Effectiveness of multi-scaling keypoints for Sign Language Translation

Kah Han Chia
Department of Computer Science
Univeristy of Bath
Bath, BA2 7AY
Year 2023/24
khc73@bath.ac.uk

***Abstract*—Sign language primarily uses visual cues to communicate. However, existing sign language translation models tend to neglect facial features or rely on outdated pose estimation models. In this paper, we propose a modern sign language translation model that is capable of generating facial features using a modern pose estimation method, DWPose. This paper also discusses the additional complexity facial features bring and the potential solution to address the set of problems. To enhance the model's capability in generating sign language poses, various preprocessing techniques are experimented with. These experiments mainly focus on applying different scaling variations to the pose keypoints. By scaling keypoints differently, it affects the loss function contribution of all the keypoints. In a way, encourages the translation model to focus on learning certain features more than others. Through the experimentation, the impact of different scaling methods on the generated keypoint output is assessed. The translation model used in this work is based on Progressive Transformer.**

Code: https://github.com/KahHan19/Diss

1. Introduction

Sign language is perhaps one of the greatest inventions in human history that has a significant impact on the lives of the deaf community. It provides the deaf community with an effective means of communication, enabling deaf individuals to express themselves, socialise, pursue learning opportunities and more. The World Health Organisation projected that there will be 700 million people with some sort of hearing disability by 2050 [1]. As the population of deaf communities grows, the need for sign language will also become more vital. Previous research demonstrates that using language translation is an effective way to learn new languages [2], this idea likely extends to Sign Language Translation as well. The current Sign Language translation models are able to translate spoken words into sequences of skeleton poses, however, most existing models are only able to generate these poses without facial features [3] [4]. As some sign languages require facial expressions to convey meaning, only generating skeletal key-points and neglecting facial information would not be practical for every sign language. The Figure 1 below showcases an example demonstrating the importance of word-mouthing in Sing-Language:

Figure 1: The German sign(DGS) for Brother(a) and Sister(b) are identical with a difference in lip movement and mouthing [5]

Previous work such as [6] proposed a translation model that generates skeletal keypoints with facial features using OpenPose [7], a pose estimation model to generate skeletal keypoints. The research found that the introduction of facial keypoints worsens the generated skeletal poses as a whole and they suspect that this may be due to varying number of keypoints from different body parts. Though the suspicion might make sense, another potential reason for this issue may come from the location and movement of facial features.

Figure 2: Example of Skeleton Pose.

Sign language translation models, like many other machine learning models, are trained using a loss function to optimise their performance. The loss function measures the

difference between the model's prediction and the ground truth labels [8] [9]. As body movements are more exaggerated compared to, for example, mouth movements, an incorrectly generated body keypoint will tend to have much more influence on the loss function compared to an incorrectly predicted facial keypoint. Another factor that motivated this suspicion is the density of facial keypoints. As illustrated in Figure 2 above, Facial keypoints tend to group densely together. Therefore, keypoints in a still face will not deviate as much from the ground truth keypoints, relative to the deviation of body keypoints. To address this suspicion, this paper explores the idea of scaling up facial keypoints. By scaling up facial keypoints, the facial keypoints to exaggerates their motion and it also forces the facial keypoints to be further apart. The translation model with normal body key-points and the scaled-up facial key-points was trained to cause the facial key-points to have a higher influence on the loss function, encouraging the model to learn facial key-points. After training, the facial key-points were scaled back down to observe the result.

The translation model used in this paper is based on the Progressive Transformer [3], Progressive transformers are state-of-the-art (SOTA) sign language translation models that translate spoken text into a sequence of skeleton poses. On the main work OpenPose was utilised as the Pose Estimator. The downside of the original progressive transformer model is that the generated skeleton poses does not include facial features. For the main aim of this paper, we extend upon the progressive transformer model to construct a Gloss-to-Test sign language translation model that is capable of generating sign language poses with facial keypoints using a modern pose estimation approach known as DWPose. As DWPose is able to generate poses in the OpenPose Skeletal format, it allows us to essentially mimic the same problem from the previous work [6] to experiment on the new proposed suspicions. Another motivation behind using DWPose is its ability to generate better quality keypoints and capture body keypoints that are hidden, which provides the model with a better quality input to learn from and improve upon. This is not to be confused with other translation models that utilise Convolutions Neural Network(CNN) [10], as CNN-based models generate images instead of skeletal pose.

2. Background

Sign Language (SL) is an expressive form of communication that requires multiple dimensions to convey meanings. These dimensions include facial expression, hand gestures, body and head movements. [11]. SL is not related to its spoken-language counterpart. It is an independent language with its own grammar, linguistic rules and it lacks written form [12] [13]. This mean that the translation of spoken language to sign language cannot be done directly, it is not a word to word translation, which is further challenged by the lack of its written form.

Figure 3: Glosses example in comparison to its sign and spoken language.

Researchers typically utilise gloss annotations to represent the written form of sign language. Glosses contain symbols and words from the spoken language to describe the multi-dimensional form of communication in a similar linguistic structure to SL. This allows the translation to be more direct and efficient. Figure 3 showcases an example of gloss corresponding to its spoken-language counterpart. Glosses are textual identifiers for individual signs, but they do not perfectly represent the meaning of each sign [14]. Although glosses do provide a more direct translation, they do come with its own set of flaws. Glosses are not easily accessible [15], may not capture the full sentimental meaning of its spoken language counter part and can be hard to understand without context. They are also not particularly necessary for Sign Language Translation, as demonstrated by studies such as [16] [17] , where substantial results can be achieved without them.

2.1. Sign Language Translation

Sign Language Translation(SLT) involves the translation between text and its corresponding visual sign representation. This can be done in both directions; SL poses to spoken language and spoken language to SL poses. With regard to the nature of this paper, we will be more focused on the text-to-visual sign representation aspect. There are typically two methods to perform such translation, Text-To-Pose(T2P) and Text-to-Gloss-to-Pose(T2G2P). T2G2P involves two translation models, translation from text to gloss and gloss to pose. Poses in this sense are the keypoint coordinates of the image. This allows a pose to be represented in a sequence of values which can be used as a translation target. When dealing with a small dataset, it is more beneficial to first translate text to gloss. As the linguistic rules for sign language and its spoken counterpart differ, it causes the mapping of text to pose translation to be a more complex task. The mapping from text to gloss is much simpler, as compared to skeletal keypoints which may sum up to a total of 100+ coordinate values. By using a two-step approach for translation, both translation stages can optimise more appropriately. Text to gloss can be optimised to generate a compact gloss representation which is sequenced, parallel to the individual sign poses and Gloss to pose can be optimised to translate gloss to keypoint poses.

2.2. Pose Estimation

Pose Estimation involves the extraction of human key-point poses within an image or video. The key-points represent coordinates of human body joint that can be plotted out for a visual representation of a pose in 2D and 3D space, as shown in figure 4. This estimated pose is utilised in many applications such as behavioural analysis, gesture recognition and human-computer interaction.

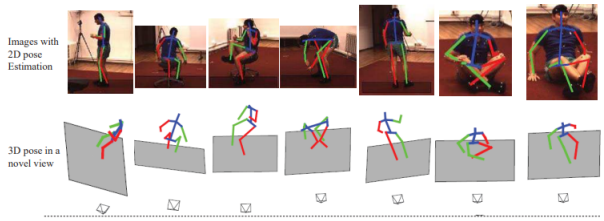

Figure 4: Image by [18], showing the difference between 2D and 3D Pose estimation.

2D pose estimation estimates the location of joints in 2D space, where coordinates of each point are represented in X and Y, whereas 3D estimation transforms a 2D image into a 3D space by introducing the Z-axis. 3D estimation is useful in the case since it has some ability to track joints hidden behind other joints. At a high level, there are generally two approaches to pose estimation [19], a Top-Down approach [20] and a Bottom-up [21] approach. A Top-down approach first detects for humans within the image using object detection, then performs pose estimation on individual bounding boxes independently. A Bottom-up approach directly performs body parts within the image and groups the body parts together into full-body poses. Since the technology for computers to visualise images is still premature, at least not to a level of how humans perceive images, these pose coordinate values are useful for the model to learn from. Figure 5 showcases an example of how both approach is done.

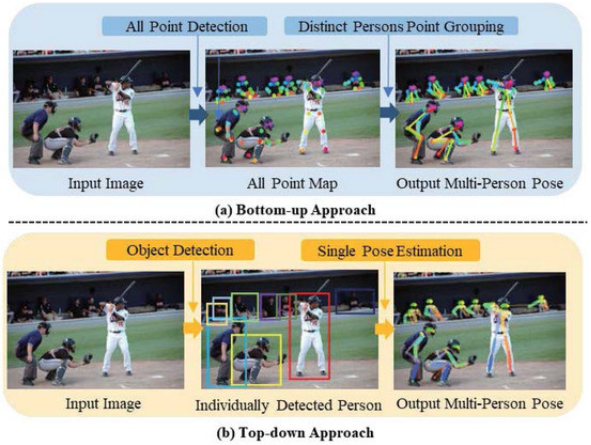

Figure 5: Comparison between Top-down and Bottom up Approach to Pose Estimation, image from [22]

2.3. Transformer

Transformers are one of the core architectures that significantly improved the field of language translation. It utilises an encoder-decoder architecture combined with an attention mechanism introduced in the "Attention Is All You Need" paper by Vaswani et al [23]. The attention mechanism allows each word in the sequence to be context-aware of its global position within the sequence. By being context-aware, the model is able to capture global dependencies effectively.

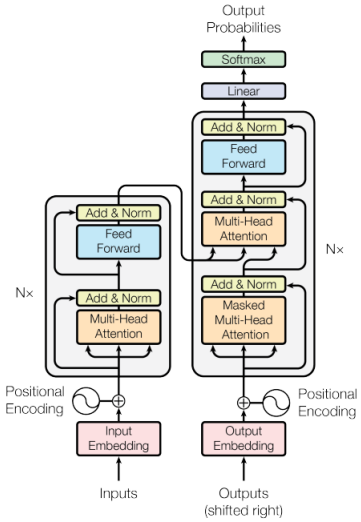

Figure 6: Transformer architecture, left is Encoder block and right is Decoder block [23]

2.3.1. Embedding layer and Positional Encoding. The embedding layer is responsible for mapping token IDs of our input sentence into a word embedding. This can be done by combining our input vector with an embedding matrix. The embedding matrix is initialised randomly at the beginning and learned during the training process. The positional encoder allows each word to carry some information about its current position in the input sequence. This is achieved using the sine and cosine functions of different frequencies depending on the position of the word in the positional embedding.

$$\begin{aligned} \mathrm{PE}(pos, 2i) &= \sin\left(\frac{pos}{10000^{\frac{2i}{\mathrm{dmodel}}}}\right) \\ \mathrm{PE}(pos, 2i+1) &= \cos\left(\frac{pos}{10000^{\frac{2i}{\mathrm{dmodel}}}}\right) \end{aligned} \tag{1}$$

The pos in the expression represents the position of the word in the sentence and i represents the position within the positional embedding vector and $dmodel$ represent the dimension of our vector. Then both the embedding vectors and the positional vector will be combined resulting a vector that is positional aware.

2.3.2. Attention Mechanism. The output of our positional encoder block will be used as an input for our attention mechanism, by using the positional aware matrix, the attention mechanism allows the model to relate words to each other and capture the long-range dependencies between the words in the input sequence. For each input words the mode computes three different vector representations, Query(Q), Key(K) and Value(V) vector.

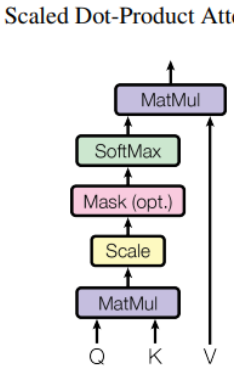

Figure 7: Scale Dot Product Attention

We utilise each of the vector and apply Scaled Dot Product attention to it as shown in Figure7 above, which essentially represents the Self Attention Equation:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{\mathbf{Q} \cdot \mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \qquad (2)$$

The product of Q and $K^\top$ results in an attention matrix, where each value represent the attention weight between Q and K. The attention essentially represents how much attention a word pays to every other words in a given sentence. Each attention weight is then normalised by value $\sqrt{d_k}$, the dimension of key vector which minimises the variance of our vector to provide stability in the attention mechanism. The normalised attention also prevents a feature from dominating other features, therefore allowing the model an opportunity to explore. We then apply softmax to convert the vector into a probability distribution function and multiply it by the value vector, as the value vector provides the information of the sentences to the model.

2.3.3. Multi-Head Attention. Instead of computing a single Q,K and V vector for each input sequence, attention mechanism utilises multiple sets of Q,K and V vectors for each attention head. Each attention head apply the same input sequence with its independent weights. This allows different heads to capture different features and relations within the input sequence. We then concatenate each attention head and apply a learnable transformation matrix to transform our result to the desired matrix size.

2.3.4. Layer Normalisation and Residual Connection. The Normalisation layer computes the mean and standard deviation across the feature dimension of the input matrix and normalises the values to provide stability for training. Residual Connection adds the output of the previous layer into the current, which allows the model to learn the parameters for each sub-layer more effectively during back propagation and it also mitigates the risk of vanishing gradient.

2.3.5. Encoder Block. The encoder block consist of two sub-layers, Multi-Head Attention Layer and Feed Forward Layer. The encoder first combines the input with the positional information derived from the positional encoding layer. This allows the model to learn the relative position of each tokens in the sequence. The position aware matrix will then be fed into the multi-head attention layer to generate the attention matrix. Finally the attention matrix is passed through a feed-forward layer, which allows the model to learn the non-linear relationships of the attention matrix. Between each sub-layer, there is a normalisation layer to stabilise the training process and a residual connection. This residual connection is done, so that the can learn the parameters for the lower level layers. As shown in Figure 6.

2.3.6. Decoder Block. The decoder block consists of a total of three sub-layers: a Masked Multi-Head Attention layer, a regular Multi-Head Attention layer, and a single Feed Forward layer. The decoder block first combine the target sequence with the positional information derived from the positional encoding layer. The positional aware matrix will then be fed into a Masked Multi-Head Attention. When a Multi-Head Attention is masked, the text in the current position is not allowed to get context from feature text. This is to ensure that the decoder can only attend to words that has been generated. Masking can be done by applying a triangular attention mask of negative infinity values to the attention scores, To demonstrate a simple masking:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \oplus \begin{bmatrix} -\infty & -\infty & -\infty \\ 0 & -\infty & -\infty \\ 0 & 0 & -\infty \end{bmatrix} = \begin{bmatrix} -\infty & -\infty & -\infty \\ 4 & -\infty & -\infty \\ 7 & 8 & -\infty \end{bmatrix} \qquad (3)$$

The generated masked matrix is then fed into the next Multi-Head attention layer, where it is combined with the output from the encoder block. This allows the model to attend to both outputs. Finally the attention matrix through a feed-forward layer. Similarly to the encoder block, at the end of each sub-layer. There consist of normalisation layer and residual connection.

2.3.7. Linear and Softmax. The Linear Layer received the output from the decoder block and transform the matrix into a higher-dimensional space to capture the complex relationship and apply softmax to produce and probability distribution of our output.

2.4. Dataset

RWTH-PHOENIX Weather 2014-T [24] is the main dataset used for this paper. This dataset contains images of SL signs and their corresponding gloss and text, which allows sign language recognition and translation tasks to be performed, making it the benchmark dataset for most SL translation models. This original dataset is an extension of the original PHOENIX-14T Dataset [25], which contains a large video-based corpus of German Sign Language called Deutsche Gebrdensprache (DGS) and it was recorded over a period of three years (2009 - 2011) from the German public tv-station. The dataset contains a total of 835,356 sign poses with a 210 by 260-pixel frame each, 2887 German words and 1066 unique glosses. While the dataset is considered a benchmark dataset, it includes blurred images. Pose estimators, such as OpenPose, may encounter difficulties in accurately estimating the skeleton poses when processing these blurred images.

3. Related Works

This section provides a detailed run through of the translation model and pose estimation method used for this paper. We also included other pose estimation methods to provide some insights on the overall performance and quality of the skeleton keypoint poses.

3.1. Pose Estimation

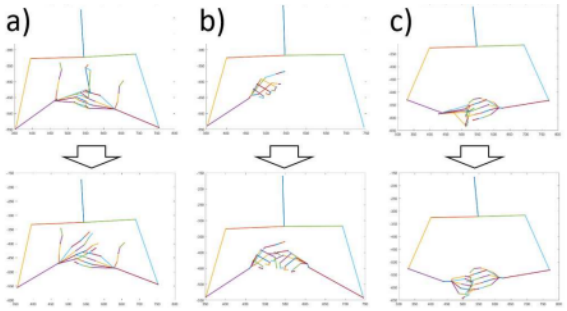

Figure 8: Example of corrections, (a,c) illogical joint and (b) capture hidden joints [26]

OpenPose [7] is an open-source pose estimation library for multi-person keypoints detection, It follows the bottom-up approach to estimate 2D pose information (e.g., body joints, facial keypoints) from images and videos. OpenPose is capable of detecting 135 keypoints, which is achieved by combining separate datasets for different body parts to create a whole body pose estimation. However, OpenPose has an incredibly large model size making it unsuitable for real-time body detection. In addition to that, the estimator is also vulnerable to hidden keypoints within images and blurred body movement due to rapid movement. Hidden keypoints are common when signing as SL involves movements where one joint is hidden behind another. One solution in this problem is to lift the 2D image into a 3D space and correct any illogical joint points using an iterative inverse kinematics [26] approach, which is shown in Figure 8.

Mediapipe Pose [27] is a top-down pose estimation method developed by Google, it is able to capture up to 33 human body pose keypoints in a 3D space. These models are optimised for mobile usage making it extremely lightweight, hence it has the ability to capture 3D poses in real-time. However, Mediapipe Pose is only able to detect up to 33 keypoints. Which is not sufficient to capture the necessary details to generate a practical sign language pose. Another example of a real-time pose estimator is Mediapipe Holistic [28], Mediapipe Holistic is an extension from Mediapipe Pose. It is able to estimate up to 33 body keypoints, 468 face keypoints and 42 hand keypoints, 21 keypoints per hand.

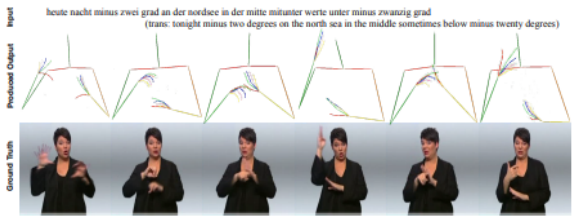

Figure 9: Examples of produced sign pose sequences [3]

On the original Progressive Transformer [3], OpenPose is utilised with Iterative inverse kinematic to generate its skeleton poses, which achieved amazing results as shown in Figure9. The results generated showcase near precise body coordinates showing huge resemblance between actual image an generated results. However, the skeleton image lack much details, as sign language relies on facial expression, hand gestures and body movement to convey meaning.

3.2. Progressive Transformer

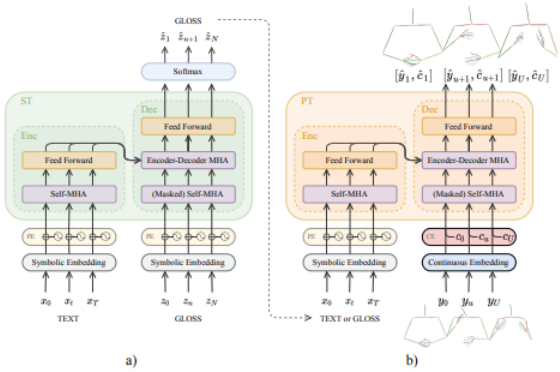

Figure 10: Progressive Transformer Architecture [3]

The main transformer used in this paper is based on Progressive transformer(PT) [3], Progressive transformer utilises two transformer networks as shown in figure 10.

One performs Text-to-Gloss(T2G) and the other is Gloss-to-Pose(G2P). As our project is only interested in generating the skeleton poses for the experiment, we are only utilising the gloss-to-pose translation block. To summarise, given some gloss input, we want the transformer to generate a sequence of skeleton keypoints poses. Progressive Transformer, unlike the original transformer model, is able to generate a sequence of continuous streams of keypoint poses. In order to generate continuous poses, a few adjustments were made to the original transformer architecture. The decoder embedding layer is first replaced with a linear embedding layer, this is to allow the model to receive continuous keypoints poses as input. To clarify, the standard approach for this task is to input skeleton keypoints frame by frame. By continuous stream of keypoints poses, means that keypoint poses are concatenated together and inputted into our translation model. Instead of learning word by word, the progressive transformer functions continuously. The Softmax at the end of the decoder block is also removed to switch the task from a classification task to a regression task.

3.2.1. Pose Estimation Tools. In the original Progressive Transformer Paper, OpenPose was utilised with Iterative inverse kinematics to generate its skeleton poses. The specific version of the OpenPose used in the paper consists of 3 facial keypoints, 5 body keypoints and 21 keypoints for each hand. This adds up to a total of 50 keypoints, as the keypoints are in 2D, we end up with a total of 100 coordinate values for X and Y coordinates. Then iterative inverse kinematics is used to lift the 2D keypoints to 3D, resulting in a total of 150 coordinates value for X, Y and Z coordinates. Referring to the original code [29] [30] [31] [30] from Saunders et al, the keypoints were scaled down by a third which allowed the training process for the skeleton keypoints to be more stable. The extracted and normalised poses will be sequenced in a line, with each line representing the different gloss sentences. Then preprocessed keypoints is passed into the counting embedding layer.

3.2.2. Counter Embedding. The counter-embedding layer receives the sequence of skeleton poses and adds a counter-embedding at the end of each keypoint frame. The counter value ranges from 0 to 1, with each value acting as a timestamp for our continuous sequence. As the input for the decoder block is no longer discrete, the End of Sequence, Start of Sequence and Padding Token which typically exist in transformer data are removed. Instead, a counter value is added to indicate 0 for the initial frame and 1 for the end of the sequence. After the counter embedding layer, the data is ready to be processed by the transformer model.

3.2.3. Output. Once the model is trained, the generated prediction should be a sequence of concatenated 150 keypoint poses with its corresponding counter value. The counter value and the z coordinates is then removed to convert the keypoints back to 2D. Lastly, the scaled the coordinate value is scaled back up down to its original size and plotted out.

3.3. Knowledge Distillation

Knowledge distillation (KD) is a concept in Machine Learning, where knowledge is transferred from a larger and more complex model (commonly referred to as the teacher model) to a smaller model and less complex model, commonly referred to as student [32]. The original concept that knowledge distillation is based on is called logit-based distillation. Logits refers to the raw output generated from the final layer of the network, logit-based distillation transfer the logits from the teacher model to student model using temperature based softmax function. Whereas, knowledge distillation transfer the learned parameters, activation form the intermediate encoder layer of the teacher model and information about the relationship between neurons within the teacher model into the student model. The knowledge allows the student model to achieve results similar to the teacher model while being more efficient. This can also be done internally, where the model utilises regularisation techniques to transfer knowledge within a model itself to improve its own performance. This is known as self-knowledge distillation (SKD).

3.4. DWPose

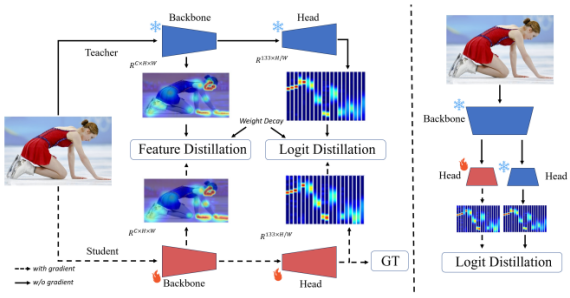

Figure 11: DWPose Pipeline of Two-Stages Distillation. On the left the first-stage distillation distills from logit and feature level. On the right, second-stage distillation distills from within the model itself(Self Distillation)
[33]

DWPose [33] is a pose estimation method proposed by Yang et al., the method explores effective knowledge distillation (KD) strategies to improve the efficiency and quality of pose estimation. As shown in Figure 11, the pipeline consists of two distillation stages.

3.4.1. First Stage Distillation. The first stage of distillation utilises the knowledge distillation method, the teacher model used in DWPose is RTMPose [34], RTMPose is a State of the art (SOTA) pose estimation model that is capable of performing high quality and real-time pose estimation. In this distillation stage, the student learns from the teacher's logit and features. These logits and features are learned directly from the encoder block of the teacher model, also commonly refereed to as the backbone. The knowledge is learned by minimising the Mean Square Error Loss Function for the

student and teacher features, the loss function calculates the difference between the student's feature F^s and the teacher's feature F^t. The total loss function is a summation of the loss function for the teacher model and the loss function for both the feature and logits of the student and teacher model. The feature distillation lost function can be formulated as:

$$L_{\text{fea}} = CHW \times \prod_{c=1}^{C} \prod_{h=1}^{H} \prod_{w=1}^{W} (F_t^{c,h,w} - f(F_s^{c,h,w}))^2 \quad (4)$$

C,H,W denotes the Channel, Height and Width of the teacher's feature and f denotes a function that reshapes F^s to match F^t. The equation for the loss function for the original classification of RTMPose is as follows:

$$L_{\text{ori}} = -\sum_{n=1}^{N} \sum_{K=1}^{K} W_{n,k} \cdot \sum_{i=1}^{L} \frac{1}{L} \cdot V_i \log(S_i) \quad (5)$$

Where $W_{n,k}$ is the target weight mask, this is important to capture hidden keypoints. L is the length of the x and y localisation bins and V_i is the label value. The distillation loss for logits L_{logits} is similar to L_{ori} without the target weight mask and finally the total Loss used to train student can be formulated as:

$$L_{\text{Total}} = L_{\text{ori}} + \alpha L_{\text{logits}} + \beta L_{\text{fea}} \quad (6)$$

Here α, β are the hyper-parameters to scale the loss, which will be tuned during training. It can also be tuned manually depending on the task, if it requires a more feature focus model or logit focus.

3.4.2. Second Stage Distillation. In the second stage, the student model from the previous distillation stage will be utilised as its own teacher model. The architecture first build a replica of the second-stage distillation teacher model but with an untrained decoder block (head). This means that the student will have the same encoder block (backbone) as the teacher model, hence the head of both models will be frozen as the hyper-parameter has been tuned in the previous distillation stage. Then the learnt feature from the shared backbone is passed into the student's untrained head and the teachers' head, which is also frozen to keep the parameters from the previous step, the backbones will generate the Logits from both student and teacher model to calculate the logit loss, this logit loss is then scaled with a hyper-parameter γ. The trained losses will the be used to train the head of the student model, resulting in a model capable of performing human pose estimations. The final loss for second-stage distillation can be formulated as:

$$L_{\text{stage\_2}} = \gamma L_{\text{logits}} \quad (7)$$

4. Methods

Figure 12: Skeleton Pose result of DWPose and OpenPose [33]

The Pose estimation method utilised in this paper is DWPose, as DWPose is shown to generate higher quality keypoints and is capable of capturing hidden keypoints, As shown in Figure 12. As for the Translation model, this paper utilised Saunders Progressive Transformer(PT), more specifically the G2P transformer block that is responsible for translating glosses to pose.

4.1. Data-Preprocessing and Training changes

For PT, the generated skeleton keypoints of the dataset are normalised by a division of 3 and fed into a counting embedding layer. As this paper uses a different Pose estimation method, changes are made to the preprocessing for the keypoint dataset in both the normalisation process and the counting embedding layer. Compared to OpenPose, the value generated by DWPose is significantly higher in terms of magnitude. As there exists a large magnitude difference, the scaling for the DWPose coordinate would also need to be adjusted. On top of that, the entire transformer model has also been adjusted to process keypoint sequences as DWPose consists of 234 coordinate values and the original PT only contains 150 coordinate values. The counting embedding layer used in this paper is built entirely from scratch as the counting embedding used in the original paper could not be located. After going through the original preprocessed Saunders sample dataset, it was found that the actual dataset slightly differs from the paper. It was claimed that the counter values range between [0-1], with 0 representing the first sequence and 1 representing the end of the sequence, but in the sample preprocessed keypoints attached in the original code, the final counting embedding for each sentence is found to be:

$$C_{\text{final}} = \frac{C_{\text{total}} - 1}{C_{\text{total}}} \quad (8)$$

The C_{final} represents the final counting value and C_{total} is the total frames for given gloss sentece. Therefore, the code is formulated to start at 0 for the first frame and Eq. 8 for the final frame. Finally, the code to display skeleton poses in the PT code was also reaplaced to display DWPose instead.

5. Experiment

The goal of the evaluation is to observe how a change in the scale-down ratio affects the results of the generated

image. The paper utilised the two commonly utilised evaluation methods in the Field of Pose Estimation, Which are Simple Loss Function, Dynamic Time Warping(DTW) and Percentage of Correct Keypoints [35].

The main Loss function used in our model is Mean Square Error(MSE), MSE is commonly utilised to evaluate the performance of a regression model and it measures the mean squared difference between the predicted values in this case our predicted keypoints and the ground truth values. This can be formulated as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{9}$$

Where n represent the number of data points in the dataset, y_i represent the target output and $\hat{y}_i$ represents the predicted output. Adaptive Moment Estimation(Adam) was also used as our main optimiser, this is mainly to stay consistent with the original work. Adam effectively scales the learning rate for each parameter based on the initial and second moments of gradient, this allows the model to start with a higher learning rate allowing the model to learn quicker during the initial stages of training and the optimiser will reduce the learning rate as training proceeds, which help prevents overshooting to allow model to converge. For this paper, the minimum learning rate is set to be 0.0002 to prevent learning rate from becoming too small and start with an initial learning rate of 0.003. An early stopping was also implemented so that the model will end the training when the model does not observe any improvement on the DTW value within 7 epochs.

The Percentage of Correct Keypoints(PCK) score measures the percentage accuracy of the predicted pose across all the poses in the datasets. This is done by calculating the Euclidean distance between the Ground Truth keypoints and the predicted keypoints across the entire dataset. If the distance between two points is lower than a predefined threshold, then the predicted keypoints would be considered as correctly predicted. The PCK score is then calculated from the percentage of correctly predicted keypoints out of the total keypoints in the dataset. A higher PCK score indicates a closer resemblance between the predicted and ground truth poses, which suggests a better performance.

Dynamic Time Warping(DTW) is a technique used to measure the similarity between two sequences of data points that are indexed in time order, these sequences are normally referred as Two Time Series Data and may vary in terms of length. The similarity is useful to align the different data sequences by warping its time indices, this is useful for our keypoints data as it preserves the time order introduced in the counting embedding layer. To compute the similarity between the sequences, the distance matrix containing pairwise distances between the keypoints of the predicted and ground truth sequences is first constructed. Then the warping path that minimises the aggregate distance between the sequences is located. Using the warping path and distance matrices, the algorithm sum the distance matrices along the warping path to calculate the DTW distance. A smaller DTW distance represents a greater similarity between the sequences and a greater similarity results in a better matching pose sequence. DTW allow us to effectively compare sequences of different lengths, In this case, the sequences represent the skeleton keypoints of individual frames, which is important for evaluating the performance of our continuously generated output.

This paper performed three different sets of experiments. Firstly, the translation model is trained with different equal scaling values. This experiment first preprocess each keypoints in the skeleton pose with the same scaling value, the objective of this experiment is to observe how much scaling is required for DWPose to be trained effectively. The second experiment preprocesses each skeleton keypoint with multiple scaling values, this experiment will scale facial keypoints differently from body keypoints. The objective of this experiment is to test the effectiveness of multi-scaling and whether. Lastly, this paper will also experiment with training the sign language translation model with facial keypoints only. The main experiments is to evaluate our DWPose-compatible Progressive Transformer model.

5.1. Evaluation

5.1.1. Quality of Skeletal Keypoints. The goal of this paper is to utilise a sign language translation model that is capable of generating additional keypoint features, As this work is based on the Saunder PT. The results of the DWPose skeleton poses will be used to compare to the OpenPose skeleton poses to demonstrate the feature and visual differences between the two models.

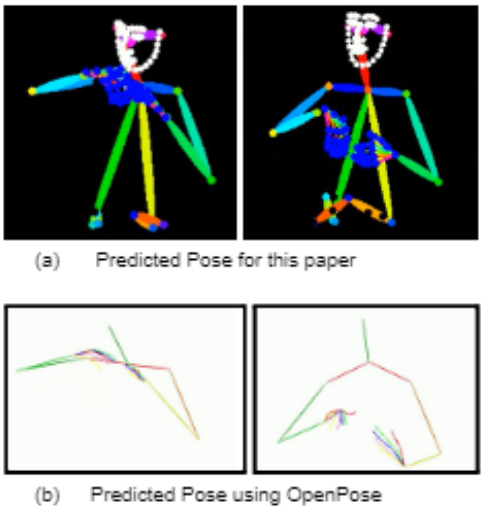

Figure 13: Skeleton Pose result of DWPose approach(a) and original approach(b)
[33]

Figure 13 above presents the visual differences between both methods of sign language generation. Although the original approach(OpenPose) is successful in capturing a hand gesture and body shoulder movements. Our implementation with DWPose successfully produced an image with

complete facial and body features. The total 2D keypoint value of the original PT contains 50 X, Y coordinate pairs, whereas compared to the DWPose approach, it captured a total of 134 X, Y coordinate pairs with 67 pairs dedicated to facial keypoints.

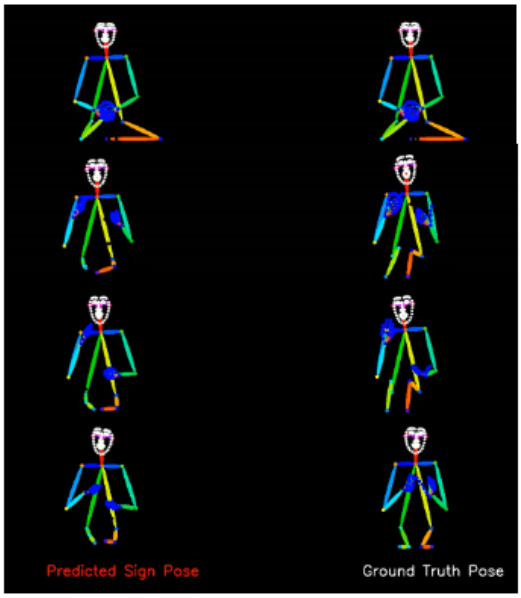

Figure 15: Model 100-100(Pose100) Generated and Ground Truth Skeleton comparison for the gloss "wetter wie aussehen morgen"

5.1.2. Equal Scaling Results. First denote our different models by "poseX", where "X" represents the scaling value of the entire skeleton pose. Two different scaling values was tested , 25 and 100. As demonstrated in Figure 14, the pose100 results converged much quicker and performed much better in every aspect of the test results. This demonstrates that a higher scale value allows for better overall performance. On the side note, the model training with any scaling was abandoned as the loss function for every step is significantly higher compared to other models and it took too long for the model to converge. Therefore, it was deemed not practical for model training. The skeleton pose for our better performing model pose 100 will also be shown further down the paper in Figure 15.

5.1.3. Multi-Scaling Results. We first denote our different models by "x-y", where "x" represents how much the body keypoints are scaled down and "y" represents how much the facial features are scaled down. In this paper, a constant parameter value of 100 was used to scale down the body keypoints and we manipulate the "y" variable to observe and analyse the impact of different y scaling parameters on the model's performance. The justification behind the value 100 stems results from our equal scaling model. The baseline model is represented by 100-100, this indicate that the whole body keypoints are equally scaled down and an extreme case was also test, at which the facial keypoints is not scaled down at all, this is useful to force the model to learn facial keypoints. Lastly, 100-25 and 100-50 was experimented to test the effective of different scaling ratio. The results of the experiment are illustrated at Figure 17 and 18. The three main comparison done in this paper to evaluate the effect of the parameter are the Loss Functions, DTW and PCK.

5.1.4. Loss Function Evaluation for Multi-Scaling. By varying the y scaling parameter, the influence of the facial keypoints on the loss function changes. When the facial keypoints are scaled down more (lower "y" value), their influence on the loss function increases relative to the body keypoints. Conversely, when the facial keypoints are scaled down less (higher "y" value), their influence decreases. This is supported by the result in Figure 18 and 17. As the scaling parameter is reduced, the loss for each model increases, with 100-25 having the highest loss. This means that the model will focus more on the facial keypoints. As shown in 18, the mean loss of each model increases as the scale down parameter decreases. The assumption made for this paper is that, if the model becomes more facial influenced, the facial keypoint generation will improve. However, this is not the case. Comparing the DTW, the best performing model is the equally normalised keypoints and it worsen as scale down parameter decreases. In-fact, by observing the extreme case in Figure 16 it is shown that the DTW eventually worsens over time when the model are fully focused on generating facial keypoints.

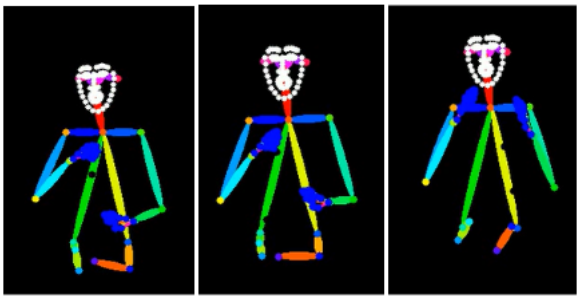

Figure 20: "O" shape mouthing

5.1.5. PCK Evaluation for Multi-Scaling. The standard measure for PCK is not as informative when it comes to whole-body translation for PT architecture. As shown in Figure 18, the PCK value improves as the weight for facial keypoints increases. The expected outcome for an improved PCK is that the model starts to learn the minimal movement for the facial keypoints movement, as expect the percentage of correct keypoints to increase. To evaluate PCK we observe the overall best-performing model for PCK, 100-0 which has a mean value of 0.969 and a maximum of 0.981. By generating the final trained model of our 100-0 model, the skeleton poses start to overfit onto a "O" shape mouth, as shown in Figure 20 and 17. Each of the skeleton poses is extracted from different gloss sentences, the generated results are as expected, where each skeleton managed to

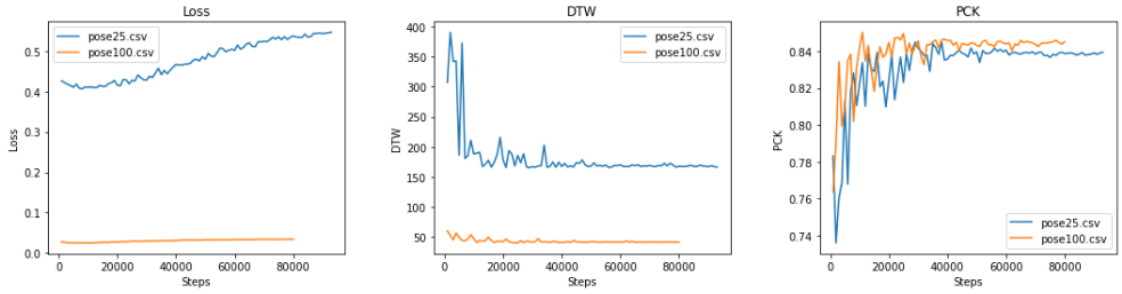

Figure 14: Loss, DTW and PCK over steps for model with equal Keypoints Scaling, "poseX", where "X" represents the scaling value of the entire skeleton pose.

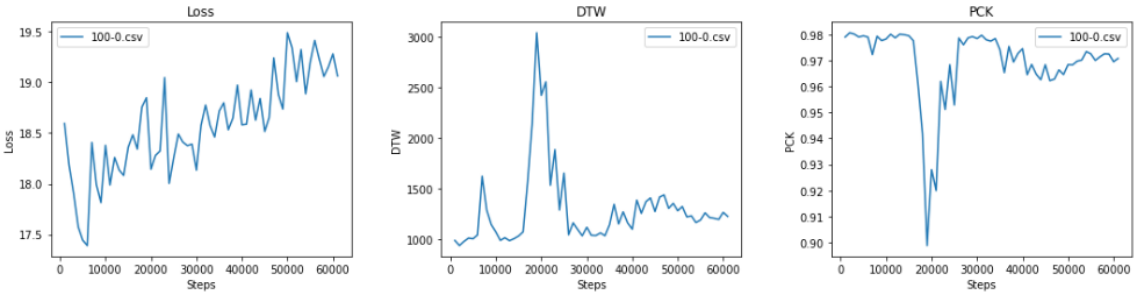

Figure 16: Extreme Case of Keypoints Scaling, Body Keypoints Divided by 100 and Facial Keypoints is not normalised.

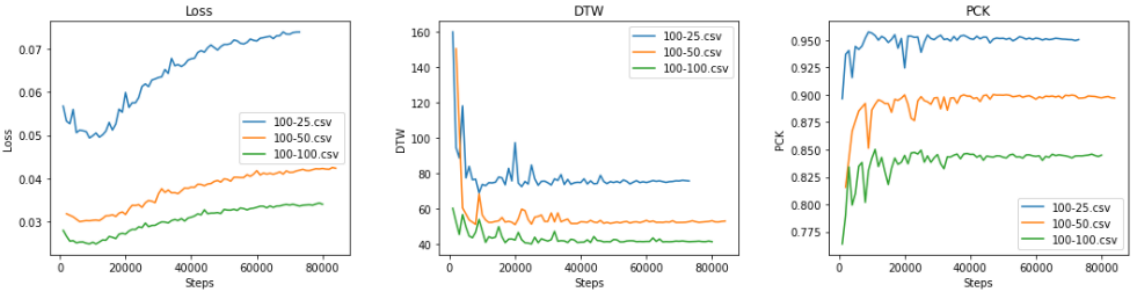

Figure 17: Loss, DTW and PCK over steps for model with Different Keypoints Scaling, "100-50", represent that the body keypoints is divided by 100 and 50 represent that the facial keypoints is divided by 50.

capture the initial body pose and seem to freeze on it. This is due to the model's lack of focus on learning the body's keypoints. Each output essentially pauses on the initial body pose and have an "O" mouth shape, with rare occasions of closed mouth. The suspected reason for this phenomenon is due to the size of the dataset for this paper. As this paper utilises a dataset that contains up to only 1066 unique glosses and 7096 sentences, it is possible that there is a lack of mouth movement across different unique glosses. The "O" shaped mouth may be a common mouthing for many glosses, causing the attention matrix to have evenly distributed attention for unique keypoints. Which causes the model to struggle in distinguishing between different mouthing based on the glosses input. This leads to the highest PCK model being the least practical when compared to the other models.

Figure 15 illustrates the model that performed the least effectively in terms of PCK. While the image showcases a certain degree of success in capturing body and head movement. Although the orientation is not perfect, the model learned to perform subtle head tilts and occasion of mouth movement. The model mainly struggles in generating tightly compact keypoints like facial and finger keypoints, but it is much more practical to use in comparison to the extreme case model and it also provides more feature information compared to the original PT work.

5.1.6. DTW Evaluation for Multi-Scaling. As shown in Figure 18, the lowest PCK model 100-100 also has the lowest DTW value. This remains true across all models, even for those with higher PCK values. As DTW measures the similarity between two sequences of data points indexed in time order, it is prefered for the DTW value to be small. This essentially goes against the evaluation of PCK,

	Loss				DTW				PCK			
	100-25	100-50	100-0	100-100	100-25	100-50	100-0	100-100	100-25	100-50	100-0	100-100
Mean	0.064	0.037	18.572	0.030	78.210	54.746	1294.076	43.270	0.949	0.894	0.969	0.839
Std. Deviation	0.008	0.004	0.479	0.003	11.693	10.894	390.984	3.558	0.009	0.012	0.015	0.014
Minimum	0.049	0.029	17.391	0.025	69.097	51.141	934.961	40.249	0.896	0.816	0.899	0.764
Maximum	0.074	0.043	19.489	0.034	160.079	150.556	3040.196	60.353	0.957	0.902	0.981	0.850

Figure 18: Basic Statistics for different scaling

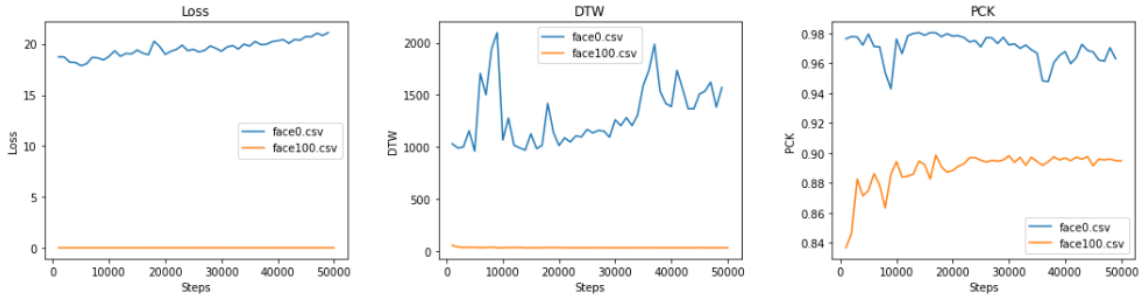

Figure 19: Loss, DTW and PCK over steps for model with equal only facial, ”faceX”, where ”X” represents the scaling value of the facial features.

as higher PCK indicates a better performing model, the resultant High DTW argues otherwise.

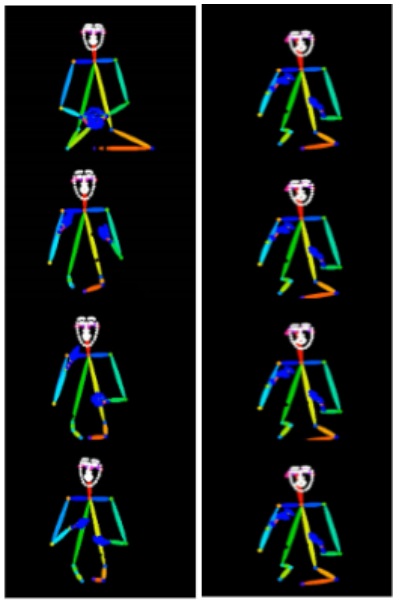

Figure 21: Generated Skeleton comparison for the gloss ”wetter wie aussehen morgen”, (a) Lowest DTW and Lowest PCK Model 100-100, (b) Highest PCK and Highest DTW Model 100-0

As illustrated in Figure 21 two models was compared for generating the same gloss sentence. For the model that generated the results with the highest DTW, there is a significant difference in the time order sequencing between the ground truth and generated result. As the data was not sequenced properly, Model 100-0 only generated a single image across the sequence, inversely, The generated pose from the Lowest DTW model successfully learns the sequences of data in the appropriate time order. Therefore, the generated result looks to be more active and more according to the ground truth pose.

Figure 22: ”O” shaped mouthing problem first face only model for two different gloss sentences

5.1.7. Face only Results. The different models expereimented in this section are denoted by ”faceX”, where ”X” represents the scaling value of the facial keypoints. The point of this experiment is to train a face focus, by removing body keypoints the complexity of mapping from gloss to pose will be reduced. In terms of Loss, DTW and PCK, the normalised facial feature performed much better as well.

But the ”O” shape mouthing persists, as shown in Figure22. Further justifying the argument above. However, by training a face-focus model. The model seems to learn the overall face movement better, but it still struggles with smaller features like the eye, mouth and nose.

6. Conclusion and Future Work

This paper successfully implemented DWPose within the progressive transformer, we highlighted the importance of facial features in sign language translation and extended the architecture of the progressive transformer to generate skeleton poses with facial features using DWPose. DWpose has brought forward many improvements to other pose estimation methods. As demonstrated in Figure 13. By improving sign language skeleton pose generation, we can improve sign language translation as a whole. This will no doubt be a useful stepping stone to helping communities in need and an inspired individual who wishes to learn sign language.

To summarise our discovery, dividing each keypoints by 100 resulted in the best skeletal keypoints. Although, scaling the keypoints differently did result in a change of loss function. Models with different scaling to encourage learning of facial keypoints did not perform as expected. The generated poses for the multi-scaled model seem to overfit to an ”O” shape mouthing. Two potential reasons for this issue are the limited dataset size and the frequent usage of an ”O” shape mouth in most sign poses. If the ”O” shape mouth is used for a large margin of the glosses in the dataset, it causes the attention for the mouthing to be more evenly distributed in the attention matrix. This pattern is consistent with our face-only model. By training our model with face glosses only, the model seems to improve in terms of head movement. However, the ”O” shape-mouthing issue still persists. For future work, it may be worth to run the proposed model with a larger dataset. More result should also be done to sign language data set to understand the properties of facial features in sign language better.

Code: https://github.com/KahHan19/Diss Word Count: 6612

References

[1] World Health Organization, “Deafness and Hearing Loss,” https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss, 2024, accessed: April 21, 2024.

[2] D. Benelhadj Djelloul and B. A. Neddar, “The usefulness of translation in foreign language teaching: Teachers’ attitudes and perceptions,” *AWEJ for translation & Literacy Studies Volume*, vol. 1, 2017.

[3] B. Saunders, N. C. Camgoz, and R. Bowden, “Progressive transformers for end-to-end sign language production,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*. Springer, 2020, pp. 687–705.

[4] X. Giró-i Nieto, “Can everybody sign now? exploring sign language video generation from 2d poses.”

[5] U. Von Agris, M. Knorr, and K.-F. Kraiss, “The significance of facial features for automatic sign language recognition,” in *2008 8th IEEE international conference on automatic face & gesture recognition*. IEEE, 2008, pp. 1–6.

[6] S.-K. Ko, C. J. Kim, H. Jung, and C. Cho, “Neural sign language translation based on human keypoint estimation,” *Applied sciences*, vol. 9, no. 13, p. 2683, 2019.

[7] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, “Realtime multi-person 2d pose estimation using part affinity fields,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7291–7299.

[8] T. Ananthanarayana, P. Srivastava, A. Chintha, A. Santha, B. Landy, J. Panaro, A. Webster, N. Kotecha, S. Sah, T. Sarchet *et al.*, “Deep learning methods for sign language translation,” *ACM Transactions on Accessible Computing (TACCESS)*, vol. 14, no. 4, pp. 1–30, 2021.

[9] Y. Gu, C. Zheng, M. Todoh, and F. Zha, “American sign language translation using wearable inertial and electromyography sensors for tracking hand movements and facial expressions,” *Frontiers in Neuroscience*, vol. 16, p. 962141, 2022.

[10] J. Zheng, Y. Chen, C. Wu, X. Shi, and S. M. Kamal, “Enhancing neural sign language translation by highlighting the facial expression information,” *Neurocomputing*, vol. 464, pp. 462–472, 2021.

[11] Z. Liang, H. Li, and J. Chai, “Sign language translation: A survey of approaches and techniques,” *Electronics*, vol. 12, no. 12, p. 2678, 2023.

[12] N. S. Khan, A. Abid, and K. Abid, “A novel natural language processing (nlp)–based machine translation model for english to pakistan sign language translation,” *Cognitive Computation*, vol. 12, pp. 748–765, 2020.

[13] W. Eźlakowski, “Grammar of polish sign language as compared to grammar of polish language,” *Sign Language Studies*, vol. 20, no. 3, pp. 518–532, 2020.

[14] M. Müller, Z. Jiang, A. Moryossef, A. Rios, and S. Ebling, “Considerations for meaningful sign language machine translation based on glosses,” *arXiv preprint arXiv:2211.15464*, 2022.

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

[16] B. Saunders, N. C. Camgoz, and R. Bowden, “Everybody sign now: Translating spoken language to photo realistic sign language video,” *arXiv preprint arXiv:2011.09846*, 2020.

[17] S. Stoll, A. Mustafa, and J.-Y. Guillemaut, “There and back again: 3d sign language generation from text using back-translation,” in *2022 International Conference on 3D Vision (3DV)*. IEEE, 2022, pp. 187–196.

[18] C.-H. Chen and D. Ramanan, “3d human pose estimation = 2d pose estimation + matching,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[19] H. Chen, R. Feng, S. Wu, H. Xu, F. Zhou, and Z. Liu, “2d human pose estimation: A survey,” *Multimedia Systems*, vol. 29, no. 5, pp. 3115–3138, 2023.

[20] T. D. Nguyen and M. Kresovic, “A survey of top-down approaches for human pose estimation,” *arXiv preprint arXiv:2202.02656*, 2022.

[21] J. Li, W. Su, and Z. Wang, “Simple pose: Rethinking and improving a bottom-up approach for multi-person pose estimation,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 11 354–11 361.

[22] C. Park, H. S. Lee, W. J. Kim, H. B. Bae, J. Lee, and S. Lee, “An efficient approach using knowledge distillation methods to stabilize performance in a lightweight top-down posture estimation network,” *Sensors*, vol. 21, no. 22, p. 7640, 2021.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[24] N. C. Camgoz, S. Hadfield, O. Koller, H. Ney, and R. Bowden, “Neural sign language translation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7784–7793.

[25] J. Forster, C. Schmidt, O. Koller, M. Bellgardt, and H. Ney, "Extensions of the sign language recognition and translation corpus rwth-phoenix-weather." in *LREC*, 2014, pp. 1911–1916.

[26] J. Zelinka and J. Kanis, "Neural sign language synthesis: Words are our glosses," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 3395–3403.

[27] Google, "MediaPipe: Pose landmarker," https://developers.google.com/mediapipe/solutions/vision/pose_landmarker/, Accessed: 2024-04-23.

[28] ——, "MediaPipe: Holistic," https://github.com/google/mediapipe/blob/master/docs/solutions/holistic.md, Accessed: 2024-04-23.

[29] B. Saunders, "Progressive transformers for speech language processing," https://github.com/BenSaunders27/ProgressiveTransformersSLP/tree/master, 2024.

[30] B. Saunders, N. C. Camgoz, and R. Bowden, "Adversarial Training for Multi-Channel Sign Language Production," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2020.

[31] ——, "Continuous 3D Multi-Channel Sign Language Production via Progressive Transformers and Mixture Density Networks," in *International Journal of Computer Vision (IJCV)*, 2021.

[32] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[33] Z. Yang, A. Zeng, C. Yuan, and Y. Li, "Effective whole-body pose estimation with two-stages distillation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4210–4220.

[34] T. Jiang, P. Lu, L. Zhang, N. Ma, R. Han, C. Lyu, Y. Li, and K. Chen, "Rtmpose: Real-time multi-person pose estimation based on mmpose," *arXiv preprint arXiv:2303.07399*, 2023.

[35] Y. Yang and D. Ramanan, "Articulated human detection with flexible mixtures of parts," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 12, pp. 2878–2890, 2012.