-stream CNN-LSTM-HMMs to discover sequential parallelism in sign language videos. *IEEE Trans. Pattern Anal. Mach. Intell.* 42, 9 (2020), 2306–2320. https://doi.org/10.1109/TPAMI.2019.2911077
[45] Oscar Koller, Hermann Ney, and Richard Bowden. 2015. Deep learning of mouth shapes for sign language. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 85–91.
[46] Oscar Koller, Hermann Ney, and Richard Bowden. 2016. Deep hand: How to train a cnn on 1 million hand images when your data is continuous and weakly labelled. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3793–3802.
[47] Petar Kormushev, Sylvain Calinon, and Darwin G. Caldwell. 2013. Reinforcement learning in robotics: Applications and real-world challenges. *Robotics* 2, 3 (2013), 122–148.
[48] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 1097–1105.
[49] Dongxu Li, Cristian Rodriguez Opazo, Xin Yu, and Hongdong Li. 2019. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. arXiv abs/1910.11006. Retrieved from https://arxiv.org/abs/1910.11006.
[50] L. Li and B. Gong. 2019. End-to-end video captioning with multitask reinforcement learning. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV'19)*. 339–348.
[51] Lijun Li and Boqing Gong. 2019. End-to-end video captioning with multitask reinforcement learning. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV'19)*. IEEE, 339–348.
[52] Xiaoxu Li, Chensi Mao, Shiliang Huang, and Zhongfu Ye. 2017. Chinese sign language recognition based on SHS Descriptor and Encoder-Decoder LSTM Model. In *Proceedings of the Chinese Conference on Biometric Recognition*. Springer, 719–728.
[53] Stuart Lloyd. 1982. Least squares quantization in PCM. *IEEE Trans. Inf. Theory* 28, 2 (1982), 129–137.
[54] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/c74d97b01eae257e44aa9d5bade97baf-Paper.pdf.
[55] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, 1412–1421. https://doi.org/10.18653/v1/D15-1166
[56] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1. Oakland, CA, 281–297.
[57] Chensi Mao, Shiliang Huang, Xiaoxu Li, and Zhongfu Ye. 2017. Chinese sign language recognition with sequence to sequence learning. In *Proceedings of the CCF Chinese Conference on Computer Vision*. Springer, 180–191.
[58] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*. arXivpreprint-arXiv:1312.5602.

[59] Boris Mocialov, Graham Turner, Katrin Lohan, and Helen Hastie. 2017. Towards continuous sign language recognition with deep learning. In *Proceedings of the Workshop on the Creating Meaning With Robot Assistants: The Gap Left by Smart Devices*.

[60] Oscar Koller, Hermann Ney, Richard Bowden, Necati Cihan Camgöz, and Simon Hadfield. 2018. RWTH-PHOENIX-Weather 2014 T: Parallel corpus of sign language video, gloss and translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT*.

[61] Noriki Nishida and Hideki Nakayama. 2015. Multimodal gesture recognition using multi-stream recurrent neural network. In *Image and Video Technology*. Springer, 682–694.

[62] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. 2018. Activation functions: Comparison of trends in practice and research for deep learning. CoRR abs/1811.03378 (2018). arXiv:1811.03378 http://arxiv.org/abs/1811.03378.

[63] Silvio Olivastri, Gurkirt Singh, and Fabio Cuzzolin. 2019. End-to-End video captioning. In *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW'19)*. 1474–1482.

[64] Tiago Oliveira, Nuno Escudeiro, Paula Escudeiro, Emanuel Rocha, and Fernando Maciel Barbosa. 2019. The virtual-sign channel for the communication between deaf and hearing users. *IEEE Rev. Iberoam. Tecnol. Aprend.* (2019).

[65] OpenFace. 2018. Retrieved January 4, 2020 from https://github.com/TadasBaltrusaitis/OpenFace.

[66] World Health Organization. 2018. Retrieved January 4, 2020 from https://www.who.int/news-room/facts-in-pictures/detail/deafness/.

[67] Yingwei Pan, Ting Yao, Houqiang Li, and Tao Mei. 2017. Video captioning with transferred semantic attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*.

[68] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL'02)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 311–318. https://doi.org/10.3115/1073083.1073135

[69] Marcus Perlman, Hannah Little, Bill Thompson, and Robin L. Thompson. 2018. Iconicity in signed and spoken vocabulary: A comparison between american sign language, british sign language, english, and spanish. *Front. Psychol.* 9 (2018), 1433.

[70] Jan Peters, Sethu Vijayakumar, and Stefan Schaal. 2003. Reinforcement learning for humanoid robotics. In *Proceedings of the 3rd IEEE-RAS International Conference on Humanoid Robots*. 1–20.

[71] Roland Pfau and Josep Quer. 2010. Nonmanuals: Their prosodic and grammatical roles. *Sign Lang.* (2010), 381–402.

[72] Lionel Pigou, Sander Dieleman, Pieter-Jan Kindermans, and Benjamin Schrauwen. 2014. Sign language recognition using convolutional neural networks. In *Proceedings of the European Conference on Computer Vision*. Springer, 572–578.

[73] Lionel Pigou, Aäron Van Den Oord, Sander Dieleman, Mieke Van Herreweghe, and Joni Dambre. 2018. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *Int. J. Comput. Vis.* 126, 2-4 (2018), 430–439.

[74] Lionel Pigou, Mieke Van Herreweghe, and Joni Dambre. 2017. Gesture and sign language recognition with temporal residual networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshop (ICCVW'17)*. IEEE, 3086–3093.

[75] Junfu Pu, Wengang Zhou, and Houqiang Li. 2018. Dilated convolutional network with iterative optimization for continuous sign language recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'18)*. 885–891.

[76] Alec Radford. 2018. Improving language understanding by generative pre-training. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/languageunderstandingpaper.pdf.

[77] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019), 9.

[78] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *Proceedings of the 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1511.06732

[79] Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. 2017. Deep reinforcement learning-based image captioning with embedding reward. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 290–298.

[80] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 7008–7024.

[81] Wendy Sandler and Diane Lillo-Martin. 2006. *Sign Language and Linguistic Universals*. Cambridge University Press. https://doi.org/10.1017/CBO9781139163910

[82] Abraham Savitzky and Marcel J. E. Golay. 1964. Smoothing and differentiation of data by simplified least squares procedures. *Anal. Chem.* 36, 8 (1964), 1627–1639.

[83] Zhiqiang Shen, Jianguo Li, Zhou Su, Minjun Li, Yurong Chen, Yu-Gang Jiang, and Xiangyang Xue. 2017. Weakly supervised dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*.

[84] Haichao Shi, Peng Li, Bo Wang, and Zhenyu Wang. 2018. Image captioning based on deep reinforcement learning. In *Proceedings of the 10th International Conference on Internet Multimedia Computing and Service*. 1–5.

[85] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *Nature* 550, 7676 (2017), 354–359.

[86] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. 2017. Hand keypoint detection in single images using multiview bootstrapping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*.

[87] Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. 2019. Contrastive bidirectional transformer for temporal representation learning. arXiv:1906.05743. Retrieved from http://arxiv.org/abs/1906.05743.

[88] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. VideoBERT: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'19)*.

[89] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*. 3104–3112.

[90] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. MIT Press.

[91] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going Deeper with Convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1–9. https://ieeexplore.ieee.org/document/7298594.

[92] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the inception architecture for computer vision. 2818–2826. https://doi.org/10.1109/CVPR.2016.308

[93] Mingxing Tan and Quoc V. Le. 2019. EfficientNet: Rethinking model scaling for convolutional neural networks. 97 (2019), 6105–6114. https://proceedings.mlr.press/v97/tan19a.html.

[94] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (Long Beach, California, USA) (NIPS'17)*. Curran Associates Inc., 6000–6010.

[95] Mel Vecerik, Todd Hester, Jonathan Scholz, Fumin Wang, Olivier Pietquin, Bilal Piot, Nicolas Heess, Thomas Rothörl, Thomas Lampe, and Martin Riedmiller. 2017. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. arXiv:1707.08817. Retrieved from https://arxiv.org/abs/1707.08817.

[96] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4566–4575.

[97] Subhashini Venugopalan, Marcus Rohrbach, Jeff Donahue, Raymond J. Mooney, Trevor Darrell, and Kate Saenko. 2015. Sequence to sequence—Video to text. In *ICCV*. IEEE Computer Society, 4534–4542. http://dblp.uni-trier.de/db/conf/iccv/iccv2015.html#VenugopalanRDMD15.

[98] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3156–3164.

[99] Christian Vogler and Carol Neidle. 2012. A new web interface to facilitate access to corpora: Development of the ASLLRP data access interface. In *Proceedings of the 5th Workshop on the Representation and Processing of Sign Languages: Interactions between Corpus and Lexicon*.

[100] Shuo Wang, Dan Guo, Wen-gang Zhou, Zheng-Jun Zha, and Meng Wang. 2018. Connectionist temporal fusion for sign language translation. In *Proceedings of the 26th ACM International Conference on Multimedia*. 1483–1491.

[101] Xin Wang, Wenhu Chen, Jiawei Wu, Yuan-Fang Wang, and William Yang Wang. 2018. Video captioning via hierarchical reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.

[102] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. 2016. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*.

[103] Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* 8, 3-4 (1992), 229–256.

[104] James Woodward. 1993. The relationship of sign language varieties in India, Pakistan, & Nepal. *Sign Lang. Stud.* 78, 1 (1993), 15–22.

[105] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang,

Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv:1609.08144. Retrieved from http://arxiv.org/abs/1609.08144.

[106] Yuancheng Ye, Yingli Tian, Matt Huenerfauth, Jingya Liu, Nataniel Ruiz, Eunji Chong, James M. Rehg, Sveinn Palsson, Eirikur Agustsson, Radu Timofte, et al. 2018. Recognizing american sign language gestures from within continuous videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2064–2073.

[107] Kayo Yin. 2020. Sign language translation with transformers. arXiv:2004.00588. Retrieved from https://arxiv.org/abs/2004.00588.

[108] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. 2016. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*.

[109] Youngjae Yu, Hyungjin Ko, Jongwook Choi, and Gunhee Kim. 2017. End-to-end concept word detection for video captioning, retrieval, and question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*.

[110] T. Yuan, S. Sah, T. Ananthanarayana, C. Zhang, A. Bhat, S. Gandhi, and R. Ptucha. 2019. Large scale sign language interpretation. In *Proceedings of the 14th IEEE International Conference on Automatic Face Gesture Recognition (FG'19)*. 1–5.

[111] Z. Zhang, J. Pu, L. Zhuang, W. Zhou, and H. Li. 2019. Continuous sign language recognition via reinforcement learning. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'19)*. IEEE, 285–289.

[112] Zhihao Zhang, Junfu Pu, Liansheng Zhuang, Wengang Zhou, and Houqiang Li. 2019. Continuous sign language recognition via reinforcement learning. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'19)*. IEEE, 285–289.

[113] Luowei Zhou, Yingbo Zhou, Jason J. Corso, Richard Socher, and Caiming Xiong. 2018. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.

Received December 2020; revised June 2021; accepted July 2021