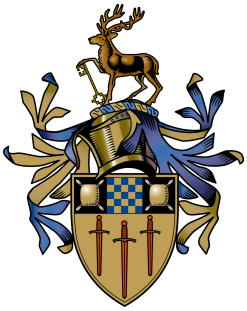
Representations for Sign Language Production

Harry Walsh

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey



Centre for Vision, Speech and Signal Processing
Faculty of Engineering and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

September 2024

Abstract

Sign languages are complex languages with their own grammatical structure and vocabulary. As a result, communication between the Deaf and hearing communities can be challenging, requiring translation rather than a simple word-to-sign substitution, typically performed by an expert human interpreter or translator. However, the demand for interpreters exceeds the supply, with only an estimated 1 interpreter for every 100 British Sign Language user in the UK, leading to a lack of access to information for the Deaf community. This disparity has motivated the development of automatic translation systems for sign language.

For decades, researchers have sought to develop automatic sign language translation systems, aiming to bridge communication gaps between Deaf and hearing communities. However, translating between spoken language text and continuous sign language video, presents significant challenges due to the multi-channel nature of sign language and limited annotated data. Research has focused on the translation of signed to spoken language, known as Sign Language Translation (SLT), with the reverse task, Sign Language Production (SLP) being based on more deterministic graphics based approaches. Early SLP systems, relying on avatar-based approaches and hand-crafted rules, often produced unnatural signing. While more recent advances in deep learning have improved the naturalness of productions, they still fall short of human expressiveness. This thesis addresses this gap by exploring representations for sign, that can aid in translating spoken language to photo-realistic sign language videos.

Historically, the SLP task has been broken into three steps: First, an initial translation from a spoken sentence to a gloss sequence, followed by a skeleton production, which is finally used to drive a photo-realistic signer or avatar. This thesis begins by investigating two representations that can be used in the first step of the translation pipeline, namely, gloss and the Hamburg Notation System (HamNoSys). The contribution also explores the influence of tokenisation, the process of segmenting a sequence into discrete units, which further alters the representation. We also leverage the recent development in pre-trained language models, like BERT and Word2Vec, to generate improved word and sentence-level embeddings. The findings demonstrate the effectiveness of the proposed techniques and help to set a baseline for the following contributions of this thesis.

The second contribution introduces a novel approach to Text-to-Gloss (T2G) translation, called Select and Reorder (S&R). This approach breaks down the translation process into two distinct steps: Gloss Selection (GS) and Gloss Reordering (GR). The GS model is responsible for selecting the most appropriate glosses for a given spoken language sentence, while the GR model reorders the glosses to create a more natural sign language sequence. This disentanglement of tasks allows each model to specialise in a specific aspect of translation. For this task, we create two new representations: Spoken Language Order (SPO) gloss and Sign Language Order (SIO) text. To build these representations, we once again leverage the power of pre-trained language models to build an alignment between the text and gloss. Through this, we show significant improvement in our metrics while also reducing the computational cost of the translation pipeline.

Previous works have suffered from the problem of regression to the mean, leading to under-articulated and incomprehensible signing. Moving to the next stage of the pipeline, the third contribution of the thesis uses dictionary examples and a learned codebook of facial expressions

to create expressive sign language sequences. However, simply concatenating signs and adding a face creates robotic and unnatural sequences. To address this, we present a novel 7-step approach to stitch signs together. First, by normalising each sign into a canonical pose, cropping, and stitching, we create a continuous sequence. Then, by applying filtering in the frequency domain and resampling each sign, we create cohesive natural sequences that mimic the prosody found in the original data. We leverage a SignGAN model to map the output to a photo-realistic signer and present a complete SLP pipeline.

Finally, the thesis explores the creation of a data-driven representation of sign language as a substitute for resource-intensive linguistic annotations. This involves employing a vector quantisation technique to learn a lexicon of motions that can be assembled to generate a natural and meaningful sign language sequence. The lexicon can be directly mapped to a sequence of skeleton poses, allowing the translation to be performed by a single network. Furthermore, by leveraging the limited linguistic annotations, we can enhance the representation by applying additional pressure on the model during the training process. The development of a data-driven representation offers a potential solution to overcome the reliance on linguistic annotation. The results show that the learned representation is a suitable replacement for gloss, outperforming previous methods.

The thesis concludes with a summary of the findings and proposes future work to further enhance the quality of SLP systems.

Key words: Sign Language Production, Natural language processing, Computer Vision, Data Representation, Sign Language Generation

Email: harry.walsh@surrey.ac.uk

WWW: https://www.surrey.ac.uk/people/harry-walsh

Acknowledgements

My early education was not easy, but thanks to a few amazing teachers, I was able to find topics and subjects that I was passionate about. This led me to study Engineering at University, where I was fortunate enough to meet some amazing people.

I would like to thank my supervisor, Prof. Richard Bowden, for his guidance and support throughout my PhD, and even before, as if not for encouragement during my undergraduate degree, I would not have pursued a PhD. As I did not know if I was capable of such a task. I also want to thank my co-supervisor, Dr. Ben Saunders, for his invaluable advice and support throughout. I wish him luck in his new endeavour.

I want to thank my friends and family for their unwavering support and encouragement. A special thanks to Cian Raychauduri, Kyle Sobanjan, Ethan Sampson, and Catherine Myburgh, who have always supported me. I would also like to thank my parents, who have been a constant source of encouragement throughout my PhD journey.

Finally, I would like to thank everyone in the Cognative Vision lab and CVSSP for providing me with support and entertainment. I am grateful for the chance to be a part of such a supportive academic community.

Declaration

The work presented in this thesis is also presented in the following manuscripts:

Chapter 3 Harry Walsh, Ben Saunders, and Richard Bowden. Changing the representation: Examining language representation for neural sign language production. In *LREC 2022 Workshop Language Resources and Evaluation Conference 24 June 2022*, page 117, 2022.

Chapter 4 Harry Walsh, Ben Saunders, and Richard Bowden. Select and reorder: A novel approach for neural sign language production. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 14531–14542, 2024.

Chapter 5 Harry Walsh, Ben Saunders, and Richard Bowden. Sign stitching: A novel approach to sign language production. In *The 35th British Machine Vision Conference (BMVC)*, 2024.

Chapter 6 Harry Walsh, Abolfazl Ravanshad, Mariam Rahmani, and Richard Bowden. A data-driven representation for sign language production. In *Proceedings of the 18th International Conference on Automatic Face and Gesture Recognition (FG 2024)*. Institute of Electrical and Electronics Engineers (IEEE), 2024.

The following publication is where I was the first author and the context is related to this thesis, however, it has been excluded as it "did not fit the story":

- Harry Walsh, Ozge Mercanoglu Sincan, Ben Saunders, and Richard Bowden. Gloss alignment using word embeddings. In *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, pages 1–5. IEEE, 2023
- Harry Walsh, Ed Fish, Ozge Mercanoglu Sincan, Mohamed Ilyes Lakhal, Richard Bowden, Neil Fox, Kearsy Cormier, Bencie Woll, Kepeng Wu, Zecheng Li, Weichao Zhao, Haodong Wang, Wengang Zhou, Houqiang Li, Shengeng Tang, Jiayi He, Xu Wang, Ruobei Zhang, Yaxiong Wang, Lechao Cheng, Meryem Tasyurek, Tugce Kiziltepe, and Hacer Yalim Keles. Slrtp2025 sign language production challenge: Methodology, results, and future work. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025
- Low Jian He, Harry Walsh, Ozge Mercanoglu Sincan, and Richard Bowden. Hands-on: Segmenting individual signs from continuous sequences. In *Proceedings of the 18th International Conference on Automatic Face and Gesture Recognition (FG 2024)*. Institute of Electrical and Electronics Engineers (IEEE), 2024
- Harry Walsh, Maksym Ivashechkin, and Richard Bowden. Using sign language production as data augmentation to enhance sign language translation. In *Adjunct Proceedings of the 25th ACM International Conference on Intelligent Virtual Agents*, IVA Adjunct '25, New York, NY, USA, 2025. Association for Computing Machinery

Contents

Nomenclature

ASL American Sign Language

BTG Bracketing Transduction Grammar

BPE Byte-Pair Encoding

BSL British Sign Language

BSLCPT BSL Corpus **T**

CNN Convolutional Neural Network

CSLR Continuous Sign Language Recognition

CTC Connectionist Temporal Classification

DGS German Sign Language - Deutsche Gebärdensprache

DTW Dynamic Time Warping

DTW-MJE Dynamic Time Warping Mean Joint Error

fps frames per second

chrF Character n-gram F-score

GAN Generative Adversarial Network

GRU Gated Recurrent Unit

G2P Gloss-to-Pose

G2T Gloss-to-Text

GS Gloss Selection

GR Gloss Reordering

GT ground truth

HOH Hard of Hearing

HamNoSys Hamburg Notation System

LSTM Long Short-Term Memory

MHA Multi-Headed Attention

MoCap Motion Capture

MDN Mixture Density Network

MeineDGS MeineDGS Annotated

MT Machine Translation

NMT Neural Machine Translation

NLP Natural Language Processing

NAR Non-AutoRegressive

NSVQ Noise Substitution Vector Quantisation

PHOENIX14T RWTH-PHOENIX-Weather-2014**T**

POS Part Of Speech

PT Progressive Transformer

P2S Pose-to-Sign

RNN Recurrent Neural Network

VQ-VAE Vector Quantised Variational Autoencoders

VQ Vector Quantisation

VAE Variational Autoencoders

SLR Sign Language Recognition

SLT Sign Language Translation

SLP Sign Language Production

SMT Statistical Machine Translation

S&R Select and Reorder

S2T Sign-to-Text

S2G2T Sign-to-Gloss-to-Text

SIO Sign Language Order

SPO Spoken Language Order

T2P Text-to-Pose

T2G Text-to-Gloss

T2G++ Text-to-Gloss++

T2H Text-to-HamNoSys

T2G2H Text-to-Gloss-to-HamNoSys

T2G2P Text-to-Gloss-to-Pose

T2S Text-to-Sign

T2SPOG Text to Spoken Language Order Gloss

Symbols

Introduced in Chapter 1

X	Spoken language sentence with W words
W	The number of words in a spoken language sentence
x_i	The i^{th} word in a spoken language sentence
Y	Gloss sequence with G glosses
G	The number of glosses in a sequence
y_i	The i^{th} gloss in a sequence of glosses
H	Sequence of Hamburg Notation System (HamNoSys) with S symbols
S	The number of symbols in a sequence of HamNoSys
h_i	The i^{th} symbol in a sequence of HamNoSys
$\hat{Y}$	Predicted gloss sequence
Y^*	Target gloss sequence
L_{Cross}	Translation loss
L_{Hand}	Handshape Predictions Loss
L_{Total}	Total Training loss
E	Embedding dimension width
HS	Sequence of handshape symbols
λ_{H}	Handshape loss weight
X_{WP}	Spoken language sentence WordPiece tokenization
$\mathbf{X}_{\text{BERT}}$	A spoken sentence embedded with a BERT model
$\mathbf{X}_{\text{W2V}}$	A spoken sentence embedded with a Word2Vec
$\mathbf{X}_{\text{ave}}$	Contextual information from Spoken language sentence
λ_{x}	Embedding scaling factor
$[CLS]$	A special token used for sentence summarization

Introduced in Chapter 2

X^{SIO}	Spoken language sentence in sign language order
Y^{SPO}	Gloss sequence in spoken language order
A	One-to-one alignment of spoken words and glosses
M	Mapping from spoken language to sign language order
$\mathbf{Y}_{\text{W2V}}$	Gloss sequence embedded with word2vec
$\mathbf{Y}_{\text{BERT}}$	Gloss sequence embedded with BERT
A_{W2V}	Alignment from word2vec
A_{BERT}	Alignment from BERT
α	Alignment filter value

Introduced in Chapter 3

U	Number of frames in a video
V	Sequence of video frames
v_i	The i^{th} frame in a video sequence
P	Sequence of poses
P^{Δ}	Distance travel over a sequence of poses
p_u	The u^{th} pose in a sequence
α_{crop}	Pose crop threshold
$\mathcal{Z}$	The z-transformer of a pose sequence
P_{stitched}	Stitched pose sequence
P^H	Normalised sequence of handshapes
P^H_{stitch}	Normalised Stitched sequence of handshapes
D	Duration of a sign
d_i	The i^{th} duration of a sign
C	Cutoff frequency
ω_{c}	Angular cutoff frequency
O	Butter-worth filter order
$\mathcal{F}^{\text{t}}$	Sequence of face tokens
f_i^t	The i^{th} face token
F_i	The i^{th} facial expression in the dictionary
$\mathbf{F}^z$	Embedded sequence of face meshes

DF	Dictionary of facial expressions
N_f	Number of facial expressions in a dictionary
U_{f}	Number of frames in a facial expression
DS	Dictionary of signs
DS^{y}	Sign dictionary gloss tags
N_s	Number of signs in a dictionary
S_i	The i^{th} sign in a dictionary
U_{s}	Number of poses in a sign
a_i	The i^{th} pose in a sign
y_{q}	Gloss query
j_{sub}	Substitute sign index
λ_{y}	Translation loss weight
λ_{f}	Face loss weight
λ_{d}	Duration loss weight
λ_{C}	Cutoff loss weight
λ_{CN}	Counter loss weight
c	Counter for the number of frames
c_u^*	The u^{th} ground truth counter
$\hat{c_u}$	The u^{th} predicted counter
y_i^*	The i^{th} ground truth gloss in a sequence
$\hat{y}_i$	The i^{th} predicted gloss in a sequence
f_i^{t*}	The i^{th} ground truth face token
$\hat{f}_i^t$	The i^{th} predicted face token
f_i^*	The i^{th} ground truth face mesh
$\hat{f}_i$	The i^{th} predicted face mesh
d_i^*	The i^{th} ground truth duration
$\hat{d}_i$	The i^{th} predicted duration
C^*	Ground truth cutoff frequency
$\hat{C}$	Predicted cutoff frequency
J	Number of joint in a skeleton in euclidian space
J_a	Number of joints in a angular skeleton representation
$\mathcal{D}$	Number of dimensions

A^{DTW}	Dynamic Time Warping (DTW) path between two pose sequences
$\mathcal{T}$	Tracking pose keypoint
Δ	Distance travelled between two points
U_{stitch}	Number of frames used to stitch two signs
U_{Min}	Minimum number of frames to stitch two signs
V_{max}	Maximum velocity
V_{min}	Minimum velocity

Introduced in Chapter 4

U_{cb}	Number of frames in a codebook
CB	Spatial-temporal codebook of signs
N	The number of entries in a codebook
$\mathbf{t}_{\mathbf{i}}^{\mathbf{z}}$	A codebook embedding
i	The i^{th} index
$\mathbf{z}$	Embedded features
$\hat{\mathbf{z}}$	Predicted embedding
u	The u^{th} index in a list
L_{Code}	Codebook loss
L_{Con}	Contrastive loss
λ_{Con}	Contrastive loss weight
$\mathbf{V}$	A normally distributed noise source
γ	Codebook dead token threshold
β	Codebook active token threshold
p_u^*	The u^{th} ground truth pose
$\hat{p_u}$	The u^{th} predicted pose
$\mathcal{A}$	Contrastive loss anchor samples
$\mathcal{B}$	Contrastive loss negative samples
τ	Scalar temperature parameter
T	A sequence of tokens
t_n	The n^{th} token in a sequence
$\mathcal{M}$	Number of tokens in a sequence

List of Figures

List of Tables

Chapter 1

Introduction

Sign languages are complex visual languages used by the Deaf and hard-of-hearing communities [172]. They are often misunderstood as simple gestures or mere one-to-one transcriptions of spoken languages. However, in reality, this could not be further from the truth. The world's 300+ sign languages possess their own complex nuances, unique grammatical structure and extensive vocabulary [37, 168]. The Deaf community has a rich history and culture, with sign languages being the primary mode of communication for many deaf individuals. Similar to spoken languages, sign language has evolved over time and can vary from region to region. They are independent of spoken languages, for instance British Sign Language (BSL) is distinctly different from American Sign Language (ASL) despite both being from English-speaking countries. As signs develop with culture and location, many exhibit iconicity, the ability to represent objects, actions, or ideas through signs that resemble the original concept [39]. For example, the sign for "HOUSE" in BSL is shown in Figure 1.1. This sign is iconic as it represents the roof and walls of a house.

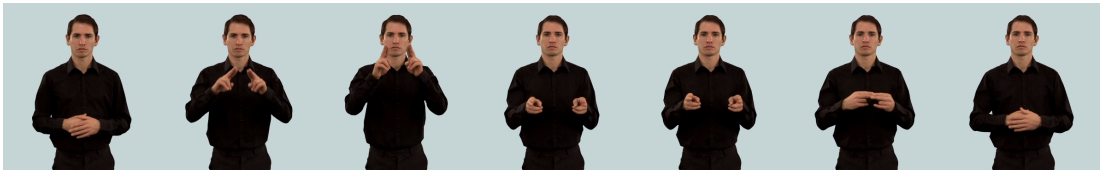


Figure 1.1: An example of the sign for "HOUSE" from the SignBank dataset [165].

Spoken languages utilise a linear sequence of sounds to convey information. In contrast, Sign languages are considered multi-channel, meaning various articulators are used simultaneously

or asynchronously to convey information [117], often grouped into two categories: manual and non-manual features. The handshape and its movement during a sign are known as the manual features. While aspects such as the head movement, the body posture (such as shoulder movements), mouthing [1], gaze [2] and facial expressions all fall into the non-manual category. In addition to these articulators, sign languages exhibit a range of unique features such as placement, co-articulation, and, fingerspelling. Placement is when a signer uses the surrounding space, known as the sign space, to convey meaning. This can be syntactic, when an object, person or place is assigned a region that can be used to refer back to it. Or, it can be topographic, when the signer uses the space to convey relational connections [185]. Such as a person's location relative to an object in a room. All these features contribute to the complexity of the language, and we provide more details in Section 2.1

Individuals with hearing loss between 71-90 dB are classified as severely deaf, with any further loss changing the classification to profound [129]. The World Health Organisation estimates that 430 million people worldwide are Deaf or Hard of Hearing (HOH) [198]. Specifically, in the UK the Royal National Institute for Deaf People (RNID) estimates that 14.2 million (or 1 in 5) can be classed as Deaf or HOH [56], and this is expected to increase over the next 10 years till 2035 to 18 million [151]. There are around 151,000 BSL users in total, of whom 87,000 are classified as Deaf [56]. This significant population faces limitations in accessing information readily available to their hearing counterparts. For individuals who have never heard a spoken language, literacy rates are significantly lower. This limited access has significant societal impacts. For example, in 2021 across the UK, 48% of deaf people were employed compared to 72% of hearing people, and only 18% attained a bachelor's degree compared to 33% within the hearing community [128]. These disparities highlight the urgent need to improve the quantity and quality of accessible information for the Deaf community. This barrier is further compounded by the global shortage of sign language interpreters, for instance, in the UK, there is 1 interpreter for every 100 BSL users. This has motivated the need for automatic sign language translation systems.

Computational sign language research has been an active area for the last 30 years [178]. This area can broadly be grouped into linguistic study and automatic translation. Automatic

[1] lip patterns

[2] the movement of the eyes

translation is seen as a tool which could help bridge the gap between the Deaf and hearing communities by improving the quantity of accessible information. As depicted in Figure 1.2, the translation can be divided into two primary tasks: 1) Sign Language Translation (SLT), the translation from signed to spoken languages, and 2) Sign Language Production (SLP), the generation of sign language videos from spoken language input. To date, a disproportionate amount of research has focused on the task of SLT [2, 25, 26, 33, 186, 204, 222, 217]. A useful tool but it is seen as more helpful for the hearing community. Whereas, SLP is considered more beneficial for the Deaf community, as it allows them to better understand and interpret the world around them.

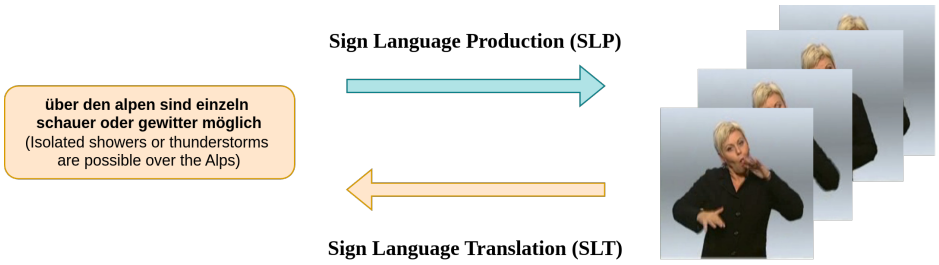


Figure 1.2: An overview of Sign Language Translation (SLT) vs. Sign Language Production (SLP).

SLP was initially tackled using statistical machine translation and avatar-based rendering approaches. However, these approaches suffered from a lack of naturalness and robotic motion as well as being limited to a small domain of discourse [8, 38, 49, 51, 224]. Since the advent of deep learning, approaches have moved to more sophisticated methods, which have both improved in terms of vocabulary and realism. But they still fail to capture all the features of this complex language. Resulting in under-articulated productions.

Early deep learning approaches broke the translation task into three stages. First, the spoken language text is translated into a gloss representation, then the gloss is translated into skeleton pose, and finally the pose is used to produce a photo-realistic signer [173, 174]. As depicted in Figure 1.3. However, these approaches suffered from 1) unnatural transitions caused by simply concatenating isolated signs together, and, 2) low-resolution output, which hinders signer comprehension. Later works, such as [153, 155, 157] have attempted to address these issues by using a single end-to-end model to directly translate from spoken language to skeleton pose.

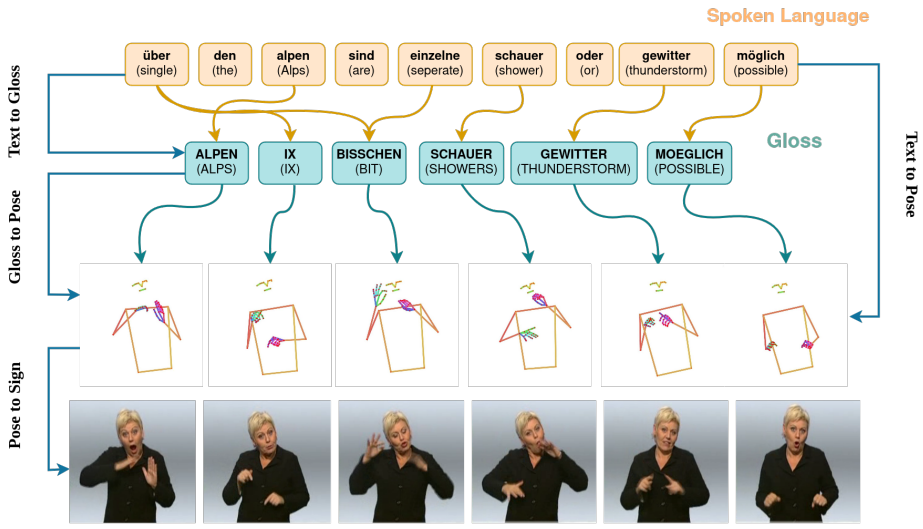


Figure 1.3: An overview of the Sign Language Production (SLP) pipeline, showing the different steps from spoken language text to Gloss intermediary to Skeleton pose to the final output photo-realistic production.

However, these approaches still suffer from regression to the mean and hallucinations [159], which limits the naturalness of the production.

To improve the quality of the translation, many approaches rely on multiple intermediary representations. Figure 1.3 (top to bottom) shows a general overview of a SLP system. It shows the skeleton pose and gloss representations that can be used to aid translation. The final output can be either a photo-realistic signer or an avatar. It is worth noting that photo-realistic signers have been shown to be more popular with the deaf community [190], and therefore, we focus on this output format.

Each of these intermediary representations has its own trade-offs, which we detail here and summers in Table 1.1. Gloss is the written word associated with each sign and can be used as a notation system for videos of sign language [172]. Gloss was originally used in the field of linguistics to aid in its study and has since been used as an intermediate representation for automatic translation. Gloss offers a person independent representation, which also collapses both spatial and temporal variants, which is common between signers. While this is useful, it also has its limitations, as it does not capture the multi-channel continuous nature of the language. In

	Gloss	HamNoSys	Skeleton Pose	Video
Expressivity	No information about the form of sign	Describes both manual and non-manual features	Joint positions of both manual and non-manual features	Represents the most natural form of expression
Scalability	Limited to several datasets	Limited to the MeineDGS Annotated (MeineDGS) dataset	Automatically extracted from video	Available for all datasets
Data Cost (MB)	237	929	6230	18366

Table 1.1: Table contrasting gloss, HamNoSys, skeleton pose and video on data cost, expressivity, and scalability. Data cost is calculated as the total size of the MeineDGS dataset in megabytes (MB).

addition to gloss, there are other linguistic notations such as HamNoSys [144] and signWriting [176], that are closer to phonetic representations of the language.

The skeleton pose [3] is a more direct representation of sign language, as it captures the spatial-temporal features, while still being appearance agnostic. It is more expressive and can capture the nuances of the language compared to gloss. But this representation is higher-dimensional and requires complex systems to automatically extract accurate keypoints from a video. Alternatively, Motion Capture (MoCap) can be used to accurately collect pose data, but this is costly and requires specialised equipment. As a result, high-quality data is limited in availability. Video recording is the most convenient and readily available source for capturing sign languages. However, it contains a large quantity of redundant information. Plus, they are costly to record, store, and transmit.

1.1 Motivation

Existing approaches utilise a combination of these features, but as discussed, each presents distinct trade-offs. Consequently, a given representation can either aid or hinder the translation process, potentially acting as a bottleneck. This raises the question: which representation is most effective for machine translation, and how can it be leveraged to enhance overall performance?

[3] a set of landmarks on key positions on the body

This thesis focuses on the production of photo-realistic sign language videos from spoken language text. Informed by the literature surveyed in Chapter 2, we identify several shortcomings of existing work. These can be summarised by the following research topics:

Enhance translation performance The majority of SLP research has predominantly focused on the later stages of the pipeline (Figure 1.3 middle to bottom). Meaning the Text-to-Gloss (T2G) task is relatively under-explored. Therefore, current systems might be bottlenecked by this step. Furthermore, there is a wide range of research from the NMT field that is yet to be applied to the SLP problem. Such as the use of pre-trained language models, data augmentation, and, additional supervision.

Alternative representations The current state-of-the-art SLP approaches rely on gloss as an intermediary representation. While gloss is useful as it provides a discrete representation of sign language, which makes it compatible with traditional sequence-to-sequence NMT architectures [73, 189]. However, gloss fails to capture many aspects of sign language, such as co-articulation, non-manual features and sign-prosody. Little work has been done to explore alternative representations to gloss in the SLP pipeline. Despite more descriptive notation systems existing such as HamNoSys [68].

Expressive and natural productions Many SLP systems attempt to regress a skeleton pose sequence from the spoken language text (Figure 1.3 right). However, these approaches have noted issues with regression to the mean [159]. Caused by deep models attempting to minimise their loss functions by predicting an average skeleton pose. This can lead to under-articulated and incomprehensible signing. Dictionary examples have previously been used to overcome these issues, but the productions are either robotic and unnatural or have an unrealistic appearance.

Gloss-free sign language production Sign language datasets which contain linguistic annotation are limited. Gloss, a commonly used annotation system, is costly to produce and requires a skilled linguist to create. Compared to spoken languages, sign languages are under-resourced and have limited parallel data. This has limited the size and the domain of discourse of many SLP systems. Therefore, to scale to unconstrained translation, some gloss-free approaches have

been proposed. Gloss-free SLT approaches have started to show competitive performance to supervised systems. But, similar improvements are yet to be seen in the SLP field.

These shortcomings provide a natural set of objectives for this thesis to address;

1. To explore alternative representations that can be used in the SLP pipeline.
2. To improve the translation performance of SLP systems.
3. To create more expressive and natural productions.
4. To reduce the need for linguistic annotation for SLP.

1.2 Contributions

Chapter 3 introduces the task of Text-to-HamNoSys (T2H) and leverages the recent developments in NMT in order to enhance translation (Targeting objectives 1 and 2). HamNoSys is a linguistic annotation system for sign language. Each character from its 200 symbol set describes different aspects of a sign, including handshape, location, motion, and non-manual expressions. In comparison, gloss fails to capture many aspects of sign language, as its vocabulary is borrowed from spoken languages. Given the detail with which HamNoSys describes the language, we propose using it as an alternative to gloss. The field of NMT has been able to leverage self-supervised techniques and large corpora of text data in order to train word embedding models. These models convert tokens into vectorised representations, which can represent the meaning of words. We leverage these models in order to create enhanced embeddings. We also explore the use of additional supervision, which, when used in combination with HamNoSys shows improved performance on a number of metrics. We demonstrate how this representation improves the translation performance by up to 5% [194].

Chapter 4 presents Select & Reorder, a novel approach that enhances translation quality [195]. As previously discussed, gloss is a useful notation for translation and is used in many approaches [158, 155, 196]. Thus, in Chapter 4 we enhance T2G translation by leveraging the large lexical overlap between spoken languages and gloss annotation schemes, therefore targeting objective number 2. Given that this process is used as the first step in a full production system, we find

that the increased translation score has a positive effect on the downstream tasks, also targeting objective 3. Traditionally, a translation system has to achieve two objectives: 1) a change in vocabulary, and, 2) a change in order. The approach breaks down the translation task into these two separate sub-tasks. Through this decomposition, we can show significant improvements in terms of BLEU-1 score, of up to 27%. To accomplish this task, we once again leverage the power of word embedding models, which have been trained on large corpora of spoken language text and therefore see millions of examples during training. Using these models we build an alignment between the spoken language and the gloss annotation scheme. Analysis of two data sets reveals that the source and target share a vocabulary of 33-35%, which is a significant overlap. This provides a number of strong anchors which we use to build accurate alignment. Using the alignment we create text in sign language order and gloss in spoken language in order to perform translation.

Chapter 5 presents a full production pipeline, named Sign Stitching [196]. Some modern approaches suffer from regression to the mean [216, 156, 153, 155], unsurprising given the limited dataset size. However, this issue was not present in older statistical based systems, given that they leverage pre-recorded dictionaries of signs. This chapter proposes applying modern deep learning approaches to a three-step translation system. Given that gloss is available across several datasets, it is used as the first representation in the pipeline. To overcome the previously discussed drawbacks of gloss, this representation is enhanced with additional information, such as the duration of each sign to be performed, a facial expression, and a filter value. This new representation is termed Gloss++. By adding the additional components, sign prosody[4] can be explicitly modelled. Using the Gloss++ representation and a dictionary of signs, we were able to create expressive sign language sequences which do not suffer from regression to the mean. Many deep learning approaches often disregard non-manual features and operate solely on a reduced set of skeleton keypoints. The proposed approach overcomes this by using Vector Quantisation (VQ) in order to learn a dictionary of facial expressions in a data-driven manner. A combination of all these techniques allows us to produce more natural and expressive outputs, thus targeting objectives 2 and 3. Finally, skeleton outputs have been shown to reduce deaf comprehension. Therefore, to complete the production pipeline we leverage SignGAN [158], an adversarial neural network that is able to generate a photo-realistic signer conditioned on the

[4]The natural rhythm that signers use to convey additional information

previous steps skeleton pose sequence. A user survey found that 87% of people preferred the productions from the stitching approach compared to previous work.

In our final contribution (Chapter 6), we target the last objective. Previous chapters relied on linguistic annotation in order to perform production. However, as discussed, annotation systems like gloss are time-consuming and costly to create. This limits the size and availability of datasets. Given that Deep learning based approaches are heavily dependent on the data there is a need to break this reliance. Existing works that achieve a strong translation score have been limited to small domains of discourse, such as the weather [25]. While large domains struggle to achieve 1/4 of the scores [101]. Once again, we applied VQ to the SLP problem. This time expanding the learnt spatial-temporal representation to encompass the full skeleton.

Through this process, we learn a data-driven representation for sign language which is capable of combining spatial and temporal variants of the same sign. Similar to how gloss is invariant to the natural variation observed between signers. The approach learns to cluster short motions based on the representation learned in the embedding space. We show that by placing additional constraints on the embedding space we can manipulate the arrangement in order to learn a better representation. This involves using linguistic annotation and a contrastive loss. But given that we aim to reduce the need for linguistic annotation, we show the ability to apply an enhanced codebook to other datasets. Meaning existing linguistic data can be applied across datasets. By controlling the vocabulary size, we can ensure that each token in the codebook remains expressive. The learnt representation is then used in a translation system, but unlike previous three-step approaches, this data-driven representation can be mapped directly back to a skeleton pose sequence. Simplifying the translation pipeline to two steps. Helping to overcome issues such as regression to the mean. Once again, we leverage the SignGAN module to generate a photo-realistic signer. In this work, we collect higher resolution training data for the model from a native deaf signer. This helps reduce motion blur and improve the realism of the final output. The metrics show the approach matches and even outperforms previous methods including the approach in Chapter 5.

1.3 Summary

Sign language is a visual language which raises the question of how best to represent it for machine translation. This thesis aims to explore this question by using a range of representations for sign language from linguistic annotation to automatic learning techniques. Through this exploration we aim to enhance SLP and create more natural and realistic output.

The following chapter aims to ground the research in this thesis by giving a comprehensive review of existing work and related topics. Chapter 3 explores HamNoSys and proposes the task of T2H. This is combined with the recent developments in NMT. Chapter 4 proposes a novel approach to T2G translation by breaking down the task into two sub problems. Chapter 5 then presents a complete SLP system capable of producing photo-realistic output, which does not suffer from the common issue of regression to the mean. The final contribution, Chapter 6 proposes learning a data-driven representation of sign language which leverages linguistic annotation to improve the overall performance.

In these four contributions, we aim to satisfy all the objectives of this thesis: exploring alternative representations, enhancing overall performance, ensuring all productions are expressive, and finally reducing the need for linguistic annotation. The final chapter summarises the work of this thesis and suggests avenues for future research.

Chapter 2

Literature Review

In this chapter, we review the literature on computational sign language research. Sign language research encompasses both computer vision and natural language processing, and therefore, we discuss aspects of both fields. We start by briefly discussing the history and linguistic features of sign, before moving on to the more recent work in Sign Language Recognition and Translation. We then discuss SLP and the challenges faced in this area. We explore the related work on Natural Language Processing (NLP) and Machine Translation (MT). Finally, we conclude the chapter by discussing the shortcomings of existing research, which motivates the contributions in the following chapters.

2.1 Sign Language

References to sign languages date back as far as the fifth century BC, when the philosopher Socrates pondered, ”If we hadn’t a voice or a tongue, and wanted to express things to one another, wouldn’t we try to make signs by moving our hands, head, and the rest of our body?” [133]. Like spoken languages, sign languages have evolved over centuries. The first recorded use of sign language in the UK was in 1576 [184], and today, an estimated 150,000 people in the UK use BSL [170].

Sign languages are used worldwide by Deaf communities; however, they are not universal. For instance, most sign languages use one hand to sign the alphabet, while only a few, such as those

used in the UK, Australia, and New Zealand, use two hands [84, 93, 165]. Even countries that share the same spoken language may have different signed languages. For example, while both the USA and the UK have English as their primary spoken language, their Deaf communities use distinct sign languages: ASL and BSL [79]. In total, there are over 300 sign languages worldwide [59], and large variation exists even within a single country. Analogous to the development of regional dialects and colloquialisms in spoken languages, sign languages also exhibit variations across regions. For example, the sign for ”red” in England differs from the lexical variant used in Northern Ireland [22].

There is a common misconception that sign languages are one-to-one transcriptions of spoken language. However, sign languages possess their own distinct grammar, syntax, and vocabulary [172]. Sign languages also have a unique structure, comprising manual and non-manual features which can be performed asynchronously. Manual features refer to the handshapes, movements, and locations used. While non-manual features encompass facial expressions, head movements, and body posture [177]. These features serve to convey crucial grammatical information, such as interrogatives, negations, and emphasis. For instance, in BSL, a raised eyebrow can signify a question, whereas a furrowed brow can denote a negative statement [142]. Moreover, signed prosody (the speed, rhythm, and intensity with which signs are executed) can also contribute to the expression of emotions and emphasis [199].

In addition to manual and non-manual features, co-articulation represents another layer of complexity in sign language. Co-articulation refers to the phenomenon where the movement of one sign influences the subsequent sign. This can lead to the blending of signs, wherein the movement of one sign seamlessly transitions into the next [67]. An example is the sign combination ”NEWSPAPER” and ”READ”, where the handshape for ”NEWSPAPER” is maintained with one hand while the other hand transitions into the sign for ”READ”. Co-articulation often occurs naturally when signing at speed, as it forms fluid transitions between signs.

Sign language can borrow words from spoken languages, in the form of fingerspelling. Fingerspelling is frequently employed to articulate proper nouns (names and places), technical terms, and words lacking a dedicated sign. Each letter of the alphabet is represented by a distinct handshape [121]. A sequence of letters can be used to spell out a word, or a single letter can be

used to represent a word. Fingerspelling is most prevalent in ASL where it constitutes anywhere between 12-35% of a discourse [136].

The aforementioned features collectively contribute to the richness and expressiveness of sign language. However, they also pose significant challenges for computational sign language research.

2.2 Sign Language Recognition & Translation

Figure 2.1 provides an overview of computational sign language translation. The field can be broadly divided into two categories: SLP and SLT, shown on either side of the diagram. SLT aims to translate from signed to spoken languages, which can be achieved by using several different intermediaries. In this section, we discuss the history of Sign Language Recognition (SLR) and SLT research, before moving on to SLP.

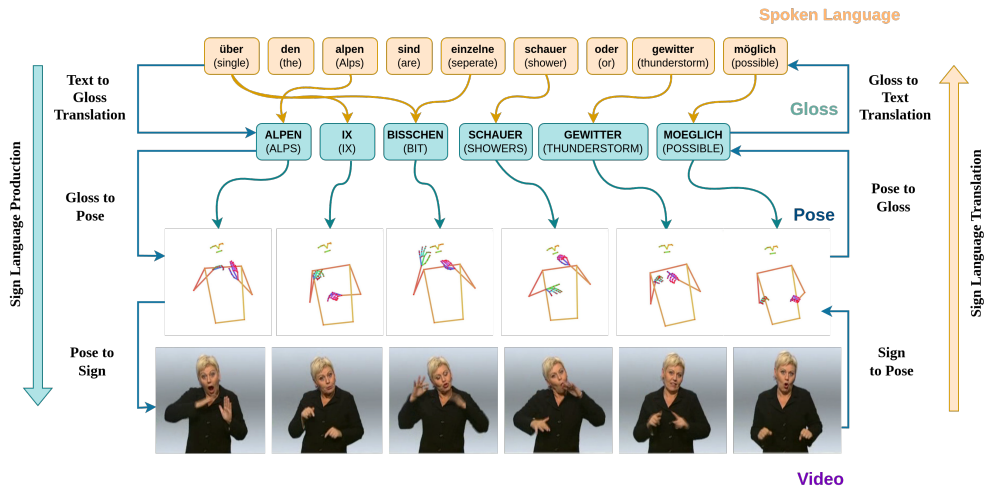


Figure 2.1: An overview of automatic sign language translation systems. Showing both Sign Language Production (SLP) and Sign Language Translation (SLT).

The ultimate goal of SLT is to facilitate communication between the Deaf and hearing communities [16], a difficult task. So, initial research focused on isolated SLR, which involves recognition of isolated individual signs. Over 40 years ago, the first rule-based method was introduced [64], utilising an electronic glove to recognise fingerspelling in ASL. Since then, various sensor-based

methods have been proposed [54, 109]. However, these methods have not gained widespread acceptance within the Deaf community [3].

The first vision-based approach was limited to a small vocabulary of 10 signs [178]. The advent of deep learning significantly improved both accuracy and vocabulary size (exceeding 2000 signs [106]). Initially, models were adapted from the action recognition literature [29, 82, 182]. Later improvements leveraged linguistic priors in the form of contrastive learning [203] and word embedding models [223].

As previously discussed, sign languages are far more complex than isolated signs, prompting the expansion of research to Continuous Sign Language Recognition (CSLR). This task involves both segmenting a video into its constituent signs and their respective classification [100]. Camgoz et al. was the first to propose end-to-end Sign-to-Text (S2T), where continuous sign language sequences are directly converted into spoken language sentences. The approach used a Convolutional Neural Network (CNN) to extract visual features and then a Recurrent Neural Network (RNN) to perform the translation to text [25]. The transformer [189] has since been applied to the problem and has demonstrated state-of-the-art results [26, 33, 217]. Camgoz et al. proposed Sign Language Transformers [26], which used a Connectionist Temporal Classification (CTC) loss on the encoder, which allowed the model to be trained end-to-end for both SLR and SLT. Note that even though a large amount of work has been done since, this approach has become the standard for computing back-translation performance for SLP [153, 155, 157].

It has been shown that translating via a gloss intermediary (Sign-to-Gloss-to-Text (S2G2T)) [25] or using gloss as a supervision yields superior performance [26]. However, due to the time and cost of glossing sign data, more recent large-scale datasets only contain spoken language translations [2]. This has led to the development of gloss-free S2T approaches. Some approaches used a sign spotter to guide the translation [164, 166], while more recent works leverage large language models such as GPT [204], mBart [222], and T5 [186].

2.3 Sign Language Production

To date, sign language translators and interpreters have been the primary method of communication between the Deaf and hearing communities. However, the number of qualified sign language interpreters is limited, with only 1,500 registered in the UK [131]. This corresponds to 1 interpreter for every 100 BSL users, this shortage has motivated the development of computational SLP systems. SLP is the reverse task to SLT, which aims to translate spoken sentences into continuous sign language.

Prior to its current direction, the field's focus was on graphical avatar-based approaches [8, 38]. The avatar's motion was driven with either MoCap[61] or parameterised glosses[38]. Parameterised glosses emerged as a popular approach, utilised in projects such as ViSiCAST [8], Tessa [38], Dicta-sign [49], WebSign [51], and, Esign [224]. These systems used pre-existing annotation schemes, such as HamNoSys [144] or SignML [92] as an intermediate representation. However, animating from these representations requires further manual annotation in terms of speed, timing and non-manual information. A time-consuming process to curate. Moreover, these approaches typically rendered signs in isolation, leading to unnatural transitions between them. The use of MoCap data has allowed for more realistic motion. However, motion capture is also costly to create and requires specialised hardware. This limits the scalability. Later non-manual features, such as mouthing and head movements, were added to avatar-based systems [47, 48]. But still the system fell into the 'uncanny valley' resulting in users feeling uncomfortable [122].

These early approaches proved to be unpopular with the Deaf community [97], primarily due to their cartoon like appearance and robotic motion [147]. With some users even saying the system is more appropriate for children than adults [202]. Furthermore, these early methods relied on a rule based approach, which required handmade rules to perform translation. Tessa, for example worked by taking a spoken language sentence as input and looked for "legal" phrases that could then be looked up from a sign dictionary [38]. This resulted in a system that was not generalizable to new signs and often lacked grammatical structure [75].

As a result, the field progressed to using Statistical Machine Translation (SMT), where the rules can automatically be learned from data [23, 88, 135]. This allowed the system to be automatically updated once new data became available, reducing the need for manual intervention. In addition, SMT enabled the model to leverage context to solve for ambiguities in the text [91]. However,

these approaches required handcrafted features to be extracted before being used in an ensemble of models [102]. This requires domain knowledge to correctly extract the features.

The introduction of deep learning has allowed for more data driven approaches that can learn a direct mapping between spoken and sign languages. The first deep SLP pipeline broke down the task into three steps, first, a translation from T2G followed by a Gloss-to-Pose (G2P) look-up, and finally a Pose-to-Sign (P2S) video generation step [173, 174]. By using a look-up table in step two of the pipeline, the model is unable to blend the motion between signs, creating unnatural transitions. Zelinka et al. [216] proposed an approach that directly synthesises skeleton pose from the spoken language text. For each word in the source sentence, the model predicts a fixed 7-frame skeleton pose sequence. This ignores the grammatical structure of the language and assumes a fixed length for each sign. Xiao et al. [210] proposed jointly learning SLR and SLP in a single model using an RNN. This approach used a simplified skeleton representation, which only contained 3 keypoints for the hands. As a result, the model fails to capture the full complexity of sign language.

A large proportion of work focuses solely on the production of manual features, often ignoring non-manuals, co-articulation, sign prosody etc. As a result, Saunders et al. introduced the progressive transformer, a novel transformer-based architecture that can generate both manual and non-manual features [155]. Adapted from the NMT field, by introducing a counter decoding technique, the model was able to deal with the long sequence length of the output sequence. The model was the first to synthesise a skeleton pose sequence directly from the spoken language (Text-to-Pose (T2P)), while also being trained end-to-end. However, peak performance still required the use of a gloss intermediary (Text-to-Gloss-to-Pose (T2G2P)). Later, the model was improved using adversarial training and a Mixture Density Network (MDN) [153, 157]. Huang et al. presented a non-autoregressive G2P transformer architecture [74], which produces the sign sequence in a single step. This approach reduces the computational cost compared to autoregressive architectures. Saunders et al. proposed learning a set of motion primitives using a mixture of experts architecture [157]. Each primitive specialised in a particular sign motion, and by training a gating network, the model can learn to blend between the primitives to create a sign sequence. In subsequent work Saunders et al. proposed using a dictionary of isolated signs with a novel frame selection network to synthesise sign. This helped reduce the problem of regression to the mean, which is common to all previous SLP architectures. More recent

work has attempted to apply diffusion models to SLP with some success [32, 179]. Tang et al. [179] used a transformer encoder to generate a contextual embedding of a gloss sequence. This was then used to condition a diffusion model, which starts from Gaussian noise and iteratively refines the skeleton pose sequence. The model was only trained on a small domain of discourse and only included manual features.

To achieve peak performance, most approaches rely on gloss as an intermediate representation to guide production. However, gloss notation, while helpful for quick transcription, falls short of capturing the nuances of sign language. It overlooks facial expressions, body language, and spatial relationships, which are integral to conveying meaning in sign languages. Alternative representations to gloss have been created namely HamNoSys [68] and SignWriting [175]. HamNoSys is a transcription system that is used to describe sign language at the phonetic level, where each sign consists of a description of the initial posture and the action over time. HamNoSys can be mapped directly to an avatar making it a suitable alternative for gloss in the SLP pipeline [89]. In Chapter 3 we explore using HamNoSys as an intermediate representation in the SLP pipeline. By defining the task of T2H we show improved translation performance. This prompted a similar task this time using SignWriting [83], which is also animatable [15].

Modern deep learning is heavily dependent on data, approximately 15 million sentence pairs are required before deep learning starts to outperform statistical approaches [98]. In contrast, sign language datasets are limited. For example, MeineDGS has only 50k parallel sentences with gloss and HamNoSys annotations [101]. These annotation systems are time-consuming and costly to create, and this has limited the size of the available datasets. In parallel to SLT (as discussed in Section 2.2) the field has shifted towards gloss-free approaches, most of which use VQ.

2.3.1 Vector Quantised Models

Kingma et al. [96] introduced the first Variational Autoencoders (VAE). This model learns to encode input data into a latent space and then decode it. The VAE imposes an additional constraint on the latent space, encouraging it to adhere to a pre-defined prior distribution, typically a standard Gaussian. The model showed impressive results, but it struggled to capture fine-grained structures. Van Den Oord [134] improved on this by introducing the Vector Quantised

Variational Autoencoders (VQ-VAE) architecture. The model integrates VQ into the latent space of a VAE. Forcing the embedding space of the VAE to be discrete, allowing state-of-the-art image and audio generation. Since then, VQ has been applied to several problems, including music generation using a hierarchical VQ-VAE [44], speech synthesis using self-supervised training [4], and more recently image generation using a diffusion model [65]. The original VQ-VAE architecture used an argmin operation to select the closest matching codebook entry. As a result, the model used straight-through gradient estimation to make the model differentiable. Consequently, the model uses three separate losses to train: a reconstruction loss, a codebook loss and a commitment loss [134]. Kaiser et al. [86] used an exponentially moving average to update the codebook. This simultaneously helped stabilise training and reduced the required loss functions to two. Vali et al. [187] then reduced the required losses to one by introducing the Noise Substitution Vector Quantisation (NSVQ) technique. Here, the vector quantisation error is approximated by substituting it for a product of the original error and a normalised noise vector. By scaling the noise vector such that it forms the distribution of the codebook error and adding it to the original embedding, the quantisation error is simulated. The result allows end-to-end training of the model, showing faster convergence compared to straight-through estimation and exponential moving averages models.

VQ has been applied to the SLP [76, 212]. Xie et al. broke the human skeleton into three separate codebooks and uses a diffusion model to translate from G2P [212]. The diffusion process starts from Gaussian noise and is guided using a CodeUnet [150] to produce the final skeleton pose sequence. The approach still relies on expensive linguistic annotation, and qualitative results show a lack of detail in the hands, resulting in under-articulated signing. In Chapter 6 we propose approaching the task using a transformer to construct the codebook and perform the translation, this builds on the published work [193]. We believe that the attention mechanism is more adept to modelling long-range dependencies and the change in order between the source and target. Later works added a down-sampling and length regulator to the approach, aiming to reduce redundant frames and repetition in the data [214].

Other works have attempted to directly translate from text to video, by using a VQ model to learn a dictionary of spatial-temporal video embeddings. However, it is unclear how the model performs across the multiple signer appearances present in the dataset. Xie et al. [211] applied a gumbel softmax to allow the translation and codebook models to be simultaneously trained.

While, Guo el at. [66] trained a model for both SLP and SLT. However, qualitative results are limited, so it is unclear how well the model performs.

2.4 Natural Language Processing

NLP encompasses a wide array of applications, including Text Simplification, Text Classification, and MT. MT, a core NLP task, focuses on the automated translation of text from a source to a target language [167]. Before the advent of deep learning techniques, statistical-based methods represented the state-of-the-art [40, 99, 132]. However, these models encountered difficulties when dealing with significant word order variations and long input sequences[58]. To address these challenges neural networks were applied, giving rise to the field of NMT.

2.4.1 Neural Machine Translation (NMT)

NMT aims to generate a target sequence given an input using deep learning [5]. This can be modelled as a conditional probability, $p(y|x)$, where X is the source sequence and Y is the target sequence.

$$p(Y|X) = \prod_{t=1}^{T} p(y_t|y_1^{t-1}, X) \tag{2.1}$$

Here, the prediction of the target token y_t is conditioned on the previous predictions and the input sequence. RNNs were the first to be used for this task. Kalchbrenner et al. [87] propose using recurrence for sequence-to-sequence modelling. By introducing the concept of a hidden state, the model could retain information from previous time steps, allowing it to learn a context embedding for a given sequence. Initially, recurrence was used in the decoder to map the hidden state to an output sequence [87]. Later, recurrence was used in both the encoder and decoder further improving performance [206].

However, RNNs struggled to model long-range dependencies due to the vanishing gradient problem [10]. So Long Short-Term Memory (LSTM) was introduced, which incorporates a gating mechanism (input, forget, and output gates) to allow the network to selectively retain or discard information from the hidden state. To improve efficiency, the Gated Recurrent Unit (GRU) was later proposed, which combines the forget and input gates into a single update

gate. All these models have a bottleneck problem, where the hidden state (that has a limited size) must retain all the information from the input sequence.

To address the bottleneck problem, the attention mechanism was introduced.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.2}$$

In essence, the attention mechanism allows a model to focus on the most relevant parts of an input sequence when generating an output. It achieves this by computing a weighted sum of the values V, where the weights are determined by the similarity between the query (Q) and the keys (K) [5]. Here, we show Luong's dot product attention [116] to measure this similarity. The softmax function ensures that the weights sum to one, effectively providing a probability distribution over the input. The scaling factor, ($\frac{1}{\sqrt{d_k}}$), is incorporated to stabilise gradients during the training process. This work laid the foundation for the next major improvement, the transformer [189].

2.4.2 The Transformer

Translation models are designed to be auto-regressive, where the previously predicted token is used as input in order to predict the next token in the sequence. RNNs require the hidden state to be updated one token at a time, making them computationally expensive and limiting parallelisation. Vaswani et al. [189] introduced the Transformer as a pure attention base model, that can be parallelised across the entire sequence. The model is composed of an encoder and decoder, each containing a stack of layers, as shown in Figure 2.2.

Each encoder block contains a multi-headed attention mechanism, followed by a feed-forward neural network with skip connections. The decoder is similar but contains an additional multi-headed cross-attention block that allows it to attend to the encoder's output. Unlike a RNN where the input is condensed into a single hidden state, the transformer has access to the input sequence embedding at each decoder layer. This reduced the problem of long-range dependencies and allowed the model to learn complex word relationships. Furthermore, the architecture employed Multi-Headed Attention (MHA) that allows the model to attend to different parts of the input simultaneously, improving the model's ability to capture complex word dependencies.

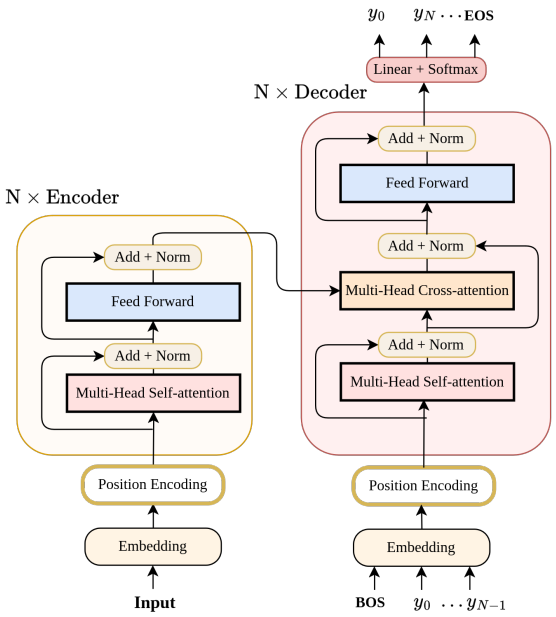


Figure 2.2: An overview of the transformer architecture.

The Transformer became the state-of-the-art architecture for a wide range of NLP tasks and inspired many different revisions. Including language modelling with a GPT [146], question and answering with BERT [42], text summery with BART [105], and sentiment analysis with RoBERTa [112]. The architecture transcended NLP, making significant inroads into other domains. This includes computer vision with Vision Transformer (ViT) [46], Object detection with DETR [28], and even sign language [26]. We leverage this architecture to generate a skeleton pose sequence. Next we discuss the last step in the SLP pipeline, the generation of a photo-realistic signer.

2.5 Photo-Realistic Signer Generation

Research indicates a strong preference among Deaf participants for sign language videos over skeleton representations [190]. Avatar approaches, due to their cartoon-like appearance, are also found to be unpopular [202]. As a result, the field has progressed to using photo-realistic approaches. Sanders et al. proposed the SignGAN architecture, a model capable of producing photo-realistic sign language videos, that are comprehensible by a native Deaf signer [154]. The

model is designed as a Generative Adversarial Network (GAN), trained with a combination of 5 losses. The generator network can be seen below, in Figure 2.3.

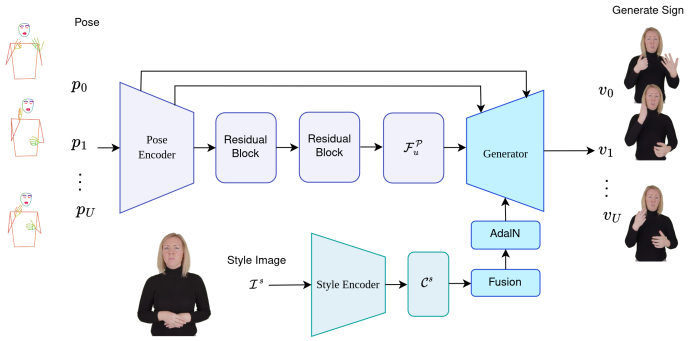


Figure 2.3: An overview of the SignGAN architecture.

A GAN can be thought of as a game between two networks, a generator and a discriminator. The generator's objective is to synthesise photo-realistic images of comparable quality to the ground truth, with the intention of deceiving the discriminator. Conversely, the discriminator's objective is to differentiate between 'fake' and 'real' images. Through this game, the generator network learns to synthesise pose and style conditioned images that are indistinguishable from the ground truth. As can be seen from Figure 2.3, the generator is an encoder-decoder network with skip connections between. The style encoder is used to extract appearance features from a given style image, it does so using a mixture of convolutions and linear layers. We apply this model when conducting a user evaluation.

More recently, diffusion-based approaches have been proposed [52]. To condition the model Bhunia et al. [12] proposed concatenating skeleton keypoints to a diffusion model's input. Other approaches utilise a range of features for conditioning, such as sketches, edge maps, and depth maps [6, 85, 125, 218]. To improve the accuracy of generated hands, Pelykh et al. [139] proposed generating the hands and body separately before blending them. These methods, however, are restricted to image generation. Video generation models, such as Sora [18] have demonstrated impressive results, that are able to generate entire videos from textual prompts. However, these models require vast quantities of data and computational resources for training. For instance, Meta's Movie Gen model is estimated to be trained for 1.25 million hours on a H100 GPU cluster [140]. To mitigate these challenges, we adopt the SignGAN model.

2.6 Sign Language Datasets

Modern deep learning is heavily reliant on data. To achieve the goal of spoken-to-sign language translation, we require aligned spoken language sentences and sign language videos. This section details the sign language datasets used in this thesis.

All datasets are split into 80:10:10 training, validation, and test sets, respectively. The training and validation sets are used during training to optimise the model's parameters, while the test set is used to evaluate the model's performance after training.

The RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**) dataset [25] is extracted from German TV weather broadcasts. This is one of the first CSLR datasets and has become a standard benchmark for both SLP and SLT models. It contains 8,257 parallel sequences, each labelled at the gloss level and includes the aligned spoken language sentence. The dataset is limited in size and only contains nine signers. Furthermore, the domain of discourse is limited to weather reports. We use this dataset to evaluate all approaches presented in this thesis. But in order to move towards unconstrained SLP, we use the MeineDGS and BSL Corpus **T** (BSLCP**T**) datasets.

The publicly available MeineDGS dataset [80] contains aligned spoken German sentences and their gloss counterparts from unconstrained dialogue between two native Deaf signers [90]. The providers of this dataset also have a dictionary for all glosses in the corpus, some of which contain HamNoSys descriptions. Following the translation protocols set in [158], we created a subset of the MeineDGS dataset with aligned sentences, glosses, and HamNoSys. MeineDGS is a larger dataset compared to PHOENIX14**T** (7.5 times more parallel examples, with a source vocabulary of 18,457) with 330 Deaf participants performing free-form signing. The size of MeineDGS overcomes some of the limitations of PHOENIX14**T**. Both the PHOENIX14**T** and MeineDGS datasets contain German Sign Language - Deutsche Gebärdensprache (DGS).

The BSLCP**T** dataset [161] contains BSL from 211 participants from 8 regions in the UK. In total, over 5,000 unique signs are performed from a range of age groups. The participants perform narratives, interviews, and participate in free conversation. We summarise some of the datasets currently available in Table 2.1 below.

Name	Language	Hours	Source	Continuous	Translation	Gloss Annot.	Spoken Vocab.	Gloss Vocab. Size	Signers	Samples	Multiview	Resolution
PHOENIX-2014T [25]	DGS	11	TV	yes	yes	yes	3K	1066	9	8,257	no	227x227
DGS Corpus [80]	DGS	50	Lab	yes	yes	yes	18,457	4,541	327	63,912	yes	640x360
SignBank [55]	BSL	-	Lab	no	no	yes	-	3,465	-	-	no	640x360
BSLCP [161]	BSL	125	Lab	yes	yes	yes	6,267	5,222	249	9,437	yes	640x360
BOBSL [2]	BSL	1,447	TV	yes	yes	no	77K	-	39	1,193k	no	444×444

Table 2.1: Combined statistics for different Sign Language datasets.

Given the limited availability of high-quality sign language data, these languages can be considered low-resource. This has prompted the development of techniques to improve translation performance.

Larger alternative datasets exist, such as the BBC-Oxford British Sign Language (BOBSL) dataset [2], which contains over 1,447 hours of signing. This data was taken from BBC broadcast footage and contains both spoken language subtitles and a video of the sign language interpreter. However, such a dataset suffers from an alignment problem, where the spoken language subtitles precede the sign language video. This shift makes it difficult for both SLP and SLT models to learn effectively. As for any given sequence, the target may only contain half the relevant information, this has resulted in a baseline SLT BLEU-4 score as low as 1.0. Furthermore, the translation comes from a hearing interpreter and not a native signer. This can cause a misalignment between the training data and the target user's [41]. For these reasons, we chose not to scale to automatically curated broadcast datasets.

2.7 Evaluation Metrics

To assess the translation performance of our approach in the following chapters, we utilised the Bilingual Evaluation Understudy (BLEU) Score [137] and the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) score [110]. These metrics are among the most commonly used evaluation measures in the field of MT. When we wish to evaluate the performance of a skeleton production system, we first apply a back-translation network, translating the production back to spoken language before applying the aforementioned two metrics. Additionally, we apply Dynamic Time Warping with a Mean Joint Position Error noted as DTW-MJE to evaluate the quality of the generated skeleton pose sequence. Next, we formally define each metric.

2.7.1 BLEU

This metric is used to automatically evaluate the quality of machine-generated translations by comparing them to human-generated references [137]. Note that the metric can be computed against multiple reference sentences, allowing for a more robust evaluation. But for the corpora used in this work, we only have a single reference sentence.

BLEU score breaks down the predicted and reference sentences into n-grams, where n is the number of grams in each segment. For example, the sentence "The cat in the hat" has;

- Five unigrams (1-gram), "The", "cat", "in", "the", "hat".
- Four bigrams (2-gram), "The cat", "cat in", "in the", "the hat"
- Three Trigrams (3-gram), "The cat in", "cat in the", "in the hat"

The BLEU score is then calculated as the n-gram precision, where the precision is the number of n-grams in the predicted sentence that appear in the reference sentence. However, the precision is modified to account for repetitions in the predicted sentence. This is done by counting the maximum number of times an n-gram appears in the reference sentence and clipping the count such that;

$$\text{Count}_{clipped} = \min(\text{SRC}_{Count}, \text{TRG}_{max\_Count}) \tag{2.3}$$

Where SRC_{Count} denotes the predicted sentence's n-gram count and TRG_{max_Count} is the reference sentence's maximum n-gram count.

Then precision for a n-gram, p_n, can be calculated as follows;

$$p_n = \frac{\text{clipped count}_n}{\text{count}_n} \tag{2.4}$$

The clipping accounts for over-translation, and a penalty is added for under-translation. This is done by calculating the brevity penalty (BP). Defined as the following, where c is the length of the predicted sentence and r is the length of the reference sentence.

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ \exp(1 - \frac{r}{c}) & \text{if } c \leq r \end{cases} \tag{2.5}$$

The final BLEU score is then calculated as the product of BP and the specific n-gram precisions;

$$\text{BLEU-N} = \text{BP}\dot{p}_n \tag{2.6}$$

In this work, we use BLEU 1, 2, 3 and 4 to evaluate the translation performance of our model. This metric evaluates the "adequacy and fluency", two key components of translation [137], where lower grams show the adequacy, while higher grams help evaluate fluency. Note that BLEU score is not perfect, as it does not account for semantic meaning and grammar. Furthermore, when evaluating signed glosses specifically, higher n-grams may under-reward valid paraphrases because sign language exhibits a wide range of ordering variations [127]. These variations are partially due to the impact of non-manuals and the influence of contact with spoken languages [145]. However, despite these limitations, the metric is widely used in the MT field due to its simplicity and computational efficiency.

2.7.2 ROUGE

ROUGE score [110], specifically ROUGE-L F1 score, measures the length of the Longest Common Subsequence (LCS) between the predicted and reference sentences. First, the precision, p, and recall, r, are calculated between the predicted (SRC) and the reference (TRG) sentence, as follows;

$$p = \frac{LCS(SRC, TRG)}{len(SRC)} \tag{2.7}$$

$$r = \frac{LCS(SRC, TRG)}{len(TRG)} \tag{2.8}$$

Where LCS gives the longest common subsequence between the two sentences, and, len returns the total length of the sequence. The final F1 score is given as;

$$\text{ROUGE-L F1} = \frac{2 \times p \times r}{p + r} \tag{2.9}$$

The ROUGE-L F1 score is a useful metric for evaluating the quality of machine-generated text, as it captures the longest common subsequence. However, it does not account for the order of

the words, valid paraphrases and synonyms, which can lead to misleading results for both T2G and T2H tasks. Furthermore, given that a HamNoSys sequence can contain more characters compared to gloss, the scores may be inflated and therefore the two tasks are not be directly comparable. Despite this limitation, ROUGE-L F1 is a useful evaluation metric.

2.7.3 DTW-MJE

To evaluate the accuracy of a generated skeleton pose sequence, we use Dynamic Time Warping Mean Joint Error (DTW-MJE). This metric is commonly used in the field of SLP [155, 157]. The metric aligns two time series by stretching or compressing them locally in time to find the optimal match, minimising the overall distance between the ground truth (GT), $\hat{p}_u$, and predicted sequence, p_u. This is necessary as the predicted sequence may vary in length to the GT. Thus, the metric first calculates the index alignment;

$$A_{i,j} = DTW(p_u, \hat{p}_u) \tag{2.10}$$

After the alignment, we compute the mean joint error between the two;

$$\text{DTW-MJE} = \sum_{u=0}^{U} |p_u[A_i] - \hat{p}_u[A_j]| \tag{2.11}$$

Note we normalise the skeletons between the range of zero and one before calculating DTW-MJE.

Similar to our text-based metrics, this evaluation has limitations. Given we only have a single reference sequence to compare each production to, this metric fails to account for slight lexical variations performed by different signers. Furthermore, each signer possesses a distinct style that affects the trajectory of the skeleton pose, even for the same sign. As a result, scores for valid translations can be lower than expected.

2.7.4 Back-Translation

In line with existing works, we apply the Sign Language Transformers [26] architecture to perform back-translation. This is done by translating the predicted skeleton pose sequence,

$P = (p_1, p_2, ..., p_U)$, back to spoken language, $X = (x_1, x_2, \ldots, x_W)$. Thus, the model learns the conditional probability, $p(X|P)$.

We apply BLEU and ROUGE metrics to evaluate the quality of the back-translated text, which serves as a proxy for assessing the quality of the generated sign language. However, we note the inherent limitations of this approach for SLP. The SLT model may exhibit biases toward specific signing styles and lexical variations present in its training data. Furthermore, due to the two-step evaluation pipeline, it is difficult to disambiguate the source of errors, as they could originate from either the SLP model or the reverse translation process itself. This can result in a lower-than-expected score, even when the produced sign language is of high quality. Despite these shortcomings, back-translation remains a valuable automated metric for evaluating the quality of generated sign language.

To complement these automatic metrics and provide a more comprehensive assessment, human evaluation, such as comprehension checks, can be used to assess the quality of the back-translated text.

2.8 Summary

In this chapter, we examined the field of computational sign language research, spanning SLT to SLP. While deep learning has significantly advanced both translation directions, challenges still exist. The complexity of sign language, including manual and non-manual features, co-articulation and prosody, demands more attention.

Chapter 3

Intermediary Representations for Sign Language Production

Sign Language Production (SLP) aims to bridge the gap between hearing and Deaf communities by translating from spoken language sentences to sign language sequences. This problem has historically been broken into three steps; 1) Text-to-Gloss (T2G), 2) Gloss-to-Pose (G2P) and 3) Pose-to-Sign (P2S). The majority of SLP research has disproportionately focused on the G2P task, with the T2G task receiving comparatively little attention. This creates a significant bottleneck in the SLP pipeline, as the quality of subsequent steps, G2P and P2S, is heavily reliant upon the initial T2G translation. In this chapter, we address this limitation by focusing on enhancing the initial translation. To achieve this, we first turn our attention to a critical factor: data representation.

Data representation is a fundamental aspect of deep learning. Despite the ability of deep models to automatically extract features from large quantities of minimally processed data, the choice of representation can ease the task, allowing the model to learn better features and therefore improve the accuracy. In the context of SLP, which is a low-resource language, the choice of representation is particularly important to achieve good performance. Figure 3.1 shows three different representations of the same sentence, “I‘m pretty kindhearted concerning things like that.”. The top representation is the original video stream, followed by the Hamburg Notation System (HamNoSys) and gloss notations.

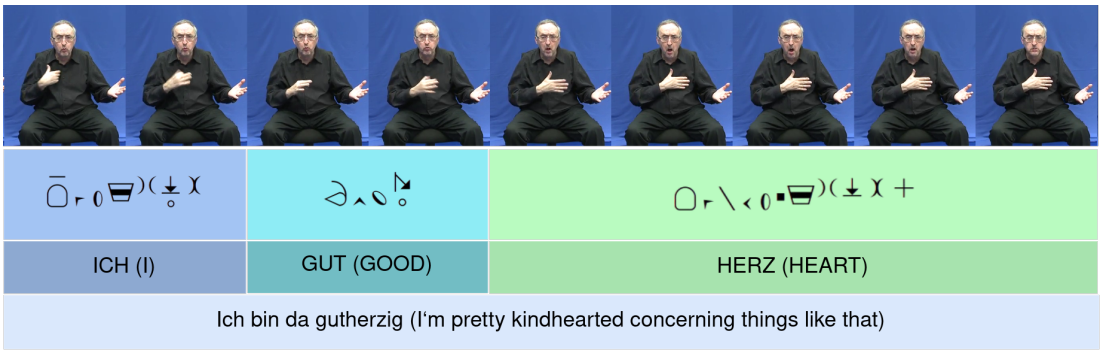


Figure 3.1: Three different representations of the sentence, "I'm pretty kindhearted concerning things like that.", showing from top to bottom: Original Video stream, Hamburg Notation System (HamNoSys), gloss, and spoken language translation. Note, brackets show the English translation.

Figure 3.1 shows an example of HamNoSys. HamNoSys is a transcription system for all sign languages, which encodes signs using a set of around 200 symbols. It can be viewed as a phonetic representation of sign language [68]. There are three main components when representing a sign in HamNoSys; a) its initial configuration b) its hand shape, and c) its action.

Traditionally, gloss has been used as the default representation for sign language translation. However, it has little information about the sign itself. In contrast, HamNoSys, a direct alternative notation system, has information directly related to the form of the sign, which can be used to provide additional information. In sign language, signs with a similar meaning can often share features, such as hand shape or movement. For example, in BSL a closed fist with only the pinky finger extended is used to sign, "BAD", "TERRIBLE", "SWEAR", "WEAK", "ARGUE" and "POISON" [9], all of which share a negative sentiment. Because of these features, we choose to investigate the impact of using HamNoSys as an alternative representation for the T2G task, and introduce the task of T2H.

Beyond the fundamental choice of representation, the method by which a sequence is segmented into discrete units (a process known as tokenisation) can significantly influence performance. In the domain of NLP, a string of characters may be tokenised into words, characters, sub-words, or even phonemes. All of which have different tradeoffs. For instance, sub-word tokenisation algorithms such as Byte-Pair Encoding (BPE) [163] and WordPiece [206] have been demon-

strated to enhance language understanding, leading to improved performance while also allowing models to handle out-of-vocabulary words [42]. But as a result, it may produce words that do not exist. In this work, we investigate the application of diverse tokenisation strategies to the representation of sign language and analyse the implications for translation performance.

Furthermore, to address the challenges posed by this low-resource language task, we investigate leveraging pre-trained language models such as BERT [42] and Word2Vec [120] to generate enhanced word and sentence-level embeddings. Given the recent success of pre-trained language models in NLP, we hypothesise that we can leverage the knowledge learned by these models to improve the performance of SLP models. We conduct experiments incorporating contextual information in the form of sentence embeddings to augment the information available to the network.

We evaluate our translation models on both the MeineDGS and PHOENIX14**T** datasets, and group our experiments into four main categories; 1) baseline performance, 2) the use of pre-trained language models, 3) the choice of tokeniser and 4) the use of additional supervision. In addition, we share qualitative results to provide insight into the performance of our models.

The rest of this chapter is structured as follows; In Section 3.1 we review the related work specific to this chapter. Section 3.2 presents our methodology. Section 3.3 explains the experimental setup for the following quantitative and qualitative results in Section 3.3.2 and 3.3.3, respectively. Finally, we draw conclusions in Section 3.4.

3.1 Related Work

3.1.1 Word Embedding models

While vector space models have been employed since the late 1970s [152], the seminal work of Bengio et al. [11] in the early 2000s introduced the concept of utilising neural networks to learn ”distributed representations” of words, now commonly referred to as word embeddings. However, it wasn't till later that the power of pre-train word embedding models was realised, and since then a range of models have been developed, such as FastText [14], BERT [42], Word2Vec [120], and, GloVe [141]. These models have been used in a range of NLP tasks, such as named entity recognition [138], sentiment analysis [169], and machine translation [207].

At their core, word embedding models transform words into a numerical vector representation. A good model will learn to arrange this high-dimensional space, clustering words that are semantically similar. This is achieved by training on a large corpus of unlabeled text data. Word2Vec is one of the most popular models, it uses a shallow neural network to learn the vector representation of words. The model is trained using either a skip-gram or Continuous Bag of Words (CBOW) algorithm. Essentially, the model learns to predict the surrounding context given a word (skip-gram) or the word given a context (CBOW) [119]. Through this method, the model learns to give similar vectors to words that are used in similar contexts. As a result, you can use these models to find synonyms by finding vectors that are close in space. Or even perform arithmetic operations on words, such as;

$$\text{Queen}_{vec} \approx \text{King}_{vec} - \text{Man}_{vec} + \text{Woman}_{vec} \tag{3.1}$$

This shows that linear transforms can be used to navigate the embedding space, and discover meaningful connections between words. In the following experiments we use Fasttext, an extension of Word2Vec, that incorporates sub-word information in the form of character n-grams [14]. This means for the word “player” and for an n-gram of two, the model would receive the vectors for “pl”, “la”, “ay”, “ye”, “er”. This helps the model understand rare words, as well as prefixes and suffixes (e.g. “er”).

With the improvement of deep learning techniques, the size and depth of models have expanded. This led to the development of BERT, a transformer-based model that has shown state-of-the-art

performance in many NLP tasks [42]. BERT is a bidirectional model, meaning it can see all tokens in a sequence, future and past. This enables the transformer architecture to learn the context of words. The model is trained with masked language modelling, where the model aims to predict a masked word in a sentence. This allows the model to learn the meaning of words, and understand the structure of language. The model is then fine-tuned on a specific task, such as translation, sentiment analysis, etc. Compared to FastText, BERT's attention mechanism enables the model to consider a larger context window, thereby producing a richer representation of the text. However, as sign language is a visual language, a domain gap exists. Therefore, it is unclear whether the additional context provided by BERT will be useful. As a result, we choose to experiment with both BERT and Word2Vec.

3.1.2 Text-to-Gloss Translation

The task of translating spoken language to sign language, facilitated through an intermediate gloss representation (T2G), remains relatively unexplored compared to later stages in the sign production pipeline. The scarcity of annotated parallel data, crucial for training effective translation models, poses a significant challenge. While NMT has demonstrated remarkable performance in large data scenarios, it often struggles with low-resource languages like those found in the sign language domain [173]. Here we survey the existing work in T2G translation.

As sign is a low-resource language, some have explored automatically generating more data. Both Moryossef et al. [124] and Yao et al. [213] proposed augmenting monolingual text data to produce pseudo-parallel text-gloss pairs. By applying rules based and back-translation techniques, they were able to generate two sets of pseudo data. The rules based method can be summarized as, 1) Lemmatization, 2) POS dependent random word deletion, and, 3) Random word permutation. While, the back-transition method employs a deep model to perform the reverse translation direction. Yao et al. also added a training schedule that starts with the pseudo data and progressively adds larger amounts of real data. Both approaches show improved performance from using the pseudo data.

Zhang et al. [220] showed the importance of hyperparameter selection. Demonstrating that performance can increase up to 2.65 BLEU score for Gloss-to-Text (G2T) while in the reverse direction (T2G) it can vary up to 3.39 BLEU score. Interestingly, higher translation scores were

achieved in the G2T direction, a significant 7.95 BLEU score difference. In addition, Zhang et al. [220] also experimented with back-translation, and the results were in line with Moryossef et al. [124] and Yao et al. [213]. While Gomez et al [63], showed improvements by fusing the linguistic features such as word dependencies into a transformer based model.

Instead of generating pseudo data, Fang et al. [53] pooled multiple sign language datasets from different languages to create a large scale multilingual dataset. As sign language annotation schemes are not universal, the dataset needed to be curated. In total 200 hours of data was collected. The dataset was used to train a large scale model (from 40 million to 1 billion parameters), named SignLLM. Two training strategies were tested, first a model where each language is trained individually with a separate encoder-decoder, and secondly a multilingual model. Despite the success of other multilingual models such as mBART [113], the multilingual model showed no improvement over the first. However, they were able to show improvements in training time by formulating the loss function as a reinforcement learning problem and selectively sampling difficult data points.

Recent works have exploited the lexical overlap between the spoken language and the gloss annotation scheme to improve performance. Li et al. [108] proposed approaching the problem using a deep “editing program”. Where instead of performing a direct T2G translation, the approach applies a set of actions, such as adding, removing, or, copying spoken language words. A process labelled “glossification”. By splitting the model into a generator that produces a sequence of actions and an executor that applies them, the approach showed improved performance. The generator is a separate auto-regressive transformer and therefore would be unable to see the effect of previously predicted actions, made by the executor. To address this, a novel causal attention mechanism is introduced, which allows the generator to see the output from the executor.

3.2 Methodology

The task of neural sign language production aims to map a source sequence of spoken language, $X = (x_1, x_2, \ldots, x_W)$ with W words, to a sequence of glosses, $Y = (y_1, y_2, \ldots, y_G)$ with G glosses (Text-to-Gloss (T2G)), or a sequence of HamNoSys, $H = (h_1, h_2, \ldots, h_S)$ with S symbols (Text-to-HamNoSys (T2H)). T2G and T2H tasks thus learn the conditional probabilities $p(Y|X)$ and $p(H|X)$, respectively. To translate into HamNoSys, it is also possible to utilise a gloss intermediary, referred to as Text-to-Gloss-to-HamNoSys (T2G2H), as depicted in Figure 3.2. In this approach, we initially perform a T2G translation, followed by a dictionary lookup to convert each gloss into its corresponding HamNoSys representation. A dictionary of glosses and their equivalent HamNoSys notations is curated for this purpose.

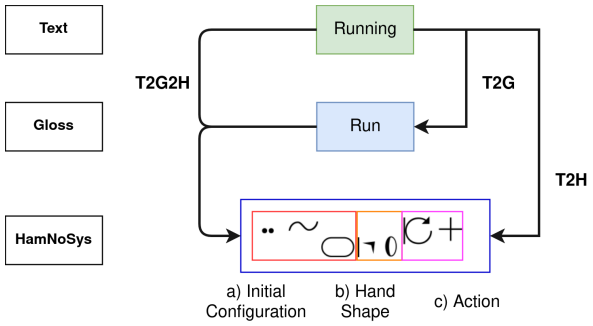


Figure 3.2: A graph to show the word "running" which would be 'glossed' as RUN and the associated sequence of HamNoSys, Top: Text, Middle: Gloss, Bottom: HamNoSys. HamNoSys is split into: a) it's initial configuration b) it's hand shape 3) it's action.

Sign language translation is not a one-to-one mapping as several words can be mapped to a single gloss. Thus, the spoken language sentence is generally longer than its corresponding gloss and HamNoSys sequences e.g. $(W > G)$, $(W > S)$. This increases the complexity of the problem as the model must learn to attend to multiple words in the input sequence. Therefore, we utilise the state-of-the-art Transformer architecture to perform the translation. The Transformer's attention mechanism has shown to be effective in capturing long-range dependencies, which is crucial for sign language translation as the source and target language have a distinctly different ordering. Figure 3.3 shows the general architecture of our model used to translate from spoken language to gloss/HamNoSys. For means of comparison, our baseline model is an encoder-

decoder transformer with MHA. The input and output sequences are tokenised at the word level, and the embedding for a given token is created using an embedding matrix. Later, we build on this base model using different tokenisers, embedding and supervision techniques. We train our model using a cross-entropy loss between the predicted target sequence, $\hat{Y}$, and the ground truth sequence, Y^*, defined as L_{Cross}.

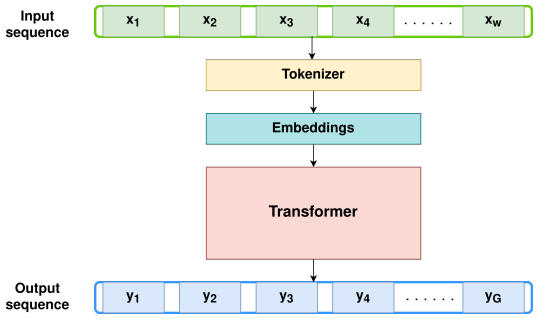


Figure 3.3: A figure showing a simplified overview of a Transformer model for machine translation.

In this section, we follow the structure of Figure 3.3 from top to bottom. We start by describing the different tokenisers used to split the source text (Section 3.2.1). Next, we explain the different embedding techniques used to create a vector from the input tokens (Section 3.2.2). Finally, we talk about the advantages of using extra supervision and explain how this is implemented in conjunction with the translation loss.

3.2.1 Tokenizers

Several tokenisation schemes have been proposed in the literature, which can be applied to both the source and target languages. These include simple approaches such as work and character-level tokenisers and more sophisticated algorithms such as BPE [163] and WordPiece [162]. The latter are known as sub-unit tokenisers, which break the input word into smaller units. This can reduce the vocabulary size, the number of singletons (words occurring only once) and lexical inflexions in the data [201]. In this work, we experiment with using different tokenisers to see how they impact the translation performance. We use the following tokenisers:

- **Word**: A word-level tokeniser segments the input sentence based on whitespace. Thus, a sentence is split into whole words.

- **Character**: A character-level tokeniser segments the text into individual symbols, reducing the vocabulary to simply the alphabet, punctuation and other symbols.

- **BPE**: BPE, first introduced in [163], creates a base vocabulary containing all the unique symbols in the data. It then learns a number of merge rules based on the most frequently occurring sequential symbols. An example of the BPE algorithm being applied to HamNoSys is shown in Figure 3.4, with the coloured boxes indicating the merges made at each step. Merging continues until a specific vocabulary size is reached. This helps reduce word inflexions; for example, the words "low","lowest" and "lower" can be segmented into "low", "est", and "er". Over the entire corpus, the suffixes ("est" and "er") can be reused, collapsing the vocabulary in this example from 3 to 1.

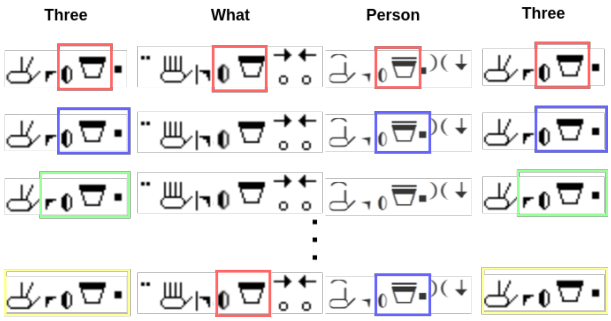


Figure 3.4: An example of how BPE can be applied to HamNoSys.

- **WordPiece**: WordPiece is another sub-unit tokenisation algorithm similar to BPE, but it also evaluates the lost benefit before merging two symbols [42]. It selects the highest score of all the potential merge operations, using the following formula:

$$\text{Score}(a, b) = \frac{\text{Count}(ab)}{\text{Count}(a) \times \text{Count}(b)} \tag{3.2}$$

where Count is the frequency of a given token in the corpus and a and b are the two tokens being evaluated for merging. This scoring function ensures that all merges are beneficial. This operation continues until the desired vocabulary size is reached. We apply this algorithm when embedding with BERT.

3.2.2 Embedding

After tokenisation, the input sequence X is embedded by projecting the discrete tokens into a continuous space [119]. The goal of embedding is to minimise the Euclidean distance between words with similar meanings. The most common technique uses an embedding matrix, which is a large lookup table where each word in the vocabulary is assigned a vector representation. This matrix is randomly initialised, and through training, the objective is to learn the optimal representation for each word. Therefore, for an input sequence $X = (x_1, x_2, \ldots, x_W)$with W words, the corresponding sequence embedding is a matrix of $[W \times E]$, where E is the model's embedding width. In models such as BERT and Word2Vec, embeddings are learnt via self-supervised training on a large corpus of spoken language data. As discussed, a domain gap exists between the spoken language and the gloss annotation scheme. Therefore, to maximise the benefit from using BERT we fine tune the pre-trained model on the MeineDGS dataset using masked-language modelling.

When using a BERT model, we define the transformation as follows. Given an input sequence x we first apply WordPiece tokenisation.

$$X_{\text{WP}} = WordPiece(X) \tag{3.3}$$

Then apply the BERT embeddings as:

$$\mathbf{X}_{\text{BERT}} = BERT(X_{\text{WP}}) \tag{3.4}$$

Note that we take the embedding from the last layer of BERT (excluding the output layer). We define the Word2Vec transformation as:

$$\mathbf{X}_{\text{W2V}} = Word2Vec(X) \tag{3.5}$$

Furthermore, we investigate the impact of incorporating contextual information in the form of sentence embeddings. We explore two methods for integrating this contextual data into the input sequence X: concatenation and fusion. The contextual information, denoted as $\mathbf{X}_{\text{ave}}$, is accompanied by a scaling factor, λ_{x}, which allows for adjusting the emphasis placed on this contextual data.

In the case of Word2Vec we calculate the average of all token embeddings within the sentence, treating this resultant vector as a representation of the entire sentence. This can be expressed as:

$$\mathbf{X}_{\text{ave}} = \frac{1}{W}\sum_{i=1}^{W} x_{W2V_i} \tag{3.6}$$

For BERT, we leverage the embedding of the classification token ($[CLS]$), which contains contextual information about the sentence [43]. For each method, we either concatenate the information to the beginning of a sequence $x = (\mathbf{X}_{\text{ave}} * \lambda_{\text{x}}, x_1, x_2, ..., x_W)$ (CON), or we fuse it into each step of the sequence $x = ((\mathbf{X}_{\text{ave}} * \lambda_{\text{x}}) + x_1, (\mathbf{X}_{\text{ave}} * \lambda_{\text{x}}) + x_2, ..., (\mathbf{X}_{\text{ave}} * \lambda_{\text{x}}) + x_W)$ (ADD).

3.2.3 Supervision

In sign language, there exists a strong correlation between hand shape and meaning [172]. Therefore, we investigate forcing the transformer to predict the hand shape alongside each gloss or HamNoSys token, to enrich the learnt representation. We scale the loss from the hand shape prediction L_{Hand} by a factor of λ_{H}. We then combine both losses from both translation, L_{Cross} and hand shape prediction, L_{Hand}, to create the total loss, L_{Total}, denoted as:

$$L_{\text{Total}} = L_{\text{Cross}} + (L_{\text{Hand}} * \lambda_{\text{H}}) \tag{3.7}$$

In this setup, the model learns the joint conditional probability;

$$p(Y|X) * p(HS|X) \tag{3.8}$$

where HS represents the sequence of hand shape symbols, such that, $HS = (h_1, h_2, \ldots, h_G)$, and Y once again denotes the gloss sequence. We hypothesise that by forcing the model to predict handshape, it will enable the model to group semantically similar signs, allowing the model to learn a richer representation of the sign language

3.3 Experiments

In this section, we test the translation performance of our models in both T2G and T2H setups. We first explain the experimental setup of our models. Subsequently, we share quantitative and qualitative results from our experiments.

3.3.1 Implementation Details

When training our T2G and T2H models, we experiment with different embedding sizes, number of layers and the number of attention heads. We observe a large change in performance based on these hyperparameters. We perform a two-stage grid search to find the best configurations for further tests. Our transformer model employs Xavier initialisation [62] with zero bias and Adam optimisation [95] with a learning rate of 10^{-4}. Additionally, the model uses dropout connections with a probability of 0.2 to reduce overfitting [171]. When decoding, we apply a beam search algorithm with a beam width of 5.

Our code base for the following experiments comes from Kreutzer et al. NMT toolkit, JoeyNMT [103] and is implemented using Pytorch. We utilise Huggingface's transformers library [201] for both BPE and WordPiece tokenisation. When embedding with BERT, we use the open source pre-trained model from Deepset [31] and when embedding with Word2Vec we use FastText's implementation from Facebook [120]. Finally, we utilize the PHOENIX14**T** dataset [25] to benchmark our model against prior state-of-the-art works [107, 124, 155, 173].

Typically, a HamNoSys sequence is longer than its corresponding gloss counterpart, ($S >> G$). Consequently, our T2H performance is artificially inflated compared to our T2G performance. Therefore, to ensure comparability between T2G and T2H results, we employ a dictionary lookup to convert glosses to HamNoSys (T2G2H) prior to calculating the evaluation metric (BLEU and Rouge scores). To facilitate this, we construct a dictionary of parallel HamNoSys and glosses pairs derived from the MeineDGS dataset [90].

3.3.2 Quantitative Evaluation

We group the experiments into four main categories: 1) baseline performance, 2) the choice of tokeniser, 3) use of pre-trained language models and 4) the use of additional supervision. After

we compare our best models to state-of-the-art work on the PHOENIX14**T** and MeineDGS datasets.

Baseline Results

Our baseline models achieved a BLEU-4 score of 2.86 (T2G), 16.26 (T2G2H) and 14.46 (T2H) on the MeineDGS dev set. Our baseline setup uses a word level tokeniser on both the input and output, providing a baseline to ablate our proposed techniques in the next three sections. We perform a hyper-parameter search and make modifications to the model architecture (number of heads, layers and embedding size) to find the best performance. The optimum number of heads is found to be 8 with 2 encoder layers and an embedding size of 512 for the T2G task. While for the T2H task, 4 attention heads are found to be optimal.

Tokenizer

Here we experiment with using different tokenisers, as described in Section 3.2.1. A parameter search was conducted to determine the optimal vocabulary size for the BPE algorithm, which we find to be 7000 and 2250 on the input and output, respectively. Notably, the optimal gloss vocabulary is approximately half of the original size. The results of our experiments are presented in Table 3.1. Character-level tokenisation drastically increases sequence length. This leads to higher computational costs and makes it harder for the attention mechanism to learn meaningful relationships, resulting in reduced performance. This was detrimental regardless of the target language (T2G2H and T2H) and output tokeniser. Therefore, we have omitted these results.

Employing a word-level tokeniser yielded reasonable results, suggesting that utilising larger linguistic units that contain more information is advantageous for translation tasks. However, given that BPE outperformed the word-level tokeniser in terms of BLEU-4 scores, we infer that relying solely on whole words presents a more challenging task, considering the presence of several word inflexions within the dataset. Therefore, we conclude that BPE emerges as the most suitable algorithm for sequence ordering, possibly due to the reduced vocabulary size. However, for the T2H task specifically, using whole units of HamNoSys is best for translation accuracy.

Tokenizer		TEST SET					DEV SET				
Input	Output	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
Word	Word	43.54	33.14	24.45	16.55	36.32	43.41	33.06	24.35	16.47	36.34
Word	BPE	**47.08**	**36.02**	**28.31**	**21.87**	36.20	**47.55**	**36.32**	**28.53**	**22.06**	35.74
Word	Char	43.54	33.14	24.45	16.55	36.32	43.41	33.06	24.35	16.47	36.34
BPE	Word	44.97	34.12	26.80	20.84	35.31	44.77	34.02	26.77	20.84	35.35
BPE	BPE	43.86	34.34	27.25	21.28	**36.61**	43.86	34.31	27.28	21.39	**36.86**
BPE	Char	29.77	10.0	5.18	1.46	2.61	30.01	10.35	5.5	1.99	2.61

a) MeineDGS Annotated (MeineDGS) on Text-to-Gloss-to-HamNoSys (T2G2H)

Tokenizer		TEST SET					DEV SET				
Input	Output	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
Word	Word	**55.04**	**42.16**	**31.92**	21.89	**55.23**	**54.88**	**42.05**	**31.86**	21.81	**55.39**
Word	BPE	41.35	34.25	29.39	25.54	48.09	41.25	34.11	29.28	25.41	48.03
Word	Char	55.01	42.09	31.79	21.59	55.14	54.94	41.99	31.76	21.63	55.3
BPE	Word	49.84	39.4	30.37	21.18	55.04	50.04	39.38	30.29	20.98	55.24
BPE	BPE	44.14	36.43	30.84	**26.21**	50.05	44.35	36.47	30.83	**26.14**	49.95
BPE	Char	37.63	11.59	5.88	1.92	37.31	37.56	11.72	6.01	1.91	37.22

b) MeineDGS Annotated (MeineDGS) on Text-to-HamNoSys (T2H)

Table 3.1: The effect of different tokenizers on Text-to-Gloss (T2G) and Text-to-HamNoSys (T2H) translation.

Embedding

Motivated by the success of pre-trained language models in other NLP tasks, we experiment with using BERT and Word2Vec embeddings in combination with fusing or concatenating contextual information, denoted as ADD and CON, respectively. We also present results with a linear layer as a baseline. The experimental results are presented below in Table 3.2.

From the results, several observations can be made. Firstly, employing a language model improves the translation performance on the T2H task (Table 3.2b). While on the T2G task, using language models is detrimental to the translation performance (Table 3.2a). We hypothesise this is due to the reduced information within the gloss representation. Secondly, we observe that adding sentence-level information into the BERT embedding negatively impacts scores, independent of what method was used (adding or concatenating).

	TEST SET					DEV SET				
Approach:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
Baseline	**43.58**	**33.27**	**24.51**	**16.47**	**41.53**	**43.05**	**32.83**	**24.14**	**16.26**	**42.02**
BERT	38.33	29.09	21.19	14.2	30.31	38.66	29.39	21.51	14.69	30.87
BERT ADD	34.3	26.31	19.47	13.43	32.34	34.75	26.43	19.41	13.23	32.38
BERT CON	36.91	28.79	21.57	15.14	34.44	36.85	28.73	21.45	14.89	34.73
Word2Vec	34.90	25.14	17.83	11.73	30.22	34.21	24.68	17.59	11.47	29.45
Word2Vec ADD	42.13	29.31	20.56	13.31	30.67	42.29	29.72	21.07	13.8	30.65
Word2Vec CON	37.22	26.57	18.01	11.88	29.26	38.41	26.70	18.53	12.17	29.51

a) MeineDGS Annotated (MeineDGS) on Text-to-Gloss-to-HamNoSys (T2G2H)

	TEST SET					DEV SET				
Approach:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
Baseline	47.36	32.89	23.54	14.80	50.87	47.44	32.62	23.27	14.46	50.85
BERT	**49.77**	**38.79**	**29.87**	**21.03**	**53.93**	**48.92**	**38.01**	**29.14**	**20.26**	**53.67**
BERT ADD	44.21	31.41	22.92	15.16	50.33	43.99	30.91	22.33	14.64	50.30
BERT CON	40.48	27.44	19.39	12.21	53.67	40.58	27.39	19.2	11.82	53.36
Word2Vec	47.31	34.22	25.23	17.09	51.52	46.62	33.71	24.77	16.43	51.14
Word2Vec ADD	48.08	34.59	25.31	16.98	51.12	48.00	34.39	25.14	16.72	51.28
Word2Vec CON	42.75	30.80	22.65	15.18	50.10	42.42	30.65	22.49	14.98	51.11

b) MeineDGS Annotated (MeineDGS) on Text-to-HamNoSys (T2H)

Table 3.2: The effect of different Embedding methods on Text-to-Gloss (T2G) and Text-to-HamNoSys (T2H) translation.

On the other hand, adding the summary context to the Word2Vec embedding marginally improved performance compared to the standalone Word2Vec embeddings on T2H (comparing Table 3.2b, row 5 and 6). We find that concatenating Word2Vec context is detrimental to performance relative to the baseline. This may be attributed to the additional context token diverting the model's attention from more important features. Overall, BERT embeddings yield the best performance, surpassing the linear layer by 5.8 BLEU-4 points and even outperforming all T2G2H methods. This demonstrates the potential of pre-trained language models to enhance sign language translation despite the inherent domain gap.

Supervision

Our final ablation study investigates the impact of the additional loss term introduced in Section 3.2.3. While this loss led to improvements in BLEU-3 and 4 scores in the T2H task, as

evidenced by a 0.85 increase in BLEU-4, seen in Table 3.3. It was found to negatively affect performance on the other metrics. While on the T2G2H task it was detrimental on all the metrics.

		TEST SET					DEV SET				
Approach:	Supervision	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
T2G2H	✗	**47.08**	**36.02**	**28.31**	**21.87**	**35.74**	**47.55**	**36.32**	**28.53**	**22.06**	**36.20**
T2G2H	✓	46.11	35.27	27.76	21.49	35.99	46.21	35.45	27.98	21.79	35.79
T2H	✗	**44.14**	**36.43**	30.84	26.21	**50.05**	**44.35**	**36.47**	30.83	26.14	**49.95**
T2H	✓	42.92	36.3	**31.42**	**27.37**	48.85	42.73	35.99	**31.07**	**26.99**	48.89

Table 3.3: HamNoSys hand shape supervision results for Text-to-Gloss-to-HamNoSys (T2G2H) and Text-to-HamNoSys (T2H) translation.

Furthermore, the initial hypothesis regarding the correlation between handshape and meaning does not universally hold. For instance, the signs ”SEA” and ”TREE,” despite sharing a flat handshape, are distinctly unrelated. Therefore, the additional handshape supervision is not always beneficial.

State-of-the-art Comparisons

	TEST SET					DEV SET				
Approach:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
T2G [173]	50.67	32.25	21.54	15.26	48.10	50.15	32.47	22.30	16.34	48.42
T2G [155]	55.18	37.10	26.24	19.10	54.55	55.65	38.21	27.36	20.23	55.41
T2G [107]	-	-	-	-	-	-	-	25.51	18.89	49.91
T2G [124]	-	-	-	-	-	-	-	-	23.17	-
T2G Baseline	58.32	39.99	28.50	20.95	57.28	58.98	41.54	30.03	22.47	57.96
T2G Best Model	**58.74**	**40.86**	**30.24**	**23.19**	**56.55**	**60.04**	**42.85**	**32.18**	**25.09**	**58.82**

Table 3.4: State-of-the-art comparison for Text-to-Gloss (T2G) translation on RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**).

Finally, Table 3.4 (PHOENIX14**T**) and 3.5 (MeineDGS) compares the best performing models to state-of-the-art work. Note in Table 3.4 the baseline is marginally higher than [155], we assume this is due to a larger hyper-parameter search. On both datasets, our best model for T2G and T2G2H uses a word level and BPE tokenizer on the input and output, respectively. While our best T2H result comes from adding additional supervision to this setup. As can be seen from Table 3.4 and 3.5 our models outperformed all other methods [107, 124, 155, 158, 173], setting a new state-of-the-art on PHOENIX14**T** and MeineDGS.

	TEST SET					DEV SET				
Approach:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
T2G [158]	-	-	-	3.08	32.52	-	-	-	3.17	32.93
T2G Best Model	33.59	20.2	14.21	10.4	35.99	33.56	20.43	14.35	10.5	35.79
T2G2H Best Model	47.08	36.02	28.31	21.87	35.74	47.55	36.32	28.53	22.06	36.20
T2H Best Model	**42.92**	**36.3**	**31.42**	**27.37**	**48.85**	**42.73**	**35.99**	**31.07**	**26.99**	**48.89**

Table 3.5: State-of-the-art comparison for Text-to-Gloss (T2G), Text-to-Gloss-to-HamNoSys (T2G2H) and Text-to-HamNoSys (T2H) translation on MeineDGS Annotated (MeineDGS).

In this comparison, we are limited to publicly available scores; hence, '-' indicates instances where the authors did not report results. Additionally, T2H results on PHOENIX14**T** are not provided due to the absence of HamNoSys annotations for certain words within its vocabulary.

3.3.3 Qualitative Evaluation

For qualitative evaluation, we first share translation examples from our best models and our baseline model in Figure 3.5, to allow the reader to better interpret the results. Note, we add a vertical black line after each word of HamNoSys to mark the end of a given sign. This visualisation demonstrates that both models effectively capture the meaning of the spoken language. However, due to the inherent ambiguity in sign language translation, where multiple valid outputs may exist for a given input, the predicted sequence does not always align with the ground truth.

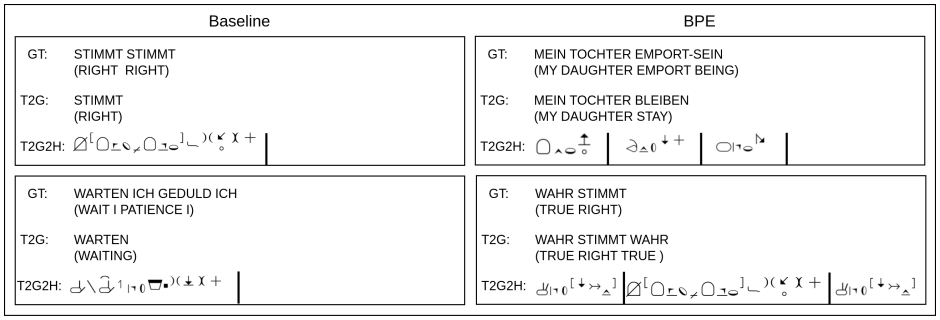


Figure 3.5: Translation examples from our baseline and best model.

The evaluation in fig. 3.6 contrasts the effects of using different word embedding models Word2Vec and BERT from Table 3.2a, at the input of the transformer. We find that BERT, given its

ability to capture sentence-level information and as evidenced by the improved translation scores, yields notable improvements. We observe that BERT reduces the number of hallucinations in the translations. In example 1, for instance, the BERT-based model produces a more concise translation that avoids hallucinating concepts such as the use of mobile phones and email, which are erroneously introduced by the Word2Vec-based model. Additionally, we observe that BERT reduces over-translation in some instances, as demonstrated in example 2. Finally, we note that both models still exhibit limitations, and some semantic content is lost during the translation process, as illustrated in example 3.

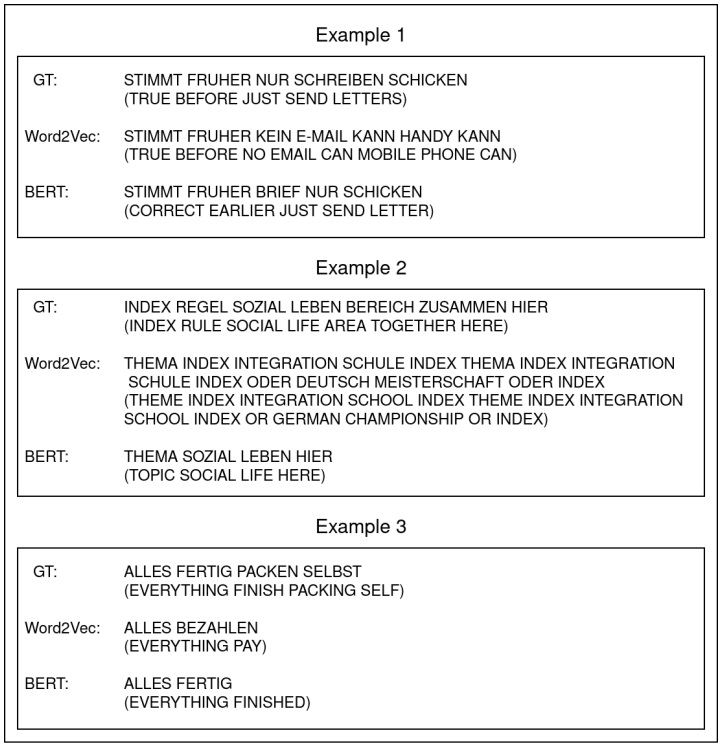


Figure 3.6: Translation examples from models that used Word2Vec and BERT for source embeddings.

3.4 Conclusion

In this chapter, we employed a transformer, a state-of-the-art translation architecture, to translate from spoken language sentences to a sequence of gloss or HamNoSys. We introduced the task of T2H translation, showing the advantages of translating to a phonetic representation instead of a spoken language lexicon (gloss). We set baseline results for future work on the MeineDGS dataset. We showed that language models can be used to improve translation performance, but using more advanced tokenisation algorithms like BPE gives a larger performance gain. Additionally, we have shown that the produced ordering can be improved by training the model to jointly predict hand shape and HamNoSys. We achieved a BLEU-4 score of 26.99 and 25.09, a new state-of-the-art on the MeineDGS and PHOENIX14**T** datasets, at the time of publication.

Previously, HamNoSys has been shown to be animatable, as it has been used to drive an avatar using a statistical-based translation system [38]. Here, we have demonstrated the potential of using deep neural networks for this translation. Therefore, it is possible to reduce the SLP pipeline to an initial T2H translation followed by rule-based animation. But as discussed, these rule-based avatars are unpopular with the Deaf community due to their robotic motion and unrealistic appearance. This led us to develop a more naturalistic animation system using alternate representations in Chapter 5 and 6.

Even though the results of the T2H task are promising, there are several drawbacks. First, connecting the T2H translation to an animation system requires several post-processing steps to ensure the model produces valid HamNoSys strings. This is especially important when using sub-unit tokenisation, which demonstrates the best performance in our experiments. As sub-unit tokenisers have the potential to generate invalid sequences, which would not be animatable. In addition, HamNoSys is currently only available for a single dataset, thus restricting the aforementioned pipeline to DGS. Plus, due to the limited dataset size, the translation performance would be restricted, and the vocabulary limited to the existing domain of discourse. Currently, a complete HamNoSys lexicon is unavailable for other languages, and as this requires expert annotation - a time-consuming and costly process, the feasibility of expanding this approach is limited.

However, gloss annotation is more widely available compared to HamNoSys and has been shown to be effective in achieving peak performance in existing G2P pipelines [155, 173].

Here, we have demonstrated that it is possible to improve translation performance using various techniques. But, we are yet to leverage the significant lexical overlap between spoken languages and gloss annotation schemes. In the next chapter, we will leverage this linguistic property to achieve further improvements in translation.

Chapter 4

Select & Reorder

Gloss intermediaries or supervision remains essential for achieving peak performance in SLP. As previously discussed, the SLP pipeline is often decomposed into three sequential tasks (T2G, G2P, and, P2S). Consequently, the quality of SLP videos is often limited by the initial quality of the T2G translation. In Chapter 3, we explored two different representations of sign language: gloss and HamNoSys. Although we found some advantages to using HamNoSys, as it is a semantically rich notation, gloss still showed competitive results and is more widely available and used in the literature. Therefore, in this chapter, we continue to enhance gloss-based translation by proposing a novel approach to T2G translation, called Select and Reorder (S&R).

Achieving effective translation requires the transformation of a source sentence into the target representation, while ensuring the preservation of the original semantic content. This transformation involves not only lexical substitution but also a change in word order [50]. Sign language annotation schemes, such as gloss, share a large proportion of vocabulary with their country of origin. This causes T2G translation to have a high lexical overlap between the source and target sequences. By first formatting the words and glosses with lemmatization[1] we find that datasets such as the PHOENIX14**T** [25] and MeineDGS [101] have a lexical overlap of 33% and 35%, respectively.

In an attempt to improve translation performance and circumvent the challenges associated with low-resource machine translation, we exploit the lexical overlap, proposing S&R, an

[1]To reduce a word to its base, known as the lemma. Thus, removing any inflexions e.g. “running” and “ran” to “run”

approach that breaks down the translation task into two sub-tasks, Gloss Selection (GS) and Gloss Reordering (GR), as is illustrated in Figure 4.1. Given the following translation example from the PHOENIX14**T** dataset, “in der nacht dreizehn grad auf sylt und null grad in den mittelgebirgstälern ” which is glossed as “NACHT SYLT DREIZEHN GRAD MITTE BERG TAL NULL GRAD”, it is clear that some glosses are a subset of the spoken language sentence. This property is exploited by the first step, GS in the S&R pipeline.

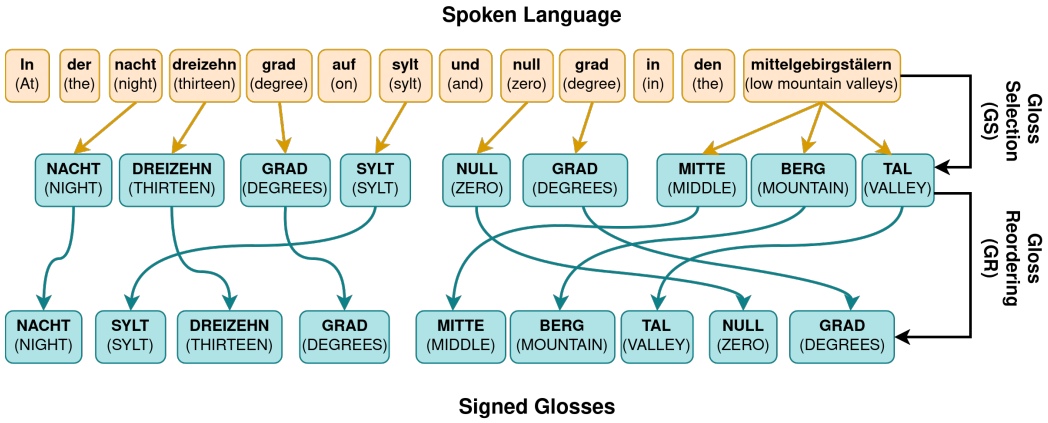


Figure 4.1: An example of Gloss Selection (GS) and Gloss Reordering (GR) being applied to a sentence from the RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**) dataset.

As shown, the first step GS, learns to predict the corresponding gloss for each word in the spoken language sentence, thus producing Spoken Language Order (SPO) gloss [118]. To create the ground truth SPO gloss for training, we obtain a one-to-one alignment between the text and gloss, found using large spoken language models such as BERT and Word2Vec.

For the next step, GR changes the gloss sequence from SPO to Sign Language Order (SIO). We explore two approaches, a statistical based pre-reordering method [126] and a deep learning approach. The statistical approach uses a Top-Down Bracketing Transduction Grammar (BTG) based pre-ordering model, that learns a number of reordering rules using our alignment, Part Of Speech (POS) tags and word classes. The corresponding deep learning approach uses a transformer with a novel reordering mask during inference. The mask constrains the model to only predict tokens that are present at the input, meaning the model is restricted to reordering only.

Both GS and GR are Non-AutoRegressive (NAR) models, executing decoding in a single pass. This characteristic leads to decreased computational requirements and accelerates the inference process, which is a valuable asset for real-time translation.

The rest of this chapter is organised as follows: In Section 4.1 we provide an overview of the literature on statistical translation, then in Section 4.2 we explain our S&R approach to T2G NMT. Section 4.3 explains the setup for the proceeding experiments in Section 4.3.2 where we present quantitative and qualitative results. Finally, in Section 4.4 we draw conclusions from the experiments.

4.1 Related Work

4.1.1 Statistical Reordering Approaches

Prior to the advent of deep learning, statistical-based models were the state-of-the-art in machine translation [19]. SMT approaches can be broadly categorised as string-based or tree-based. The optimal approach is often language-dependent, understandable given the diversity of languages and the complexities of the relationships between them [13].

String-based SMT systems operate on large units of a sentence. These can be further categorised into word-based, phrase-based [19], and n-gram models [30]. While word-based models operate on individual words, and phrase-based models on groups of words, n-gram models consider smaller sequences of words, typically bi-grams (two words) or tri-grams (three words). Both approaches generally treat reordering as a sequential process, producing the target sentence from left to right. This can be problematic for languages with long-range dependencies, where words that are related syntactically may be separated by a considerable distance.

Tree-based SMT systems construct a tree structure, which is then used for translation. These can be further categorised into syntax-based [57] and hierarchical approaches [34]. Syntax-based trees can be constructed using linguistic features such as POS tags, where the words of the sentence form the leaves, and each node represents a feature tag. This structure can then be used to reorder the sentence. In contrast, hierarchical models construct a tree based on the sentence's grammatical structure. This allows the translation task to be broken down into smaller sub-translation tasks. By selecting sub-nodes, smaller sentences can be extracted and then translated [13].

Both tree-based and string-based approaches attempt to jointly address reordering and lexical changes during the decoding phase, a challenging task. Despite significant improvements over baseline methods, these approaches still struggled to capture long-range dependencies in text, meaning they were unable to effectively learn the complex relationships between words [114]. This limitation was a significant drawback, as word order in the source and target languages can differ drastically. Consequently, researchers proposed a two-step approach, where reordering is performed before or after translation, known as pre-ordering [205] or post-ordering [7],

respectively. Similarly, in this chapter we follow this design choise and break the translation into two steps, GS and GR.

The reordering process followed a similar approach to tree-based syntax-driven SMT systems. A set of rules, either manually derived from linguistic knowledge or learned from data, is used to reorder the tree structure. Manually designed rules require expert knowledge of each language pair and are time-consuming to create [35]. In contrast, learned rules can be extracted from large corpora of parallel data [208]. This chapter focuses on the latter approach, as it is more scalable and applicable to a gloss notation system.

Specifically, we utilise a BTG algorithm, first introduced by Wu et al. [205], to construct a tree of the source and target sentences. This algorithm maximises a scoring function to build an optimal structure. However, the original scoring function has a complexity of $\mathcal{O}(n^5)$ [130]. For computational efficiency, we employ a Top-Down BTG parser, which starts at the root and progressively breaks down the sentence into smaller units, allowing the algorithm to prune invalid structures more efficiently [126]. We incorporate a combination of linguistic features, including POS tags, word classes and a novel alignment, to learn the reordering rules for a spoken language and gloss notation system.

4.1.2 Additional Metrics

As discussed in the literature review (Section 2.7), sign languages exhibit a wide range of order variations, meaning that a single translation may have several valid paraphrases. Therefore, evaluating translation quality with a single reference can be problematic. To address this issue, we incorporate additional metrics, specifically chrF++ [143] and BERTScore [219], to provide a more comprehensive evaluation in this chapter. This section details the calculation of these metrics.

chrF++

To start, the Character n-gram F-score (chrF) metric is a character n-gram F-score metric that was specifically designed to address the limitations of word-based metrics like BLEU [137]. It calculates precision P_c and recall R_c based on character n-grams, which allows it to capture

word order changes more effectively. The F-score is then computed similarly to the ROUGE score, however, with the addition of a beta, β, parameter changes the weighting of the precision and recall.

$$F_\beta = \frac{(1+\beta^2) \cdot P_c \cdot R_c}{R_c + \beta^2 \cdot P_c} \tag{4.1}$$

chrF++ [143] extends the original chrF metric by incorporating word n-grams (specifically, unigrams and bigrams) in addition to character n-grams. This allows it to capture both character-level and word-level information. The final chrF++ score is a weighted combination of the character n-gram F-score and the word n-gram F-score.

BERTScore

BERTScore [219] is a metric that leverages pre-trained language models, specifically BERT [42], to evaluate translation quality. It computes similarity scores between the candidate and reference sentences based on contextual embeddings. Each token is then passed through the BERT model to obtain its contextual embedding. The similarity between the candidate and reference tokens is calculated using cosine similarity. A greedy search algorithm is then used to find the best match between the candidate and reference tokens. Finally, the F1-score is computed based on the matched tokens. This metric allows for the evaluation of the semantic content of the sentence rather than focusing on specific lexical variations. We find this metric to be a valuable addition to the reordering evaluation. For this metric, we use Microsoft's DeBERTa [69, 70], a multilingual implementation to calculate embeddings. This model has a vocabulary size of 250,000 tokens and contains 12 layers with 86 million parameters.

4.2 Methodology

Similarly to Chapter 3 we seek to enhance translation performance, but this time we focus on the more widely available notation system, gloss. Therefore, the T2G translation aims to learn the mapping from a source spoken language sequence $X = (x_1, x_2, \ldots, x_W)$ with W words, to a sequence of glosses, $Y = (y_1, y_2, \ldots, y_G)$ with G glosses. Thus, the T2G model learns the conditional probability $p(Y|X)$.

A model that learns $p(Y|X)$ jointly learns a change in lexicon and order, a significant challenge for low-resource machine translation. To address this, we propose a novel approach that disentangles the translation process into two distinct subtasks: GS and GR. This decomposition, illustrated in Figure 4.2, allows each model to specialise in a different aspect of translation. Furthermore, we introduce Text to Spoken Language Order Gloss (T2SPOG), a new task designed to focus solely on the change in lexicon, independent of gloss ordering.

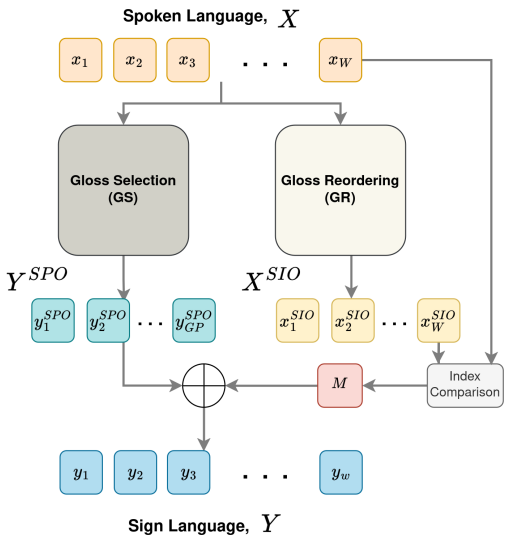


Figure 4.2: An overview of the Select and Reorder (S&R) approach

The GS model learns the mapping of a sequence of words $X = (x_1, x_2, \ldots, x_W)$ to a sequence of SPO glosses and pad tokens, $Y^{\text{SPO}} = (y_1^{SPO}, y_2^{SPO}, \ldots, y_W^{SPO})$. Our approach relies on creating a one-to-one alignment, A, of words to glosses, which limits us to sequences where $(W \geq G)$, hence $X$and Y^{SPO} share the same sequence length, W. To create the gloss in SPO

for the GS model the alignment, $A()$, is applied to the gloss;

$$Y^{\text{SPO}} = A(Y) \tag{4.2}$$

We define GR as a permutation task, where the model learns to reorder words in spoken language order, X, to words in sign order, $X^{\text{SIO}} = (x_1^{SIO}, x_2^{SIO}, \ldots, x_W^{SIO})$. The source and target sequence share the same vocabulary and sequence length, W. Thus, GR learns $p(X^{\text{SIO}}|X)$. To create the text in sign order for the GR model the alignment, $A()$, is applied to the text;

$$X^{\text{SIO}} = A(X) \tag{4.3}$$

As shown by Figure 4.2, to obtain a full translation, the outputs of the GS and the GR networks must be joined. We call this full method Select and Reorder (S&R). To correctly join the outputs, the GR subtask creates a mapping, $M()$. Applying the mapping to the SPO gloss gives a full translation (gloss in sign language order);

$$p(Y|X) = M(p(Y^{\text{SPO}}|X)) \tag{4.4}$$

Both input and target sequences are tokenised at the word level. The GS and GR networks are trained using cross-entropy loss, L_{Cross}, calculated using the predicted target sequence, $\hat{Y}$and the ground truth sequence, Y^*.

In order to perform GS and GR, an alignment, A, must be found. Therefore, in the following sub-sections, we first explain how the alignment is created, followed by the architecture for the GS and GR models. Finally, we explain how the mapping, M, is obtained, which enables the creation of the full translation.

4.2.1 Alignment

Using the lexical overlap between the source and target language, a pseudo alignment can be found. For example, given the sentence "*what is your name?*" it is clear to see which words correspond to which glosses in the translation *"NAME YOU WHAT"*. Using two different word embedding techniques, Word2Vec [120] and BERT [31], we create a mapping between our spoken language words, X, and the glosses, Y. We can define a word gloss pair as a strong

alignment if they share the same meaning. A strong connection can be established if the pair share a similar lexical form (e.g. word = run, gloss = RUN), for which we use Word2Vec. Given that Word2Vec uses character n-grams to calculate the embeddings, it is more suited to finding connections between words and glosses that are simple inflexions. Where an accurate lexical mapping cannot be found, we use BERT to find connections based on meaning (e.g. word = weather, gloss = WEATHERFORECAST). When using DGS we first apply a compound word splitting algorithm [183] before creating the alignment.

For a sequence of words, X, and a sequence of glosses, Y, we apply Word2Vec as:

$$\mathbf{X}_{\text{W2V}} = Word2Vec(X) \tag{4.5}$$

$$\mathbf{Y}_{\text{W2V}} = Word2Vec(Y) \tag{4.6}$$

where $\mathbf{X}_{\text{W2V}} \in \mathbb{R}^{W \times E}$ and $\mathbf{Y}_{\text{W2V}} \in \mathbb{R}^{G \times E}$, with E representing the embedding size of the model. We take the outer product between the resultant two embeddings to give us the Word2Vec alignment:

$$A_{\text{W2V}} = \mathbf{Y}_{\text{W2V}} \otimes \mathbf{X}_{\text{W2V}} \tag{4.7}$$

where $A_{\text{W2V}} \in \mathbb{R}^{G \times W}$. We filter the strongest connections, keeping those that are above a constant, α. Then we repeat the process this time using BERT:

$$\mathbf{X}_{\text{BERT}} = BERT(X) \tag{4.8}$$

$$\mathbf{Y}_{\text{BERT}} = BERT(Y) \tag{4.9}$$

$$A_{\text{BERT}} = \mathbf{Y}_{\text{BERT}} \otimes \mathbf{X}_{\text{BERT}} \tag{4.10}$$

Where $\mathbf{X}_{\text{BERT}} \in \mathbb{R}^{W \times E}$, $\mathbf{Y}_{\text{BERT}} \in \mathbb{R}^{G \times E}$ and $A_{\text{BERT}} \in \mathbb{R}^{G \times W}$. When embedding with BERT, a wordpiece tokenizer is applied to the text. We average the sub-unit alignment in order to create an alignment at the word level. We find BERT embeddings capture the meaning of tokens, making this approach better for finding an alignment between words and glosses that have different lexical forms. The BERT alignment is used to find any remaining connections not found by the Word2Vec alignment, where our total alignment is defined as;

$$A_{Total} = A_{\text{BERT}} + (\alpha * A_{\text{W2V}}) \tag{4.11}$$

Figure 4.3 and 4.4 shows a heat map of the alignment found between a German spoken language sentence and the corresponding gloss sequence from the PHOENIX14**T** and MeineDGS dataset, respectively.

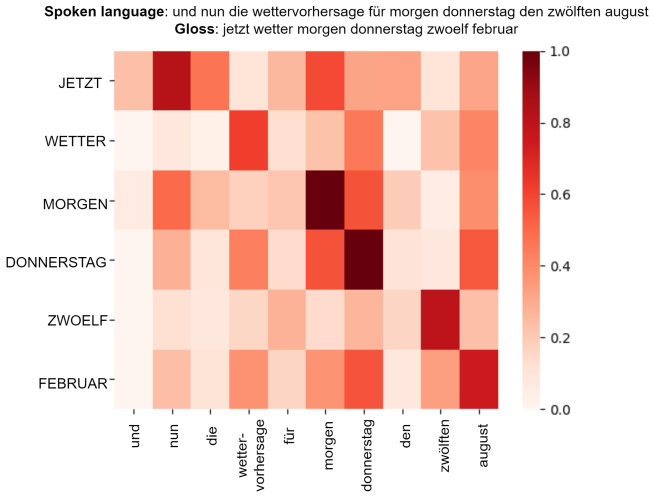


Figure 4.3: An example of the alignment found using BERT embeddings to connect the spoken language to the glosses on the PHOENIX14**T** dataset. (SRC: ”and now the weather forecast for tomorrow Thursday the twelfth of August”, TRG: ”now weather tomorrow Thursday twelve february”)

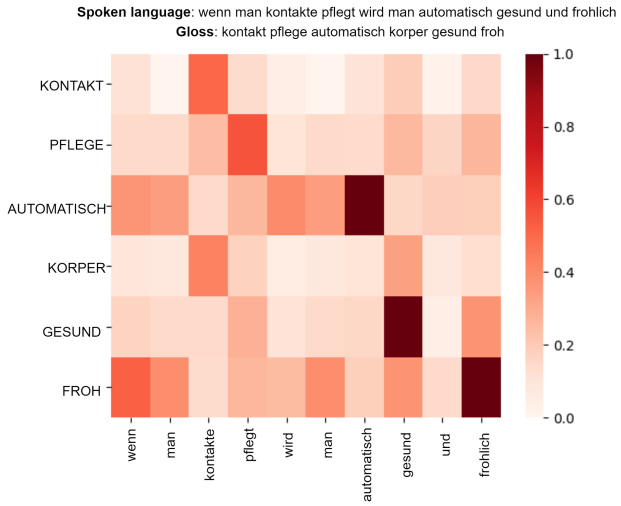


Figure 4.4: An example of the alignment found using BERT embeddings to connect the spoken language to the glosses on the MeineDGS dataset. (SCR: ”when you keep in touch you automatically become healthy and happy”, TRG: ”contact care automatic body healthy glad”)

Figure 4.3 shows that a clear alignment is found between the word and gloss ”MORGEN”, as they share the same lexical form. Additionally, an alignment is found between words with the same meaning e.g. ”JETZT” and ”nun”.

To create the final one-to-one alignment, A, we apply a beam search algorithm to find a near-optimal alignment between the spoken language and glosses. The beam search is a heuristic algorithm that finds the best path through a graph by efficiently exploring the most promising options. It maintains a limited set of top candidates, called the beam, and expands them at each step by finding the path that maximises the alignment score. The alignment score is calculated by summing the values of the alignment matrix, A_{Total}, along the path. The alignment is used to change the order, creating the SPO glosses and the SIO text, used as the targets for each sub-task, as shown in Figure 4.1.

Note, as the alignment creates a one-to-one mapping, the proposed approach is limited to many-to-one sequences e.g. where the source sequence is equal to, or longer than the target, ($W \geq G$). Furthermore, as the T2SPOG task is a many-to-one task, the beam search is stopped after G iterations. Hence, any words that are not aligned are mapped to a pad token, '*'. This ensures the sequence lengths of Y^{SPO} and X are the same.

4.2.2 Select

As shown by Figure 4.1 (top to middle row), GS can be defined as the task of choosing the corresponding glosses for each word in the spoken language sentence. The GS model is a encoder-decoder transformer [189], depicted in Figure 4.5.

The encoder and decoder are passed the same spoken language sentence, whilst removing the auto-regressive feature of the decoder for reduced computational cost. This also makes each prediction independent of the previous, removing any possible negative feedback from incorrect predictions at inference time. Additionally, we alter the decoder's forward masking to allow the model to see all tokens in the sequence. As depicted in Figure 4.5, both the encoder and decoder consist of multiple transformer layers, with each layer containing a multi-head self-attention mechanism and a feed-forward neural network. In addition, the decoder block contains a cross-attention mechanism.

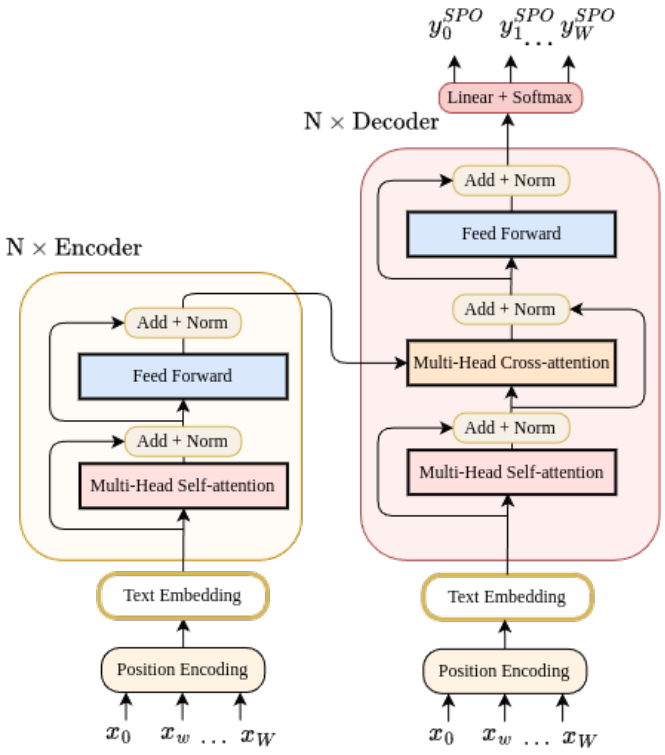


Figure 4.5: A diagram of the Gloss Selection (GS) Architecture

The encoder creates a contextual embedding of the sequence that can then be used by the decoder to predict the glosses. The decoder is trained to predict the glosses in SPO order, using the alignment found in Section 4.2.1. By fusing the context at each layer of the decoder, the model has greater access to the sentence context, which is essential for achieving peak GS performance. Hence, the model is an encoder-decoder despite the fact that both are passed the same input sequence.

4.2.3 Reorder

The goal of the second sub-task, GR, is to create a mapping, $M()$, that reorders a sequence from SPO to SIO. We establish a mapping by analysing the index movements of words as they pass through the model from input to output. A visualisation of applying $M()$ can be seen in Figure 4.1 (middle to bottom row).

To facilitate the creation of ground-truth data for this task, we leverage the alignment discussed in Section 4.2.1, referred to as A. This alignment enables us to generate SPO gloss and text in SIO. Consequently, we have the option to train our reordering model on either the gloss or the text. We opt to train on the text for two reasons. Firstly, gloss does not offer a perfect representation of sign language due to its inherent limitations. Secondly, we hypothesise that training on the higher-resourced language (text) will yield superior performance, as it contains richer structural information about the language.

In this section, we propose two approaches to tackle GR. We start by explaining the statistical approach from [126], followed by our deep learning method.

Statistical Approach

Our first approach uses the BTG method to learn a mapping from spoken to sign order [126]. The approach represents a source sentence as a binary tree, where each non-terminal node can be one of three types: straight, inverted or terminal. The structure of the tree is dependent on the POS tags and word classes of the sentence. To create our word classes, we use the Brown clustering method [20]. Words are grouped into a single cluster if they are semantically related. Words are assumed to be semantically related if the distribution of surrounding words is similar. We use a pre-trained language model to tag the spoken language words with their POS tag.

The model acquires a set of rules designed to restructure the tree in a manner that maximises reordering accuracy. To assess this accuracy, we rely on the alignment provided in Section 4.2.1.

Learnt Approach

Our second approach uses deep learning to learn $p(X^{\text{SIO}}|X)$, using a transformer. Once again, we remove the auto-regressive feature from the decoder and change the masking to allow the model to see all tokens in the input. At inference time we apply a mask to the output, which ensures the model predicts all tokens that are present in the input, hence the model is limited to reordering. The mask is a binary vector with entries only in the index's of the tokens present in the input. At each decoding step, the predicted token is removed from the mask. If duplicate glosses are present, then it is only removed once all copies have been predicted.

4.2.4 Select and Reorder

The GS model learns $p(Y^{\text{SPO}}|X)$ and the GR model learns $p(X^{\text{SIO}}|X)$. To obtain a full translation, the output of the two models must be joined, as shown by Figure 4.2. As depicted, the predictions of the GR cannot be directly applied to the gloss in SPO. By analysing the index movement of words between the input and output a mapping, M, can be created that changes the order from spoken to sign. We train each task independently and join the outputs by applying the mapping, $M()$ from GR to the output of the GS model;

$$Y = M(Y^{\text{SPO}}) \tag{4.12}$$

This provides the full translation from a spoken language input to a target gloss sequence.

4.3 Experiments

4.3.1 Implementation Details

To initialise the encoder and decoder of the transformer, we use Xavier initialiser [62] with zero bias and Adam optimisation [95]. The initial learning rate is set to 10^{-4} with a decrease factor of 0.7 and patience of 5. To mitigate overfitting during training, we employ dropout connections with probabilities of 0.35 and 0.2 for the GS and GR models, respectively [171]. During decoding, we apply a greedy algorithm for both the GS and GR models. We filter the word2vec alignment (A_{Vec}) with a confidence threshold of 0.9. The BTG reordering model is trained for 30 iterations with a beam size of 20. We set the number of word classes to 50 when clustering with [20] and we tag the spoken language with POS using the Spacy Python implementation for German. When embedding with BERT, we use an open-source pre-trained model from Deepset [31]. Finally, we used fasttext's implementation of Word2Vec for word level embeddings [120].

All latency experiments were conducted on a single workstation with the following hardware specifications: an Intel Core i9-10900K CPU, an NVIDIA GeForce RTX 3090 GPU with 24 GB of GDDR6X VRAM, and 64 GB of DDR4 RAM. The software environment included Ubuntu 20.04 LTS and PyTorch 1.12.0 with CUDA 11.0. Inference speed was measured by timing the forward pass on the GPU using torch.cuda.Event. All models were evaluated with a batch size of 512, with the reported metric being the average inference time in milliseconds (ms) over 100 repetitions.

4.3.2 Quantitative Evaluation

In this section, we evaluate our proposed approaches on the MeineDGS and the PHOENIX14**T** dataset. We group our experiments in four sections:

1. Gloss Selection (GS).
2. Gloss Reordering (GR).
3. S&R (GS + GR) and State-of-the-art Comparison.
4. Inference speed tests.

Gloss Selection

Firstly, we evaluate our GS approach. As discussed in Section 4.2.2 we create an alignment for both datasets in order to perform GS. Table 4.1 and 4.2 show the results for the T2SPOG task. In both cases, the GS output is the same but compared against the SPO (row 1) and the ground truth gloss (SIO) (row 2), hence the BLEU-1 score is the same. As the model was trained to predict SPO order, it is not surprising that the BLEU-4 score is higher. However, the performance drop when evaluated against SIO is small. Suggesting there is some shared ordering between the SPO and SIO. The high BLEU-1 scores demonstrate the effectiveness of this method, achieving 42.91 on the challenging MeineDGS dataset.

PHOENIX14**T**	TEST SET							DEV SET						
	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore
GS (spo)	60.13	39.22	27.40	20.19	57.10	50.07	78.94	62.69	41.22	29.04	21.31	58.32	50.91	79.56
GS (sio)	60.13	35.15	21.84	14.49	54.60	48.14	78.50	62.69	38.86	25.67	17.84	56.37	49.61	78.96

Table 4.1: A table showing the result of performing Gloss Selection (GS) on the RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**) dataset.

MeineDGS	TEST SET							DEV SET						
	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore
GS (spo)	43.06	23.23	12.71	7.02	42.65	34.85	70.57	42.91	23.21	12.47	6.89	42.63	35.15	70.58
GS (sio)	43.06	20.97	10.48	5.39	40.60	33.95	69.96	42.91	20.51	9.86	4.95	40.63	34.13	69.95

Table 4.2: A table showing the result of performing Gloss Selection (GS) on the MeineDGS Annotated (MeineDGS) dataset.

Our additional metric, chrF++, closely follows the ROUGE score, which is unsurprising given that both are F1-based metrics. However, BERTScore, a metric that uses contextual embeddings to calculate similarity with the reference sentence, shows strong performance. This suggests that even though our reordering metrics performed poorly (as shown by the reduced BLEU-3 and 4 scores), the semantic content of the sentence is still maintained, as demonstrated by the BERTScore, which was as high as 79.56 on the PHOENIX14**T** dev set.

Gloss Reordering

Next, we compare our different reordering approaches. When evaluating the learnt GR model, any words not present in the training set are replaced with unknown tokens. Thus, the BLEU score

for the learnt method is not 100, even though the model has to predict all words that are present at the input. As can seen from Table 4.3 and 4.4, the statistical method outperforms the learnt approach, with the statistical method achieving 26.51 and 28.64 BLEU-4 on the PHOENIX14**T** and MeineDGS dev sets, respectively. Suggesting that POS tags and word classes are effective features for reordering. The learnt method is found to be detrimental to the ordering of the SPO gloss. We believe this result is due to the lack of large-scale training data. As suggested by [111] 15 million parallel examples are needed for learnt methods to start outperforming statistical methods.

The predictive metric, BERTScore, agrees with most other metrics that statistical-based reordering provides the best performance, showing an improvement of up to 2.14 on the MeineDGS test set over the learned method. The change in score between the two indicates that the metric is still sensitive to order variations. This indicates that the metric is capable of evaluating both the semantic content and word ordering.

PHOENIX14**T**	TEST SET							DEV SET						
Mapping:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore
GT (spo)	100.00	64.00	45.60	34.07	76.17	76.79	92.71	100.00	66.36	48.72	37.43	76.21	77.61	92.43
Learnt	99.20	59.43	38.58	25.74	58.28	75.06	90.21	99.14	60.14	39.50	26.93	59.17	76.03	90.37
Statistical	**100.00**	**61.87**	**42.64**	**31.05**	**74.66**	**76.42**	**92.45**	**100.00**	**76.52**	**62.83**	**53.44**	**84.61**	**81.87**	**94.67**

Table 4.3: A table showing the results of performing Gloss Reordering (GR) from Spoken Language Order (SPO) to Sign Language Order (SIO) on the RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**) dataset.

MeineDGS	TEST SET							DEV SET						
Mapping:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore
GT (spo)	100.00	64.69	42.98	29.35	80.37	78.52	93.02	100.00	65.20	43.72	29.89	80.20	78.67	92.85
Learnt	97.60	59.36	**40.36**	**29.33**	60.32	75.93	90.03	97.62	59.45	40.40	29.29	60.75	76.05	89.91
Statistical	**100.00**	**60.06**	36.87	22.91	**77.47**	**76.28**	**92.17**	**100.00**	**82.12**	**68.67**	**57.93**	**91.24**	**83.38**	**95.37**

Table 4.4: A table showing the results of performing Gloss Reordering (GR) from Spoken Language Order (SPO) to Sign Language Order (SIO) on the MeineDGS Annotated (MeineDGS) dataset.

Row 1 of both tables presents the BLEU scores calculated between the SIO gloss (ground truth) and the SPO gloss. This provides a performance baseline, indicating the maximum score the GS model could achieve, if it was 100% accurate. Table 4.3 shows a high BLEU-4 score of 34.07, indicating there is some shared ordering between SPO and SIO, specifically the ordering

obtained through the alignment method. Therefore, the GS model has the potential to generate a valid translation.

State-Of-The-Art Comparison

The end-to-end S&R approach joins the output from the GS model and the mapping, $M()$, from GR to produce a full translation e.g. $p(Y|X) = M(p(Y^{SPO}|X))$. We used the mapping from the statistical approach as it was shown to give the best performance in Section 4.3.2. In Table 4.5 (PHOENIX14**T**) and 4.6 (MeineDGS) we compare our S&R approach to state-of-the-art work. Note we can only compare scores that are publicly available, therefore '-' denotes where the authors did not provide results.

For comparison on MeineDGS, we train a T2G transformer that achieves a competitive BLEU-4 score compared to [158]. The model is trained till convergence with a beam size of 5 and a word level tokeniser.

PHOENIX14**T**	TEST SET							DEV SET						
Approach:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore
T2G ch3	58.74	**40.86**	**30.24**	**23.19**	**56.55**	-	-	60.04	**42.85**	**32.18**	**25.09**	**58.82**	-	-
T2G	50.67	32.25	21.54	15.26	48.10	-	-	50.15	32.47	22.30	16.34	48.42	-	-
T2G	55.18	37.10	26.24	19.10	54.55	-	-	55.65	38.21	27.36	20.23	55.41	-	-
T2G	-	-	-	-	-	-	-	-	-	25.51	18.89	49.91	-	-
GS	**60.13**	35.10	21.84	14.49	53.78	48.14	78.50	**62.69**	38.86	25.67	17.84	56.37	49.61	78.96
S&R	**60.13**	35.15	22.65	15.60	54.60	**48.58**	**78.71**	**62.69**	40.01	27.07	19.07	56.83	**50.04**	**79.35**

Table 4.5: Baseline comparison results for Text-to-Gloss (T2G) translation on the PHOENIX14**T** dataset.

Our results show that reordering is beneficial to the GS model, increasing the BLEU-4 score by 1.23 and 1.11 on the PHOENIX14**T** Dev and Test sets, respectively. On the MeineDGS dataset, the reordering mapping was found to only benefit the dev set, increasing the BLEU-4 by 1.4, whilst being detrimental to the test set, decreasing the BLEU-4 by 1.25. The reordering performance is significantly reduced when applied to the output of the GS model, decreasing from the theoretical maximum of 53.44 to 19.07 on PHOENIX14**T**. We argue this is due to the number of false positives and false negatives in the output of the GS model.

As can be seen from Table 4.5 and Table 4.6, the S&R models achieved a new state-of-the-art BLEU-1 score on PHOENIX14**T** and MeineDGS, outperforming all other methods [107, 155,

MeineDGS	TEST SET							DEV SET						
Approach:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	chrF++	BERTScore
T2G Baseline	31.25	15.08	7.26	3.38	34.98	-	-	31.12	14.32	6.49	3.04	34.71	-	-
T2G ch3	33.59	20.20	**14.21**	**10.40**	35.99	-	-	33.56	20.43	**14.35**	**10.50**	35.79	-	-
T2G	-	-	-	3.08	32.52	-	-	-	-	-	3.17	32.93	-	-
GS	**43.06**	**20.97**	10.48	5.39	**40.60**	**33.95**	**69.96**	**42.91**	**20.51**	9.86	4.95	40.63	34.13	69.95
S&R	**43.06**	19.46	8.88	4.14	39.40	33.63	69.48	**42.91**	**22.37**	11.77	6.35	**41.74**	**35.09**	**70.26**

Table 4.6: Baseline comparison results for Text-to-Gloss (T2G) translation on the MeineDGS Annotated (MeineDGS) dataset.

158, 173]. This represents a 4.41% and 27.86% improvement compared to the previous best. We find our approach outperforms a neural editing program [107], an RNN [173], and a basic transformer [158] on BLEU-1 to 2 and ROUGE scores on PHOENIX14**T**. While on MeineDGS, our approach outperforms a traditional transformer and Saunders et al. [158] on all metrics. However, compared to the results obtained from Chapter 3, the model struggled with gloss ordering, as indicated by the lower BLEU-4 scores. This could be for two reasons: firstly, the model could be overfitting to the order of a specific signer and by disentangling the two tasks; we are removing this signer-dependent information, as the GR task does not have access to the signed glosses. Secondly, the optimal performance from Chapter 3 was achieved by using sub-unit level tokenisers, which could be more effective at capturing the sign order, given their ability to reduce singletons, thus increasing repetitions in the data.

Comparing the results between the two datasets, we observe much lower BLEU-1 scores on the more challenging MeineDGS dataset, which is unsurprising given its larger vocabulary size. We observe a BLEU-1 reduction of approximately 17.07 when moving from the PHOENIX14**T** to the MeineDGS test set. However, when analysing the BERTScore, which uses contextual embeddings to calculate a score for the candidate sentence, we observe a much lower reduction in performance of approximately 8.75 on the test set. This indicates that even though lower scores are achieved on MeineDGS, the model may be generating valid paraphrases with similar semantic content.

As discussed, one of the main failure modes of the S&R approach is the reordering task, as evidenced by the BLEU-3 and 4 scores. However, this could be mitigated by including more linguistic features that were shown to be effective tools for reordering in the statistical approach. By adding this information into the deep learning approach, the reordering performance may

improve. We also note some failure cases with over-translation caused by the gloss selection network specifically. Given the architecture, it is encouraged to select the closest corresponding gloss for each word. As the spoken language sequence is generally much greater than the gloss sequence, this results in over-translation. This is particularly evident in the third example of Figure 4.6, where the model predicts 5 glosses for a spoken language sequence of 12 words, while ground truth has 2 glosses. This could be mitigated by adding an additional loss term to the GS model that penalises over-translation.

Model Latency

Table 4.7 demonstrates a significant advantages of our S&R model. It achieves an impressive 3.08x times speedup when compared to a traditional transformer architecture. Both GS and GR models utilise the same NAR decoder, but the incorporation of a reordering mask results in increased latency for the GR model. In contrast, our GS model, which has shown strong translation performance on its own, exhibits a large speed increase of 18.32x. Note that the speedup is proportional to the number of tokens in the output sequence, therefore greater savings could be achieved on larger datasets. These findings highlight the practical utility of our approach, particularly in computationally constrained scenarios.

Model	Latency	Speedup
Transformer	4380ms	1.00x
GS	239ms	18.32x
GR	1181ms	3.71x
S&R	1420ms	3.08x

Table 4.7: Inference latency comparison on MeineDGS.

4.3.3 Qualitative Evaluation

Figure 4.6 shows example translations from the MeineDGS test set. We compare our approach to the baseline transformer that achieved 31.12 BLEU-1 and 3.04 BLEU-4. We show the output from the S&R approach as well as the intermediate output from GS.

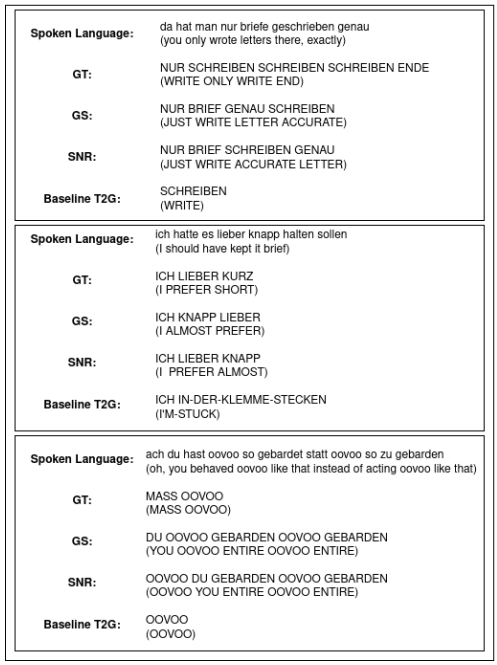


Figure 4.6: Example MeineDGS translations from a baseline transformer, the GS and S&R models

These translations show that our approach is better at retaining the meaning of the spoken language sentence, likely due to the 37.88% improvement in BLEU-1 score. However, in some cases, GS can over predict the number of tokens, especially for long input sequences as shown by the third example.

4.4 Conclusion

In this chapter, we presented Select and Reorder (S&R), a novel two-step approach to T2G translation, splitting the problem into two concurrent tasks of Gloss Selection (GS) and Gloss Reordering (GR). To split the translation in two, we first create an alignment between the source and target sequences. By leveraging the lexical overlap between the spoken language words and the signed glosses, we created a mapping between the two. This mapping is then used to reorder the glosses to match the spoken language order and vice versa. This approach disentangles the order from the vocabulary, allowing the GS model to focus on maximising the correct vocabulary while leaving the arguably more difficult ordering task to a separate model. We showed our proposed GS model achieves a significant increase in BLEU-1 score of 11.79 on the MeineDGS dataset. In addition, we showed that reordering can be learned by a GR model, but statistical-based methods perform more strongly with the current data limitations. Finally, we showed the results of combining GS with the mapping from a statistical reordering approach, finding that the S&R outperformed a neural editing program [107], an RNN [173], and a traditional transformer [158].

However, Gloss is only an intermediary representation for sign language and is intended for annotation only. The ultimate goal is to produce a photo-realistic sign language video. The literature shows there have been several proposed approaches to SLP, but they have mainly focused on manual features and have suffered from the problem of regression to the mean [216, 156, 153, 155]. This is a result of attempting to directly regress a pose sequence from either the spoken language or gloss sequence. Compounded by the natural variability seen in sign language, this results in under-articulated productions that lack stylistic cohesiveness. As shown here, the S&R approach can be used to improve the quality of gloss translation, and therefore the quality of the final sign language sequence. Next, we explore a full SLP pipeline, which is capable of leveraging the improved performance from S&R.

Chapter 5

Sign Stitching

Sign Language Production (SLP) is essential to facilitate two-way communication between the Deaf and Hearing communities. In Chapter 3 and 4, we explored the first stages of the SLP pipeline, converting spoken language to an intermediate sign language representation. In this chapter, we focus on the next stage of the pipeline, skeleton pose production. Specifically, we focus on the production of non-manual features and the comprehensibility of the final output. As these areas have been shown to be important aspects of SLP and have also been identified as shortcomings of current state-of-the-art systems [74, 155, 173, 210, 216]. The approach is designed to be compatible with the previous chapter's work, to help alleviate the T2G bottleneck and elevate the quality of the final output.

Sign language is inherently multi-channel, with channels performed asynchronously and categorised into manual (hands and body) and non-manual (facial, rhythm, stress and intonation) features. Analogous to the tone and rhythm used in spoken language, signed language exhibits prosody. The natural rhythm, stress and intonation that signed languages use to convey additional information [200]. Other phenomena, such as the impact of sentiment on the manual and non-manual features have also been studied, showing that emotion can change the form of a sign to exhibit distinctive profiles [148]. For sign language to be truly understandable, both manual and non-manual features must be present. To date, the majority of the research has focused on the manual features, with non-manual features often being overlooked [74, 155, 173, 210, 216].

Sign language corpora containing linguistic annotations are limited due to the cost and time required to create such annotations [101]. Previous works have attempted to directly regress a

sequence of skeleton poses from the spoken language or representations such as gloss [74, 76, 153, 157, 212]. However, given that sign language is a low-resource language and the complexity is under-represented in small datasets, previous approaches have suffered from regression to the mean [216, 156, 153, 155], resulting in under-articulated and incomprehensible signing.

In this chapter, we propose a novel approach to SLP that effectively stitches together dictionary examples to create a meaningful, continuous sequence of sign language. An overview of the approach can be seen in Figure 5.1.

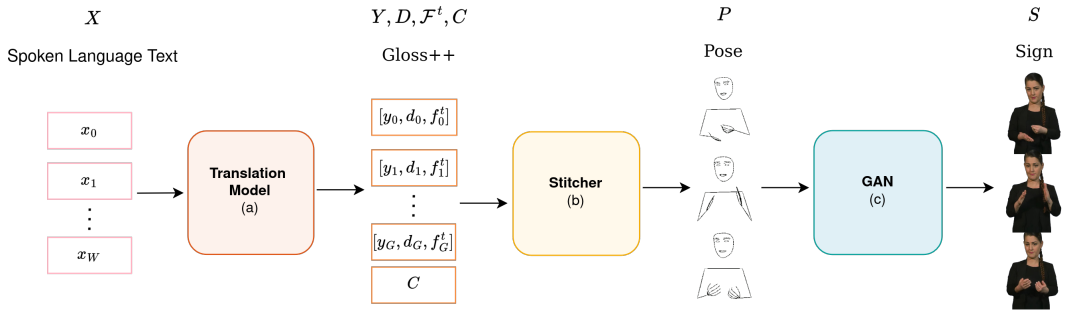


Figure 5.1: System overview of our approach to SLP. a) Spoken language to Gloss++ (gloss, duration, facial token and cutoff) Transformer. b) Stitcher that creates a skeleton pose sequence from Gloss++. c) The SignGAN module that produces a photo-realistic signer conditioned on a skeleton pose sequence.

By using isolated signs, we ensure the sequence remains expressive, overcoming previous shortcomings related to regression to the mean. However, each example lacks non-manual features, so we propose a NSVQ transformer architecture to learn a dictionary of facial expressions that can be added to each sign to create a realistic sequence.

By training a translation model to predict an enhanced gloss representation, named Gloss++, we can create a stitched sequence without robotic and unnatural movements. The Gloss++ representation contains the original gloss in addition to a sign duration, facial token and filter cutoff. Using this, we can resample each sign to the predicted duration, allowing us to alter the velocity associated with signing stress and rhythm [199]. The facial token is used to select a facial expression from the learnt dictionary, and the filter cutoff is used to adjust the trajectory of the sign, akin to how signers modify a sign to convey sentiment [17, 71]. Our approach

demonstrates that it is capable of modifying a stitched sequence to emulate aspects of prosody seen in the original continuous data.

The stitching module creates a continuous sequence of sign language, which is then passed to the SignGAN model, a GAN capable of generating photo-realistic sign language videos [154]. Thus, we present a full Text-to-Sign (T2S) SLP pipeline that contains both manual and non-manual features. The user evaluation agrees that the approach outperforms the baseline method [155] and improves the realism of the signed sequence. Given the effectiveness of the approach, we also leverage it as a data augmentation showing that stitched sequences can enhance the back-translation model.

Finally, when ground truth gloss timing is not available, such as for the PHOENIX14**T** dataset, we apply our stitching approach to perform sign segmentation. By creating an equivalent sequence and aligning it to the original data, we can infer the duration of each sign.

The rest of this chapter is structured as follows: Section 5.2 details the proposed approach following Figure 5.1 from left to right. After, we detail the curation process of Gloss++, explaining how the durations, cutoff points, and facial expressions are created. Section 5.3 describes the datasets used to construct the isolated dictionary and the experimental setup. Subsequently, quantitative and qualitative results are presented. Finally, Section 5.4 concludes the chapter.

5.1 Related Work

5.1.1 Signer Key Point Representation

Skeletal keypoints are a predefined set of landmarks on the human body used to represent human pose. Keypoints are commonly employed as a low-dimensional representation of the human body in machine learning and computer vision applications. They can be automatically extracted from an image using models such as OpenPose [27] or MediaPipe [115]. In this work, we use the MediaPipe model to extract keypoints from sign language videos. Specifically, MediaPipe Holistic is used to extract 21 keypoints for each hand, 19 for the body, and a 478-point face mesh. In subsequent work, we reduce the full face mesh to 128 keypoints for computational efficiency. This method allows us to overcome the need for expensive MoCap data.

While MediaPipe has become a standard for this task [115], it sometimes provides incorrect keypoint predictions, negatively impacting downstream tasks. Additionally, we find the 3D predictions less reliable. Thus, to enhance the quality of skeleton pose estimation and uplift the prediction to 3D, Ivashechkin et al. proposed a method that combines forward kinematics with a neural network to ensure valid predictions [77]. This optimisation solves for skeletal joint angles, J_a, which are used to check that the prediction conforms to the limitations of the human skeleton. We apply this Euclidean or angular representation in this current Chapter and Chapter 6 to represent the signer.

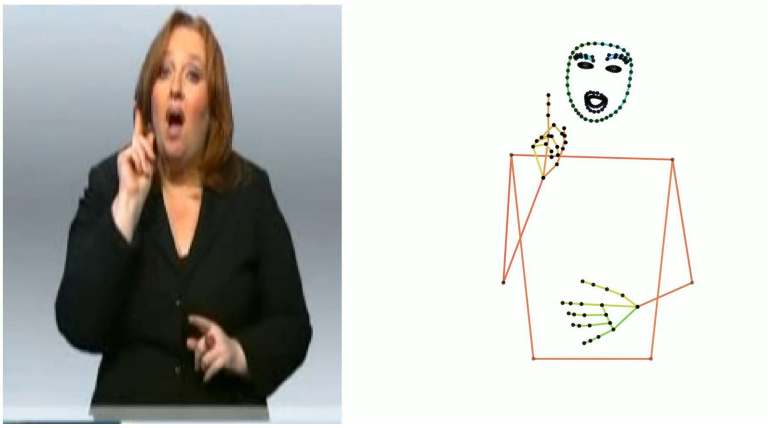


Figure 5.2: An example skeleton pose from the RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**) dataset. (left, orginal signer. Right, 178 keypoint pose representaion.)

5.2 Methodology

In Chapter 3 and 4, we explored translating to an intermediate representation of sign language. Now we extend this further to the translation from spoken to signed languages, by converting a source spoken language sequence, $X = (x_1, x_2, \ldots, x_W)$ with W words into a video of photo-realistic sign, denoted as $V = (v_1, v_2, \ldots, v_U)$ with U frames. To accomplish this, we use two intermediate representations, following Figure 5.1 from left to right. First, the spoken language is translated to Gloss++. An enhanced representation that contains a sequence of glosses, $Y = (y_1, y_2, \ldots, y_G)$, face tokens, $\mathcal{F}^{\mathrm{t}} = (f_1^t, f_2^t, \ldots, f_G^t)$ and durations, $D = (d_1, d_2, \ldots, d_G)$, all with length G. Additionally, for each sequence, we predict a low pass filter cutoff, $\mathcal{C}$ (Figure 5.1.a). Each gloss and facial expression is stitched together using these parameters, to produce a continuous sequence of poses, denoted as $P = (p_1, p_2, ..., p_U)$ with U frames (Figure 5.1.b). Finally, we use the skeleton pose sequence to condition the SignGAN module, allowing us to produce a photo-realistic signer. Next, we provide a detailed explanation of each step in our pipeline, and then we elaborate on the process of creating Gloss++.

5.2.1 Translation Model

Here we introduce the task of Text-to-Gloss++ (T2G++), given a spoken language sequence $X = (x_1, x_2, \ldots, x_W)$, our goal is to generate a corresponding sequence of Gloss++, which contains $Y, D, \mathcal{F}^{\mathrm{t}}, \mathcal{C}$. For this task, we design a transformer with four output layers, enabling the model to predict gloss plus the corresponding duration (in frames) and facial expression, in addition to a low-pass filter cutoff in Hz for each sequence. Thus, the model learns the conditional probability $p(Y, D, \mathcal{F}^{\mathrm{t}}, \mathcal{C}|X)$.

The model is an encoder-decoder transformer with MHA. The spoken language and gloss sequences are tokenised at the word level, and the embedding for a sequence is generated using a token embedding layer. Following the embedding layer, we add sine and cosine positional encoding.

The encoder learns to generate a contextualised representation for each token in the sequence. This representation is then fed into the decoder, which consists of multiple layers of self and cross MHA along with feedforward layers, and residual connections. The gloss, facial expression

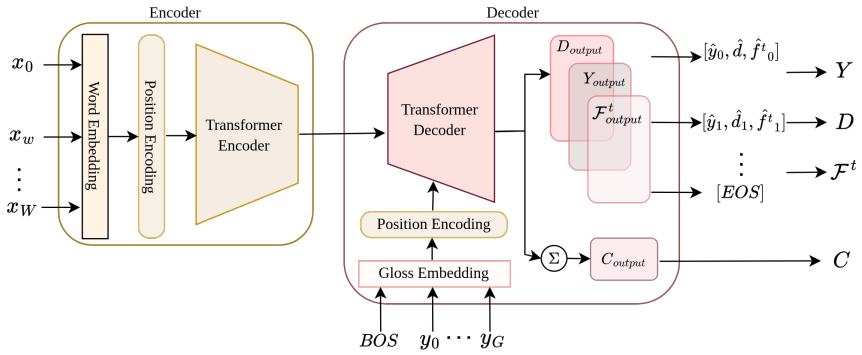


Figure 5.3: An overview of the Translation module.

and duration predictions are obtained by passing the decoder output through their respective output layers. To obtain the cutoff prediction, we pool the decoder embedding across each time step and pass the output through a linear layer. The model is trained end-to-end with the following loss function;

$$L_{\text{Total}} = -\lambda_{\text{y}} \sum_{i=1}^{Y} y_i^* \log(\hat{y}_i) - \lambda_{\text{f}} \sum_{i=1}^{\mathcal{F}^{\text{t}}} f_i^{t*} \log(\hat{f}_i^t) + \lambda_{\text{d}} \frac{1}{n} \sum_{i=1}^{n} (d_i^* - \hat{d}_i)^2 + \lambda_{\text{C}} (C^* - \hat{C})^2 \quad (5.1)$$

Each component is scaled by a factor, λ_{y}, λ_{d}, λ_{f} and λ_{C} before being combined to give the total loss, L_{Total}. The predictions from this model are passed to the stitching module to generate a skeleton pose sequence.

5.2.2 Stitching

For each dataset, we collect an isolated instance of each gloss in our target vocabulary. For each sign, we extract Mediapipe skeletons [115] and run an additional optimisation to uplift the 2D skeletons to 3D [77]. The optimisation uses forward kinematics and a neural network to solve for joint angles, J_a. We choose to store our dictionary as joint angles, as this allows us to apply a canonical skeleton. This ensures the stitched sequence is consistent even if the original signers have different bone lengths. We define a dictionary of, N_s, signs as $DS = [S_1, S_2, ..., S_{N_s}]$ where each sign in the dictionary consists of a sequence of angles, such that $S_i = (a_1, a_2, ..., a_{U_{\text{s}}})$ and $a_i \in \mathbb{R}^{J_a}$, where U_{s} is the duration in frames. In addition, we define a learnt dictionary of, N_f, facial expressions as $DF = [F_1, F_2, ..., F_{N_f}]$, where $F_i \in \mathbb{R}^{U_{\text{f}} \times J \times \mathcal{D}}$. As illustrated in Figure 5.4, the stitching pipeline is comprised of seven steps, we now detail each step.

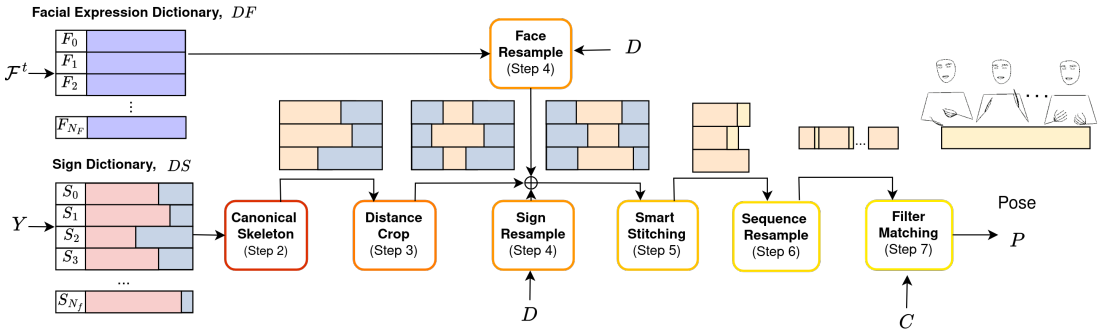


Figure 5.4: An overview of the stitching module.

Step 1) Given a list of glosses, Y, the corresponding signs are selected from the dictionary. If a gloss is absent, it is initially lemmatised and formatted. If a match is not found in the dictionary, a word embedding model is applied to compute the cosine similarity with all words in the dictionary. The closest sign is then selected as the substitute. Such that:

$$j_{sub} = \arg\max_{j}\left(\frac{\sum_{j=1}^{N_s} \varepsilon(y_{\mathrm{q}}) \cdot \varepsilon(DS^{\mathrm{y}}{}_j)}{\sqrt{\varepsilon(y_{\mathrm{q}})^2} \cdot \sqrt{\sum_{j=1}^{N_s} \varepsilon(DS^{\mathrm{y}}{}_j)^2}}\right) \qquad S = DS[j_{sub}] \tag{5.2}$$

Here $\varepsilon()$ represents the word embedding model, y_{q} is the query gloss and DS^{y} is the dictionary's corresponding gloss tags. We find word embeddings capture the meaning of words, enabling substitutions such as replacing RUHRGEBIET (RUHR AREA) with LANDSCHAFT (LANDSCAPE). Simultaneously, in this step, we select the corresponding facial expressions, F, from the dictionary, DF, given the predicted face tokens, $\mathcal{F}^{\mathrm{t}}$.

Step 2) The selected signs are converted from angles into a 3D canonical pose. We normalise the rotation of the signer, such that the midpoint of the hips is located at the origin and the shoulders are fixed on the xy-plane. This ensures the skeleton is consistent across all signs. Consequently, we convert from a sequence of angles $S_n \in \mathbb{R}^{U_{\mathrm{S}} \times J_a}$ to a sequence of skeleton poses, $P = (p_1, p_2, ..., p_{U_{\mathrm{S}}})$ with the same number of frames, U_{S}. Each skeleton pose, p_u, is represented in $\mathcal{D}$-dimensional space and consists of J joints, denoted as $p_u \in \mathbb{R}^{J \times \mathcal{D}}$.

Step 3) The dictionary signs often start and end from a rest pose. Therefore, to avoid unnatural transitions, we crop the beginning and end of each sign. For this, we track the keypoint $\mathcal{T}$ corresponding to the wrist of the signer's dominant hand and measure the distance travelled.

Thus, for each dictionary sign we create a sequence, $P^{\Delta} = (p_2^{\Delta}, p_3^{\Delta}, \ldots, p_{U_\text{S}}^{\Delta})$ $n \in 2, 3, \ldots, U_\text{S}$, representing the distance travelled for a dictionary sign. We remove the beginning frames once the sign has moved by a set threshold, α_{crop}. The crop index is given by:

$$index_\text{start} = \arg\max_u \left(\sum_{u=1}^{U_\text{S}} P_u^{\Delta} - \alpha_{crop} \cdot \sum_{u=1}^{\max(U_\text{S})} P_u^{\Delta} \right) \tag{5.3}$$

To crop the end, we reverse the order of the frames and repeat the process. However, for short sequences, this method may over-crop the sign, rendering it meaningless. We introduce an additional constraint $index_{end} - index_{start} < U_{min}$. If this condition holds, we reduce the threshold, α_{crop}, by a factor of 0.1 and recalculate.

Step 4) As detailed above, a duration is predicted for each gloss. Here, we utilise the duration to resample the length of each sign and facial expression, emulating the natural rhythm in the original data. This process involves upsampling or downsampling the sign using linear interpolation. However, due to co-articulation in the continuous data, some signs are predicted to be only a few frames long, which may not be sufficient for an isolated example. To address this, we impose a minimum sign length to ensure the produced sequence remains expressive. Once the facial expression and sign are resampled to the same length, we shift and rotate the face onto the body creating the complete skeleton.

Step 5) Having created a list of signs in a canonical pose, cropped, and resampled to match the original data, the next step involves joining these signs into a single sequence using the smart stitching module. The objective is to achieve a natural transition between the end of one sign and the start of the next. To accomplish this, we track the dominant hand of the signer and calculate the distance, Δ, between the end of the first sign and the start of the second sign. Then we can determine the required number of frames, U_stitch, needed to create a smooth transition. Such that:

$$U_\text{stitch} = \arg\min_u (V_\text{min} < \frac{\text{FPS} * \Delta}{u} < V_\text{max}) \tag{5.4}$$

Where V_max and V_min are the start and end velocities of the two signs. This calculation ensures that the signer's velocity is bounded by the end velocity of sign one and the initial velocity of sign two. In cases where multiple solutions exist, we select U_stitch that minimises the standard deviation between the start and end velocity. Following this, we employ linear interpolation to generate the missing frames.

Step 6) We concatenate the signs and the stitched frames to form a single sequence. We then sum all the predicted durations and resample the sequence so it matches the ground truth.

Step 7) Finally, we apply a low-pass Butterworth filter to each keypoint over time [24]. The predicted cutoff value determines which frequencies are removed and corresponds to the -3dB attenuation point. This step aims to enhance the stylistic cohesiveness of the sequence by smoothing out any sharp, quick movements not present in the original sequence. The transfer function can be formulated as;

$$H(z) = \frac{1}{1 + \left(\frac{\mathcal{Z}}{e^{j \cdot \omega_c}}\right)^{2O}} \tag{5.5}$$

Here we apply a 4th order filter thus, O is 4, and the angular cutoff frequency is given by $\omega_c = 2\pi C$. $\mathcal{Z}$ corresponds to the z-transform of the skeleton pose sequence. This process generates natural skeleton sequences, that remain expressive and stylistically cohesive. Next, we map these skeleton poses to a photo-realistic signer.

5.2.3 SignGAN

Skeleton outputs have been shown to reduce Deaf comprehension compared to a photo-realistic signer [190]. Therefore, to gain feedback from the Deaf community we train a SignGAN model [154].

Given a skeleton pose sequence, $P = (p_1, p_2, ..., p_U)$, generated by our stitching approach the model aims to generate the corresponding video of sign language, $V = (v_1, v_2, \ldots, v_U)$ with U frames. Each video frame is conditioned on the appearance of an individual extracted from a style image. For the user evaluation, we use the appearance of a single individual obtained by passing a style image through the style encoder.

5.2.4 Gloss++ Generation

Here we detail how each component of Gloss++ is generated.

Facial Expression

To be truly understandable and accepted by the Deaf community, non-manual features must be present in the final output. We use a discrete sequence-to-sequence model to generate the translation. Therefore, we must learn a discrete vocabulary of facial expressions, DF, that can be added to the isolated signs. As depicted in Figure 5.5 We apply a NSVQ to learn a spatial-temporal dictionary of facial expressions.

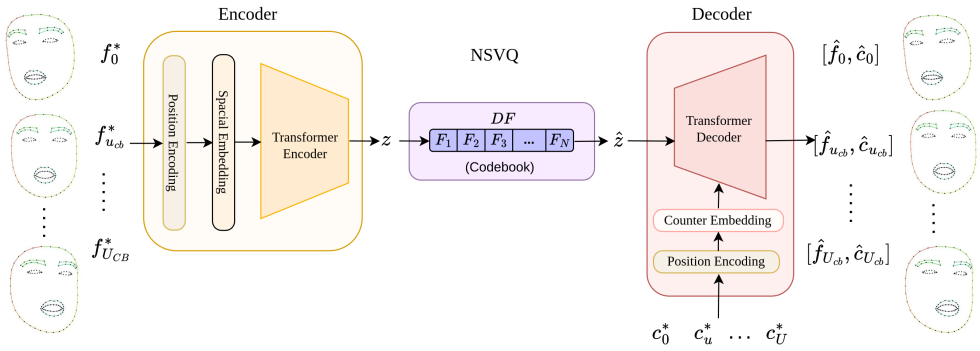


Figure 5.5: The Face NSVQ transformer architecture.

For each gloss in the original data, we extract the corresponding face mesh, denoted as F. We then resample each sequence to a constant length, U_{f} and scale it to be a constant size. Signers in the dataset are often looking off-centre, therefore, we normalise the average direction of the face so that it is looking directly forward. Similar to Figure 5.3 (Encoder), we add positional encoding and then embed the sequence using a single linear layer. After the embedding is passed through the transformer encoder to the codebook. The NSVQ codebook learns a set of N_f embeddings. We denote the embedded face sequence and therefore each codebook entry as $F_i^z \in \mathbb{R}^{U_{\mathrm{f}} \times E}$, where E is the embedding dimension. Each input is mapped to one codebook entry, the difference between the selected codebook entry and the input is then simulated using a normally distributed noise source. The mathematical product of the simulated noise and the encoder output is then passed to the decoder. We use the counter decoding technique from the Progressive Transformer (PT) [155], to drive the decoder. The decoder learns to reconstruct the original face sequence and the input counter values. Thus, the model is trained with the

following loss function;

$$L_{Face} = \frac{1}{U_{\mathrm{f}}} \sum_{u=1}^{U_{\mathrm{f}}} ((f_u^* - \hat{f}_u)^2 + \lambda_{\mathrm{CN}}(c_u^* - \hat{c_u})^2) \tag{5.6}$$

Where λ_{CN} is a scaling factor and c is the counter value. Once fully trained, we pass each codebook embedding, F_i^z, through the decoder to give the learnt dictionary of facial expressions in Euclidean space, $DF = [F_1, F_2, ..., F_{N_f}]$.

Duration

When ground truth timing information is not available, we propose a novel sign segmentation approach based on the stitching method described above. Given the ground truth gloss labels, we generate the stitched sequence, P_{stitched}, but without step 4 (sign resampling).

Comparing the stitch sequence with the ground truth, we find that the motion can vary due to different lexical variants not found in the dictionary. However, we find that the handshape is often still consistent. So, we take the keypoints that correspond to the signer's hands and normalise the rotation so that the index finger metacarpal bone is fixed on the y-axis and the palm is fixed on the xy-plane, giving P^H_{stitch}, P^H. Our next step is to align the two sequences so that we can infer the duration of the signs in the ground truth. For this we apply DTW, such that;

$$A^{DTW}_{i,j} = DTW(P^H_{stitch}, P^H) \tag{5.7}$$

As we know the duration of the isolated signs in the stitched sequence, by analysing the alignment path, A^{DTW}_j, we can infer the duration of the signs in the original ground truth sequence.

Cutoff

Experiments reveal that each sequence contains a distinct range of frequencies correlated to the signer's style. Fast motions contain high frequencies, while soft, slow signing involves lower frequencies, typically within the range of 1 to 25 Hz.

To generate the ground truth cutoffs used in training, we once again apply our stitching approach. For each sequence in the ground truth data, P, we produce the equivalent stitched sequence, P_{stitched}. We then apply a low-pass filter to P_{stitched}within the range of 1 to 25 Hz and measure

the intersection and set difference of the frequencies, denoted as $(FT(P) \cap FT(P_{\text{stitched}}))$ and $(FT(P) \setminus FT(P_{\text{stitched}}))$, where the Fourier transform is represented as $FT()$. Subsequently, we fit a parametric spline curve to the intersection and set difference. To determine the cutoff, we find the frequency that maximises the intersection while minimising the set difference. This provides the cutoff frequency for that sequence. We opt to use this method over analysing the frequency in the original sequence because an ideal filter is unavailable, and therefore the Butterworth filter introduces unintended effects on the frequency and phase below the cutoff.

5.3 Experiments

5.3.1 Implementation Details

In our experiments, we conducted a grid search for optimal hyperparameters and identified the following settings as the most effective. Our encoder-decoder translation model is constructed with an embedding size of 512 and a feed-forward size of 1024. We find that the optimal number of layers and heads is dataset-dependent, with smaller datasets requiring fewer layers compared to MeineDGS where we use 3 layers and 4 heads. The models utilise dropout with a probability of 0.1 [171], ReLU activations between layers [1], and pre-layer normalisation for regularisation and training stability. Training employs a 'reduce on plateau' scheduler with a patience of 5 and a decrease factor of 0.8. The layers are initialised using a Xavier initialiser [62] with zero bias, and during training, Adam optimisation is employed [95]. The initial learning rate is set to 10^{-4}, and we train the model until convergence. During decoding, we utilise a greedy search algorithm. The loss scaling factors, λ_{y}, λ_{d}, λ_{f} and λ_{C} are set to 1.0, 0.1, 0.3 and 0.2, respectively. When stitching, we enforce a minimum sign length, U_{min} of 4 frames. The cropping threshold, α_{crop} is dataset-dependent, we find values between the range of 0.1 to 0.35 most effective. All sequences are subsampled to 12 frames per second (fps) for computational efficiency.

For each dataset, we create a dictionary of 500 facial expressions. We scale the counter loss by 0.01, and we set an embedding dimension of 512. The encoder and decoder are initialised with the same settings as our translation model.

For comparison, we train a progressive transformer on each dataset until convergence using the parameters from [155].

Dictionary

The approach is tested on three datasets, the Public Corpus of German Sign Language, 3rd release, the MeineDGS dataset [101], PHOENIX14**T** [25] and the BSLCP**T** [160]. For each dataset we test two different dictionaries: 1) collected from isolated examples, and 2) a dictionary created from continuous data. Next, we provide further details about each:

Isolated (ISO): Here, the signs are sourced from individuals who perform each sign in isolation, typically starting from and returning to a resting position. When experimenting on the BSLCP**T**

we use the Signbank dataset [36], which contains over 3,000 signs and includes all the lexical variants found in the BSLCP**T** dataset. However, no such dictionary exists for the PHOENIX14**T** and MeineDGS dataset, therefore we collect a dictionary from a range of sources such as [101]. The MeineDGS dataset has a target vocabulary of 10,801. However, without the gloss variants, the core gloss vocabulary reduces to 4,434. We collect a total of 7,206 signs to experiment with. We use the method described in Section 5.2 (Stitching), step 1 to overcome issues with an incomplete vocabulary. To create word embeddings we apply Facebook's implementation of Fasttext [120]. When experimenting on the BSLCP**T** we use the English implementation whereas on PHOENIX14**T** and MeineDGS we use the German implementation.

Continuous (CON): Here we create a dictionary using the gloss timing annotations. The signs are taken from the test and dev data only, so that the back-translation model has not seen the signs during training. As the examples come from continuous sign, we omit the cropping step of the stitching pipeline. These dictionaries have an abundance of signs to choose from when stitching. We filter the dictionary and remove short signs as these are most likely co-articulated and therefore not suitable out of context. We randomly select the first sign in the sequence, and subsequent signs are chosen to ensure the most natural transition. Therefore, we select the sign from the dictionary in which the location of the wrist is closest to the last frame of the previous sign.

5.3.2 Quantitative Evaluation

Text-to-Gloss++ Translation Results

We start by evaluating the T2G++ translation performance described in Section 5.2.1. Table 5.1 shows the performance on all three datasets. We observe that the difficulty of a dataset is proportional to the vocabulary and the total number of sequences used in training. We find the best performance on the PHOENIX14**T** dataset which has the highest number of sequences per token, achieving 18.11 BLEU-4. In comparison to baseline methods (Saunders et al. [155] and Stoll et al. [173]), we observe that having the model predict duration, face and cutoff we can achieve higher BLEU-1 scores on the dev set, but at the cost of a lower BLEU-4. However, in comparison to Chapter 3 and 4, the model falls short of more advanced T2G approaches.

	TEST SET					DEV SET				
Dataset:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
BSLCP**T**	26.02	11.19	4.15	1.67	25.06	26.88	11.40	4.72	1.28	26.96
MeineDGS	30.13	13.04	5.45	2.36	31.43	29.72	12.46	4.89	1.87	30.90
MeineDGS Chapter 3	33.56	20.43	14.35	10.50	35.79	33.59	20.20	14.21	10.40	35.99
MeineDGS Chapter 4	42.91	22.37	11.77	6.35	41.74	43.06	19.46	8.88	4.14	39.40
PHOENIX14**T**	55.48	36.54	25.18	18.11	53.83	56.55	37.32	25.85	18.74	54.81
PHOENIX14**T** [155]	55.18	37.10	26.24	19.10	54.55	55.65	38.21	27.36	20.23	55.41
PHOENIX14**T** [173]	50.67	32.25	21.54	15.26	48.10	50.15	32.47	22.30	16.34	48.42
PHOENIX14**T** Chapter 3	60.04	42.85	32.18	25.09	58.82	58.74	40.86	30.24	23.19	56.55
PHOENIX14**T** Chapter 4	62.69	40.01	27.07	19.07	56.83	60.13	35.10	22.65	15.60	53.78

Table 5.1: The results of translating from Text-to-Gloss++ on the BSL Corpus **T** (BSLCP**T**), RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**) and MeineDGS Annotated (MeineDGS) dataset.

The results show significantly lower BLEU-4 scores on the MeineDGS dataset, which can be attributed to its larger domain of discourse. In contrast, the BSLCP**T** dataset, despite its more constrained domain (evidenced by a smaller target vocabulary size), demonstrates the lowest sequence-to-token ratio among all the tested datasets. This characteristic results in a BLEU-4 score of 1.67 on the test set. Overall, the model's performance aligns with our previous findings given the architecture.

Text-to-Pose Translation Results

It should be noted that in the following experiments, the back-translation model's performance (shown as GT top row of Table 5.2, 5.3 and 5.4) should be considered the upper limit of performance. In this section, we evaluate the T2P performance using back-translation. To allow for a comparison we train two versions of the PT with the setting presented in [155]. PT is the standard architecture, while PT + GN is trained with Gaussian Noise added to the input. In line with the original paper, we find that Gaussian Noise improves the performance. However, our approach still outperforms both models except on DTW-MJE. As discussed previously, other works suffer from regression to the mean [216, 156, 153, 155] caused by models attempting to minimise their loss function and thus are incentivised to predict a mean pose. This metric fails to evaluate the content of the sequence, but the higher score does indicate that our model is expressive as it is producing sequences further from the mean. For back-translation, we

BSLCP**T**	TEST SET						DEV SET					
Approach:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
GT	0.000	17.3	3.96	1.37	0.54	21.76	0.000	17.32	3.71	1.08	0.39	21.89
PT [155]	**0.288**	4.40	0.65	0.18	0.00	8.22	**0.292**	4.00	0.61	0.10	0.00	8.02
PT + GN [155]	0.267	4.96	0.55	0.13	0.00	8.82	0.258	4.47	0.63	0.09	0.00	8.89
Stitcher (ISO)	0.588	16.37	2.86	0.75	0.28	**20.84**	0.592	16.39	2.82	0.58	0.00	19.55
Stitcher (CON)	0.575	**16.99**	**3.65**	**1.03**	**0.41**	20.65	0.573	**16.52**	**3.19**	**0.73**	0.00	**20.53**

Table 5.2: The results of translating from Text-to-Pose on the BSL Corpus **T** dataset.

MeineDGS	TEST SET						DEV SET					
Approach:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
GT	0.000	20.87	5.60	1.89	0.80	23.78	0.000	20.75	5.43	1.81	0.76	23.41
PT [155]	**0.229**	6.11	0.94	0.21	0.05	8.36	**0.228**	6.22	0.98	0.17	0.00	8.44
PT + GN [155]	0.225	7.18	1.48	0.40	0.01	8.38	0.224	9.22	1.63	0.38	0.01	8.57
Stitcher (ISO)	0.581	16.63	3.75	0.94	0.22	21.69	0.592	16.9	3.67	0.95	0.32	**21.34**
Stitcher (CON)	0.637	**18.64**	**4.17**	**1.07**	**0.39**	**21.80**	0.637	**18.27**	**4.07**	**1.19**	**0.43**	21.25

Table 5.3: The results of translating from Text-to-Pose on the MeineDGS Annotated (MeineDGS) dataset.

outperform the PT on all metrics. Showing significant improvements in BLEU-1 score of 98% and 269% on the MeineDGS and BSLCP**T** dev set (comparing PT + GN and Stitcher (continuous), Table 5.2 and 5.3).

Deep learning models exhibit a bias toward the data they were trained on and often show poor out-of-domain performance. Unsurprisingly, the performance improves when using a continuous dictionary. We find only a small increase in BLEU-1 of 0.13 on the BSLCP**T** dev set (Table 5.2), most likely due to the isolated dictionary containing the lexical variants found in the original data. Whereas we see a larger increase on the MeineDGS dataset (Table 5.3).

Previous work has primarily focused on G2P translation. Therefore, to facilitate a meaningful comparison we present two versions of the model. First, a G2P version, where we use the ground truth data and just apply the stitching module, and, secondly, our T2P approach (translation then stitching). Results for comparison are provided by Xie et al. [212]. We find our approach outperforms previous work on the BLEU-1 to 2 scores increasing the score by 56% and 19%, respectively (comparing Table 5.4, row 7 and 11). We also find significant improvement in the ROUGE metric.

PHOENIX14**T** Approach:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
GT	0.000	32.41	20.19	14.41	11.32	32.96
PT [155]	0.197	6.27	3.33	2.14	1.59	9.50
PT + GN [155]	0.191	11.45	7.08	5.08	4.04	14.52
NAT-AT [74]	0.177	14.26	9.93	7.11	5.53	18.72
NAT-EA [74]	0.146	15.12	10.45	7.99	6.66	19.43
PoseVQ-MP [212]	0.146	15.43	10.69	8.26	6.98	-
PoseVQ-DDM [212]	0.116	16.11	11.37	**9.22**	**7.50**	-
Stitching G2P (ISO)	0.593	21.47	8.79	4.25	2.49	20.32
Stitching G2P (CON)	0.587	23.58	12.31	8.05	5.95	24.84
Stitching T2P (ISO)	0.594	22.78	9.68	5.17	3.12	21.30
Stitching T2P (CON)	0.572	**25.14**	**13.54**	8.98	6.67	**26.49**

Table 5.4: The results of translating from Gloss-to-Pose (G2P) and Text-to-Pose (T2P) on the RWTH-PHOENIX-Weather-2014**T** dataset.

The hypothesis that translating from G2P would be easier due to the close correspondence between gloss tokens and pose sequence proved incorrect. The superior performance of the T2P task suggests that gloss, despite being a convenient-written form for sign language, fails to capture the nuances necessary for accurate skeleton pose generation. For example, given the following gloss sequence from the BSLCP**T**, "NEXT-YEAR TAIPEI", and the corresponding spoken language translation, "It's in Taiwan next year.". There is a clear discrepancy between the two, with the spoken translation using the country instead of the specific city. Given that the evaluation is performed on the spoken language equivalent, we suggest that providing the model with access to the context and additional textual detail could enhance performance. Thus, the model generates a sequence which aligns closer to the spoken language translation rather than the gloss notation.

Using a continuous dictionary, we can outperform all models except VQ based approaches on BLEU-3 to 4. As the VQ model is learning sub-units of a gloss sequence, we suggest this gives it an advantage for higher order n-grams, as each token can represent multiple signs.

Segmentation Results

To evaluate the stitching segmentation approach, we calculate the duration for each gloss in the MeineDGS dataset test set. We achieve a sign level frame F1-score of 0.6373 a similar score compared to [123] that achieves a top score of 0.63. Demonstrating the effectiveness of stitching

for sign segmentation. It is worth noting that our approach requires gloss information, but is computationally inexpensive compared to the LSTM used in [123].

Data Augmentation Results

The stitching approach has been shown to be effective at producing high quality output, as evidenced by the translation scores. The final quantitative evaluation tests to see if the proposed approach can be used as a data augmentation strategy to generate pseudo skeleton data. As sign language is a low-resource language, SLT models, such as the back-translation model we use for evaluation, have struggled to generalise to unseen data. Using the Stitching approach, we recreate the PHOENIX14**T** datasets using the gloss annotation, the curated timing information, and either the isolated (ISO) or continuous (CON) dictionary. We train a back-translation model using the stitched data and the ground truth (GT). We can alter the parameters of the stitching pipeline to generate greater variation in the skeleton production, which in turn enhances the back-translation model, as shown in Table 5.5.

All results are tested on the test and dev splits from the PHOENIX14**T** dataset. From the results, we make the following observations.

1. Jointly training on the stitched data reduces the model's performance on the original data (Table 5.5a)
2. Pre-training the model on the stitched data and then fine-tuning on the ground truth improves the model's ability to generalise to unseen data (Table 5.5b). But only when created with an isolated dictionary. We assume this is due to several signs in the continuous dictionary being co-articulated, and therefore not suitable when used out of context.
3. By creating larger variations in the sign ordering, we can improve the model's ability to recognise longer sequences of signs, shown by the higher BLEU-2 to 4 scores. However, at the cost of a reduced BLEU-1 (Table 5.5c).
4. Finally, we observe that the greatest augmentation strategy is to create more temporal variations in the signing. Suggesting the model is likely overfitting to the timing information in the training data. We see the greatest improvement in the BLEU-1 to 4 scores (comparing Table 5.5a GT and scale 1.5 d) of 4.58, 3.64, 2.64 and 1.95, respectively.

	TEST SET					DEV SET				
Trained On:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
GT	**32.41**	**20.19**	**14.41**	**11.32**	**32.96**	**32.36**	**20.02**	14.25	11.13	**33.46**
Stitched	16.74	6.88	4.53	3.49	17.72	16.78	7.67	5.29	4.21	18.34
GT + Stitched	31.84	19.45	13.71	10.73	32.04	31.19	19.63	**14.26**	**11.37**	32.57

a) Models jointly trained on ground truth (GT) data and synthetic stitched data.

	TEST SET					DEV SET				
Pre-Trained On:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
ISO Data	**37.86**	**24.36**	**17.58**	**13.68**	**37.61**	**37.26**	**23.71**	**17.13**	**13.52**	**37.95**
CON Data	28.84	15.97	10.98	8.35	28.75	28.12	15.66	10.58	7.92	28.70
ISO + CON Data	36.33	22.99	16.46	12.76	36.26	36.25	23.10	16.67	13.07	36.75

b) Models pre-trained on stitched data generated using isolated (ISO) and continuous (CON) sign language dictionaries.

	TEST SET					DEV SET				
Permutations:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
0	**37.86**	**24.36**	17.58	13.68	37.61	**37.26**	23.71	17.13	13.52	37.95
1	36.22	22.84	16.22	12.65	36.09	36.26	22.98	16.45	12.76	36.49
3	37.04	24.22	**17.69**	**13.89**	**37.83**	37.18	**24.07**	**17.57**	**13.81**	**38.00**
10	37.67	24.18	17.23	13.24	37.36	36.47	23.26	16.83	13.16	36.91

c) Model pre-trained on stitched data generated using the isolated (ISO) dictionary with N random permutations of sign order.

	TEST SET					DEV SET				
Duration scale:	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
0.5	35.75	22.49	15.84	12.27	35.94	34.28	21.05	14.89	11.53	34.94
0.7	37.23	24.65	18.09	14.29	38.39	37.67	**25.43**	**19.17**	**15.38**	39.05
0.9	37.09	23.65	16.82	12.98	37.09	36.88	23.39	16.83	13.11	36.72
1.1	36.24	23.31	16.94	13.27	36.55	36.49	23.33	16.88	13.13	36.84
1.3	37.48	23.658	16.80	12.97	36.78	36.21	22.81	16.40	12.69	36.45
1.5	**38.61**	**25.66**	**18.75**	**14.71**	**38.91**	**38.28**	24.97	18.13	14.08	**39.07**
0.7 + 1.0 + 1.5	37.60	23.64	16.56	12.65	36.89	36.67	23.1	16.66	13.02	36.92

d) Model pre-trained on stitched data generated using the isolated (ISO) dictionary, with variations in the duration of each sign. Scaled by a constant factor, N.

Table 5.5: The results of using sign stitching for data augmentation on PHIX.

5.3.3 Qualitative Evaluation

Video Outputs

To demonstrate the approach's effectiveness, we present skeleton and video outputs for two sign languages (BSL and DGS)[1]. To enable a fair evaluation, we also share failure cases where incorrect predictions from the initial translation produce repetitive sequences.

Stitching Example

Here we provide additional figures to show the productions from the sign stitching approach. Using the ground truth gloss labels and timings, we can accurately recreate the original sequence. As discussed, on the BSL Corpus **T** dataset, we use the SignBank dataset as our dictionary, which has gloss variant labels. Allowing us to select the same form of each sign. Note, for clarity, we plot a reduced set of facial keypoints.

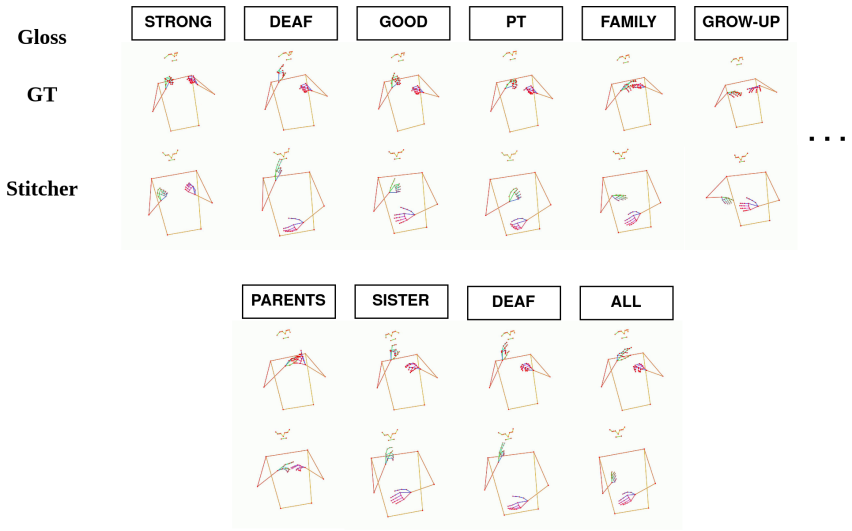


Figure 5.6: An example of the sign stitching approach on the BSL Corpus **T**.

[1] https://github.com/walsharry/Representations-for-Sign-Language-Production/blob/master/ch5/README.md

Translation Examples

Here we share a translation example. Showing each step of the pipeline from Text-to-Gloss (T2G) (rows 1 and 2), then Gloss-to-Pose (T2P) (rows 2 and 6) and finally Pose-to-Sign (P2S) (rows 6 and 7). The approach is able to recreate the sequence with a small variation in the sign style due to the different forms of the signs in our dictionary.

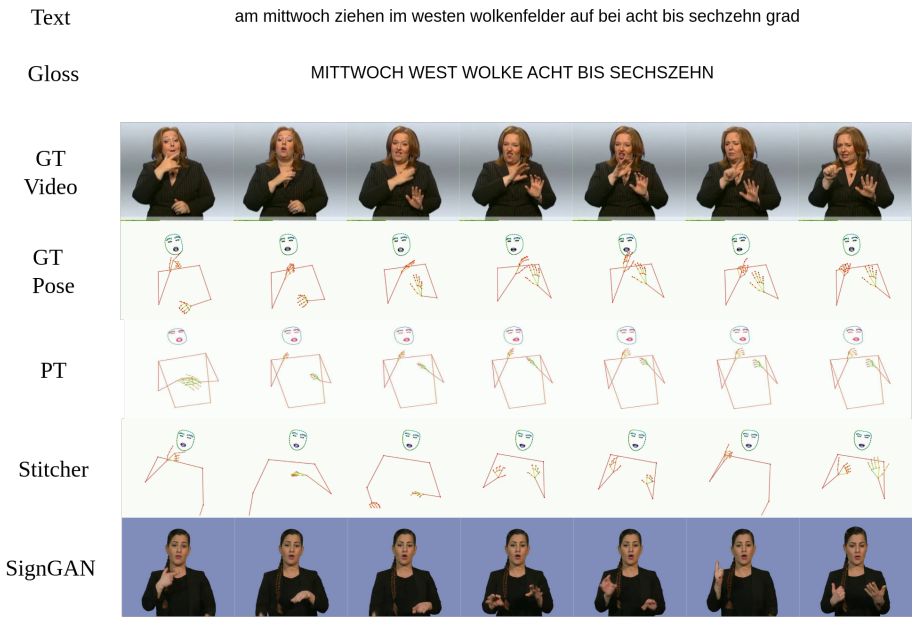


Figure 5.7: A Text-to-Sign (T2S) translation example on the RWTH-PHOENIX-Weather-2014**T** dataset. Showing the original spoken language (Text), the corresponding glosses (Gloss), the original video (GT Video), the ground truth skeleton pose (GT pose), the output from the Progressive Transformer (PT), the produced stitched sequence (Stitcher), and the output from SignGAN.

Note in columns 2 and 3, the approach is using a different form of the Sign ‘WEST’, hence the different motion but the same handshape.

Survey Results

The survey was composed of 10 questions. The first four questions involved a comparison to the PT, followed by six questions presenting an ablation of different components of the stitching approach. For each question, participants were presented with two videos and asked ”Please

compare the A and B videos with respect to the quality of the overall motion and the hands. “. Then selecting one of three options: A, B, or no preference.

In total, 12 participants completed the survey. The participant group consisted of 16.7% native signers, 33.3% L2 BSL learners, 16.7% Level 1 BSL learners, and 33.3% non-signers.

Regarding the results, 87.5% of the participants preferred our approach over the PT, with the remaining participants expressing no preference. All participants (100%) agreed that applying the filter improved realism compared to unfiltered sequences. The resampling component was found to be less critical, as 37.5% of participants expressed no preference between resampled and normal sequences. An interesting finding was the preference for the isolated dictionary, with 41.7% of participants favouring its use, while 20.8% preferred the continuous dictionary, and the remainder had no preference.

5.4 Conclusion

In this chapter, we presented a novel approach to SLP. Previous research has been limited by the problem of regression to the mean [159] and has predominantly focused on only manual features. These limitations are addressed here by employing a dictionary of expressive examples that can be stitched together, creating a natural, continuous sequence. Furthermore, by clustering facial expressions into a vocabulary, we can create a sequence that contains both manual and non-manual features. The inclusion of both channels ensures that the generated sequences accurately capture both hand movements and facial expressions, which are essential for conveying meaning in sign language.

The Gloss++ representation and the stitching process are the key innovations of this approach. By carefully cropping and joining individual signs, we create a fluid and natural sequence that mimics the way sign language is used in real life. This method addresses the issue of abrupt transitions between signs, which was common for older approaches where the motion appears disjointed and robotic [8, 38]. This approach smooths the transitions, resulting in a more cohesive and realistic output. Consequently, we produced sign language sequences that can be used to enhance the performance of an SLT model. By creating temporal and sign order variations in the generated pseudo-data used to pre-train an SLT model, we can improve the performance of these models and achieve better translation accuracy.

The user evaluation further validates our approach, with a majority of participants preferring our method over existing techniques. This feedback highlights the importance of incorporating both manual and non-manual features in SLP and demonstrates the effectiveness of our stitching process. In conclusion, our novel approach to SLP addresses key challenges in the field and offers a robust solution for generating realistic and expressive sign language sequences.

On MeineDGS which has the largest domain of discourse, the performance is significantly lower in comparison to the other datasets. This highlights the need for larger datasets and a move towards unconstrained translation. Such a dataset would need to be orders of magnitude greater than currently available, possibly comparable to those used to train LLMs, such as gpt-3, that used a 300 billion token dataset [21]. However, curating such a dataset is a time-consuming and expensive process, primarily due to the cost of gloss annotation. Therefore, for the field to progress, it is essential to develop methods which break the reliance on this linguistic annotation.

However, there is still a need for a low dimensional representation, as RGB video is costly to train on directly. In addition, directly attempting to regress a skeleton pose sequence results in regression to the mean and thus, under-articulated signing. Hence, there is a need to develop a new representation that can be used as a substitute to gloss. In the next chapter, we explore a method for automatically generating new representations. That would allow SLP to scale to new languages and datasets, without the need for manual annotations.

Chapter 6

Vector Quantised Sign Language Production

Sign language is inherently multi-channel, involving both manual and non-manual articulators [188]. A sequence of signing can be decomposed into a series of cheremes, analogous to the phonemes in spoken languages [104]. Cheremes, derived from the Greek word for hand, describe features such as handshape, orientation, location, movement, and non-manual expressions. When transcribing sign language, linguists commonly employ sub-unit and gloss-level representations [68, 175], which have been explored in this thesis. Unfortunately, curating transcriptions is time-consuming and costly, as a result, linguistic resources are often limited or non-existent [16].

Sign Language Production (SLP) is the task of translating from a spoken language sentence to a continuous sign language sequence. To facilitate natural communication, SLP must include both manual and non-manual components. Previous works have attempted to learn a direct mapping from T2P. However, as discussed, they suffer from regression to the mean [216, 156, 153, 155]. Alternative two-step methods (T2G2P), such as the method presented in Chapter 5, rely on expensive linguistic annotation [65, 74, 153].

In this chapter, we propose to create a data-driven representation for sign language that can be used as a replacement for expensive linguistic annotation and as a lower dimensional alternative to video and skeleton pose. Our approach learns a codebook of motions from continuous 3D

pose data using a NSVQ model [187]. The codebook can be considered the lexicon of our new representation and used to tokenise a continuous skeleton pose sequence into a sequence of discrete codes. Following Figure 6.1 from top to bottom, we tackle the problem as a traditional sequence-to-sequence task, translating from a spoken language sentence to a sequence of codebook tokens (Figure 6.1 row one to two). Unlike the previous two-step approaches, our intermediary representation can be directly mapped to a sequence of skeleton poses that include non-manual key points (Figure 6.1 row two to three). Therefore, the translation is performed by a single model, which is trained end-to-end.

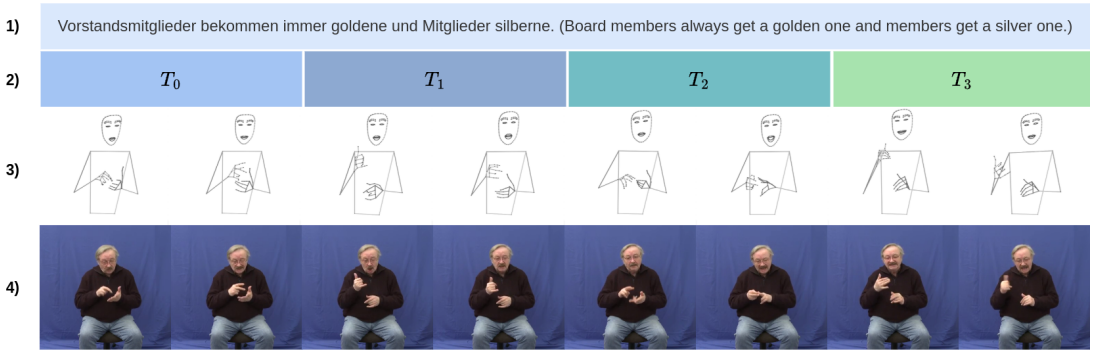


Figure 6.1: A overview of our approach to Sign Language Production (SLP). Showing from top to bottom 1) the source spoken language sentence, 2) the data-driven token representation of sign, 3) the synthesised skeleton pose sequence, and, 4) the original video.

The method maps similar motions to the same codebook token, meaning the approach is able to collapse the natural variation seen in the data. This is a key advantage over previous works, as it allows the model to learn a more compact representation of sign language in comparison to skeleton pose data. This is similar to how gloss provides a person independent representation. However, unlike gloss, our representation is data-driven and can be learned from any dataset without the need for expert knowledge. The model relies on similar spatial and temporal features to group similar motions together, meaning that sign like SISTER and CHOCOLATE could be mapped to the same token, given the hand-shape and motion are the same and the only difference being the starting location of the sign (SISTER starts at the nose, whereas, CHOCOLATE starts at the chin). To avoid this, we introduce a contrastive learning approach that encourages the model to learn a more discriminative representation based on the meaning of the motions. This enhances the codebook and improves the back-translation performance.

When decoding the predicted codebook tokens, there is no guarantee that the sequence will be continuous. Discontinuities can occur between the start of one token and the end of the previous. Inspired by the impressive result from the previous chapter, we add a stitching module that creates a natural transition between tokens. Improving the realism and the comprehension of the final sequence. Finally, the signGAN model is applied to generate a photo-realistic signer conditioned on the pose sequence. In comparison to the previous chapter and other works, we show improved performance on PHOENIX14**T** and the more challenging MeineDGS dataset. An extensive ablation study reveals the effectiveness of our approach. Increasing the back-translation scores on PHOENIX14**T** by up to 20% and 60% in terms of BLEU-1 and BLEU-4, respectively. Finally, we share quantitative results.

6.1 Methodology

The aim of SLP is to enable seamless translation from spoken to signed languages. To accomplish this, we convert a source spoken language sequence, denoted as $X = (x_1, x_2, \ldots, x_W)$ with W words, into a continuous sequence of skeleton poses, denoted as $P = (p_1, p_2, ..., p_U)$ with U frames. Where each pose consists of J joints in $\mathcal{D}$ dimensional space e.g. $p_u \in \mathbb{R}^{J \times \mathcal{D}}$. SLP is a significant challenge, given the target length is substantially greater than the source ($U >> W$) and the dimensionality of the skeleton poses. This inherent difficulty persists even when employing state-of-the-art sequence-to-sequence models for the translation task [189]. To overcome this, we first learn a codebook of tokens (Section 6.1.1), each represents a short sequence of signing and can be decoded to a sequence of skeleton poses. The translation can then be performed by a single model, translating from spoken language text to a sequence of these tokens (Section 6.1.5). The final pose sequence is used to condition the SignGAN module (Section 6.1.7) to generate a photo-realistic signer. We elaborate on each stage of the approach in the subsequent sections.

6.1.1 Codebook

The objective of the codebook is to learn a set of motions from a dataset of continuous signing. Our approach employs a transformer encoder-decoder architecture with a NSVQ module. The NSVQ architecture was chosen over previous techniques for two primary reasons: it produces more stable gradients, leading to more effective optimisation, and it simplifies the training process by requiring only a single loss function. This contrasts with previous techniques that rely on balancing two or three loss functions and risk failing to converge if they are incorrectly weighted during training.

Next, we explain each model in turn, following Figure 6.2 from left to right;

Encoder

Given a short sequence of skeleton poses, $P = (p_1, p_2, ..., p_{U_{\text{cb}}})$ with U_{cb} frames, we add positional encoding to each pose. We then embed the sequence using a linear layer, which acts only in the spatial dimension. Then the sequence is passed to the spatial-temporal transformer

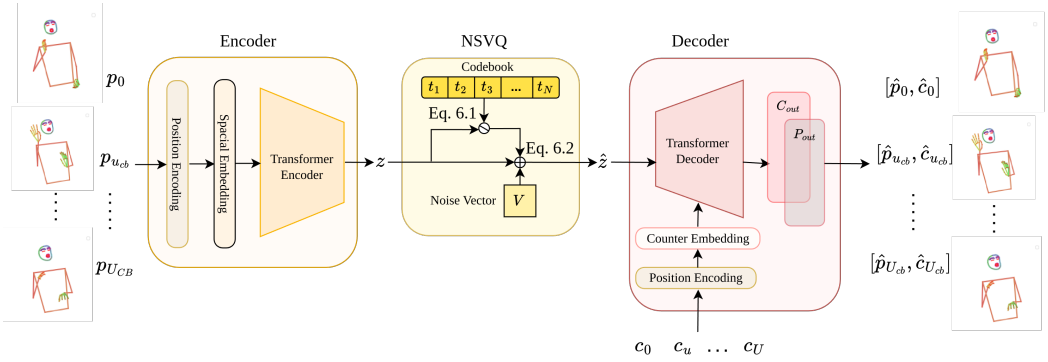


Figure 6.2: An overview of the Codebook training architecture.

encoder, allowing the network to learn dependencies within the sequence. The embedded features can be defined as $\mathbf{z} \in \mathbb{R}^{U_{\text{cb}} \times E}$, where E is the embedding size. Note we train each codebook entry to represent a sub-unit of a full continuous sequence, hence $U_{\text{cb}} << U$.

NSVQ

The NSVQ codebook learns a set of embeddings from the encoder, we denote the codebook as $CB = [\mathbf{t_1^z}, \mathbf{t_2^z}, ..., \mathbf{t_N^z}]$, where N is the number of embeddings in the codebook and each $\mathbf{t_i^z} \in \mathbb{R}^{U_{\text{cb}} \times E}$. Therefore, the length of each sequence, U_{cb}, determines how many frames each codebook entry represents. To train this module, each output from the encoder, $\mathbf{z}$, is mapped to a single codebook token $\mathbf{t_i^z}$. This is called VQ and is defined as;

$$\mathbf{t_i^z} \; ; \; i = \arg\min_{\mathbf{t_i^z}} ||\mathbf{z} - \mathbf{t_i^z}||^2 \tag{6.1}$$

Equation (6.1) is non-differentiable. To overcome this, the NSVQ simulates the quantisation error by adding noise to the input vector, such that the simulated noise forms the same distribution as the original VQ error. The NSVQ is trained end-to-end, and the output to the decoder can be defined as;

$$\hat{\mathbf{z}} = \mathbf{z} + ||\mathbf{z} - \mathbf{t_i^z}|| * \frac{\mathbf{V}}{||\mathbf{V}||} \tag{6.2}$$

Where $\mathbf{V}$ is a normally distributed noise source. Figure 6.2 depicts how Equation (6.1) and (6.2) are used during training.

Decoder

The decoder learns to reconstruct the original skeleton pose sequence from the quantised embedding. Here we use counter decoding from Saunders et al. [155] to drive the decoder, in addition to non-autoregressive decoding. Meaning a sequence is processed in a single step, for reduced computational cost and faster inference speeds. We find that using this approach outperforms a simple multilayer perceptron on reconstruction error. The value of the counter is defined as;

$$c_u = \frac{u+1}{U_{\text{cb}}} \tag{6.3}$$

Where u is the current position in the sequence and U_{cb} is the total sequence length. As shown in Figure 6.2 (Decoder), we use a linear layer to embed the counter values and add positional encoding. We then apply a spatial-temporal transformer decoder, conditioned on the quantised embedding, to produce the output skeleton pose sequence. From this, we project the embedding back to the pose and counter values using two linear layers. Our architecture is trained end-to-end using the following loss function;

$$L_{\text{Code}} = \frac{1}{U_{\text{cb}}} \sum_{u=1}^{U_{\text{cb}}} (p_u - \hat{p_u})^2 + \alpha (c_u - \hat{c}_u)^2 \tag{6.4}$$

where α is a scaling factor that we determine empirically and $\hat{p}$, $\hat{c}$, are the predicted skeleton pose and counter values. While p and c are the ground truth skeleton pose and counter values, respectively.

6.1.2 Codebook Replacement

Codebook collapse is a common challenge when training a codebook with VQ [45]. This occurs when several tokens within the codebook are no longer selected during quantisation, resulting in dead codebook tokens. This can occur when the codebook embeddings no longer accurately represent the data distribution. Strategies exist to detect and replace these dead tokens [60, 187, 215].

We propose two replacement strategies to reduce dead codebook entries and encourage a more uniform distribution of active entries. Entries with usage below a threshold percentage are replaced with either (1) a randomly selected active entry, plus a small magnitude of Gaussian

noise, or (2) a randomly selected embedding from the encoder, denoted as ($\mathbf{z}$). By tracking the usage of each token over a given number of batches we can determine active tokens when the percentage used is greater than β and dead tokens when the percentage used is less than γ. We set a schedule for training, initially using replacement more often and slowly decreasing the frequency throughout training. Once the learning rate decreases past a given factor we stop using replacement, allowing the network to fine tune its parameters.

6.1.3 Contrastive Learning

When linguistic annotation is available, we apply an additional contrastive loss. Specifically, we add a supervised contrastive loss [94] to the encoder of the codebook transformer, as shown in Figure 6.2. This makes use of gloss labels and time stamps to tag each input sequence with its corresponding gloss ID. For long input sequences that contain frames from multiple glosses, we select the most common ID. The contrastive loss pulls sequences belonging to the same gloss together, while simultaneously pushing apart sequences belonging to different glosses. We hypothesise that the additional loss allows the encoder to overcome the natural variation between signers, helping the model become person-invariant. While simultaneously training the model to discriminate between signs of a similar form but with distinct meaning, e.g. SISTER and CHOCOLATE.

We define the loss as,

$$L_{\text{Con}} = \sum_{i=0}^{I} \frac{-1}{|\mathcal{A}(i)|} \sum_{a=0}^{\mathcal{A}(i)} \left(\frac{exp(z_i \cdot z_a/\tau)}{\sum_{b=1}^{\mathcal{B}(i)} exp(z_i \cdot z_b/\tau)} \right) \tag{6.5}$$

Here i is the index of the anchor. $\mathcal{A}(i)$ is the set of indices that correspond to positive samples in the batch and $|\mathcal{A}(i)|$ is the number of samples in a batch. While, $\mathcal{B}(i)$, is the set of negative samples. τ is a scalar temperature parameter. We define a sequence to be positive if it shares the same gloss ID with the anchor, while we define a negative sample if it has a different ID. The contrastive loss is scaled by λ_{Con} before being added to the codebook loss. Therefore, the total loss is defined as;

$$L_{\text{Total}} = L_{\text{Code}} + \lambda_{\text{Con}} L_{\text{Con}} \tag{6.6}$$

6.1.4 Pose Sequence Tokenization

To perform a translation from text to tokens, we first tokenise the continuous skeleton pose sequence. We build the codebook to be a sub-unit representation. Thus, given a continuous sequence of poses, P we create a sequence of tokens, $T = [t_1, t_2, ..., t_{\mathcal{M}}]$ where $\mathcal{M}$ is the number of tokens, which corresponds to the length of the original sequence, such that $\mathcal{M} = \lfloor U/U_{\text{cb}} \rfloor$. With the encoder and codebook frozen, each segment is processed by the encoder, and the corresponding token is found using Equation (6.1). The tokenised skeleton pose sequence is then used as the translation target.

6.1.5 Text-to-Codebook Translation

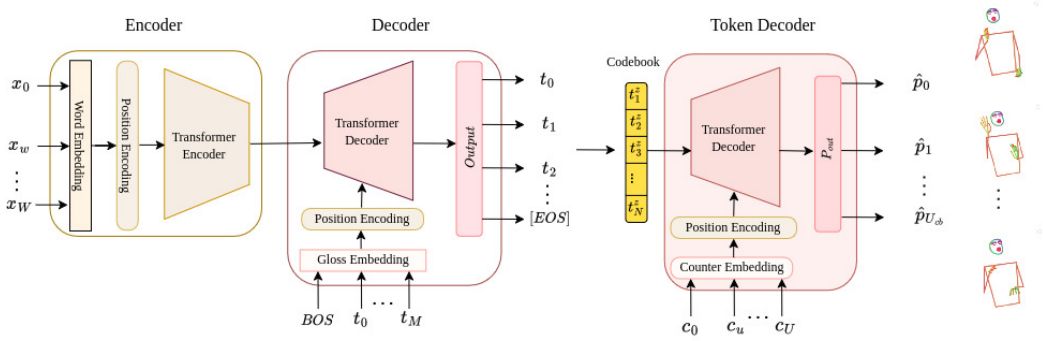


Figure 6.3: An overview of the Text-to-Codebook Tokens Translation architecture.

Given a spoken language sequence, $X = (x_1, x_2, \ldots, x_W)$ we aim to produce the corresponding sequence of codebook tokens, $T = [t_1, t_2, ..., t_{\mathcal{M}}]$, therefore the translation model learns the conditional probability $p(T|X)$.

First, the spoken language sequence, X, is embedded using a linear layer and positional encoding is added. The embedding is then passed through the encoder, giving the context embedding that is utilised by the cross-attention mechanism in the decoder. Autoregressive decoding is employed, starting with the first token and continuing until the end-of-sentence token is predicted, as illustrated in Figure 6.3. Each token is embedded using a linear layer, with positional encoding, similar to the encoder.

The predicted tokens are then decoded back to the skeleton pose space, as shown in Figure 6.3 (Token Decoder). Each segment is joined together to create the continuous sequence.

6.1.6 Codebook Stitching

As discussed, the predicted sequence of tokens, T, can be mapped to a sequence of skeleton poses, P. However, discrepancies may arise between the final pose of one token and the initial pose of the next, resulting in discontinuities. To address this, we employ the smart stitching and sequence resampling modules from Chapter 5. This effectively joins codebook entries together, as a result, we generate more natural continuous sequences.

6.1.7 SignGAN

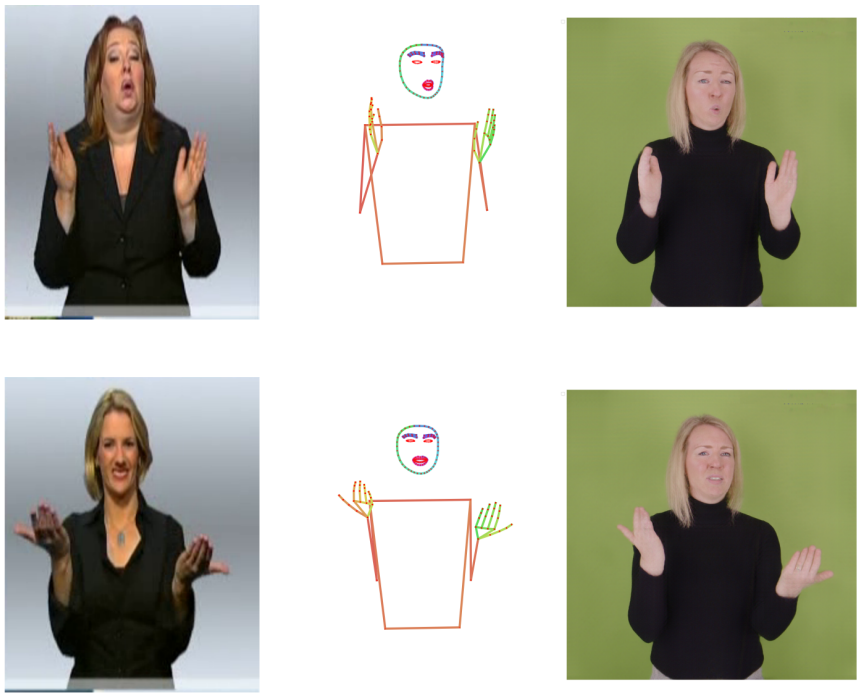


Figure 6.4: A figure showing the original signer (left), skeleton pose (middle) and GAN output (right).

Once again, we apply the SignGAN model to generate a photo-realistic signer conditioned on the skeleton pose sequence. We use the same architecture as described in Section 5.2.3. But this

time, we retrain the model on a curated dataset. For this, we employ a camera rig which contains a 6K front camera in addition to 6 other 1080p cameras, all of which are synchronised. We then record a native deaf signer performing translations. This allows us to train the SignGAN model with higher quality data, increasing the GAN output from 512 (seen in the previous chapter) to 1080p. We find that this approach produces more realistic signers, with fewer artefacts.

Figure 6.4 shows that the approach is able to capture non-manual features and faithfully reproduce them on the photo-realistic signer. However, the model is not perfect and can sometimes produce artefacts. For example, in the figure above, a slight blur can be seen on the signer's hands.

6.2 Experiments

6.2.1 Implementation Details

In our experiments, we search for the best hyperparameters and found the following settings to be the most effective. We build our encoder-decoder translation model using a single layer with four heads, opting for an embedding size of 512 and a feed-forward size of 1024. The resultant architecture contains 7.8 million parameters. When decoding, we employ a beam search algorithm with a size of 5 and a length penalty of 2.0.

Our codebook model consists of a smaller encoder-decoder that has 1.2 million parameters. The model has 2 layers with 4 heads and is built with an embedding and feed-forward size of 128. We set a codebook replacement threshold of 0.1%. Initially, we conduct replacement once per epoch and gradually reduce the frequency by 10 every 50 epochs. We set the initial learning rate to 10^{-4} and stop the codebook replacement once the learning rate reaches 10^{-6}.

Both models employ dropout with a probability of 0.1 [171]. We use ReLU activation between the layers and apply pre-layer normalisation for regularisation and training stability. We train with a reduce on plateau scheduler with a patience of 5 and a decrease factor of 0.9. To initialise the transformer encoder and decoder layers we employ a Xavier initialiser [62] with zero bias and Adam optimisation [95]. The learning rate is initially set to 10^{-4} and we train the model till convergence. We down-sample all skeleton pose sequences to 12 fps, to reduce the computational cost.

For comparison on the MeineDGS dataset, we train two variants of the progressive transformer until convergence with the settings presented in [155].

6.2.2 Quantitative Evaluation

In this section, we provide a quantitative evaluation of our SLP approach. Initially, we search for the optimal vocabulary size and window size for our codebooks, showing that the result is dataset-dependent. Following this, we conduct an ablation study on the MeineDGS dataset, demonstrating the advantages of both the contrastive loss and the stitching techniques. We then apply an enhanced codebook trained on MeineDGS to the PHOENIX14**T** dataset. Finally, to facilitate a meaningful comparison on PHOENIX14**T**, we train a G2P model and assess its performance against prior works.

Codebook Vocabulary Size

Our first experiment searches for the best vocabulary size for each dataset. We fixed the window size to 4 frames and trained each codebook till convergence. After we freeze the codebook and use it to tokenise each dataset for translation.

PHOENIX14**T**	TEST SET						DEV SET					
Vocabulary Size:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
1000	0.1014	25.57	14.64	10.28	7.93	26.88	**0.1019**	26.88	15.16	10.64	6.77	22.51
2000	0.1029	28.00	16.57	11.85	9.23	28.55	0.1027	27.23	15.90	11.41	8.90	27.92
3000	**0.1012**	28.21	16.87	12.12	9.45	28.68	0.1026	27.78	16.30	11.50	8.84	28.08
4000	0.1028	**30.57**	**18.94**	**13.65**	**10.76**	**31.14**	0.1024	28.14	**17.03**	**12.29**	**9.58**	29.59
5000	0.1037	28.90	17.50	12.62	9.83	29.65	0.1032	**28.54**	17.00	12.15	9.52	**29.62**
6000	0.1043	27.76	16.68	12.02	9.37	28.89	0.1037	27.44	16.14	11.49	8.95	28,65
7000	0.1053	28.26	16.81	12.08	9.50	29.44	0.1066	25.53	14.93	10.82	8.60	27.14
8000	0.1055	26.23	14.97	10.61	8.35	26.98	0.1056	25.59	14.73	10.54	8.31	26.70
9000	0.1232	22.00	11.29	7.76	5.98	23.23	0.1203	22.34	11.21	7.62	5.82	23.06
10000	0.1137	21.48	10.31	7.02	5.42	21.20	0.1133	20.79	9.89	6.52	4.93	20.76
25000	0.1129	17.46	8.39	5.89	4.71	19.20	0.1104	17.45	8.49	5.777	4.51	19.85

Table 6.1: The results of translating from spoken language Text-to-Pose with different codebook vocabulary sizes on the RWTH-PHOENIX-Weather-2014**T** dataset.

As shown in Table 6.1, the best vocabulary size is found to be 4,000 on PHOENIX14**T**, achieving an impressive 30.57 BLEU-1 and 31.14 ROUGE score. The optimum is roughly 4 times larger than the original gloss vocabulary, suggesting the model is learning a set of detailed motions and the different co-articulations between signs. On the larger dataset, MeineDGS, a larger vocabulary is found to be optimal at 6,000 as shown in Table 6.2. This is Understandable given

MeineDGS	TEST SET						DEV SET					
Vocabulary Size:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
1000	0.1459	18.71	4.23	1.16	0.38	20.24	0.1451	18.72	4.07	1.02	0.26	20.11
2000	0.1386	18.96	4.52	**1.34**	0.49	20.63	0.1394	18.97	4.38	1.23	0.36	20.36
3000	0.1447	17.09	4.07	1.18	0.43	21.03	0.1454	16.92	3.82	0.98	0.30	20.62
4000	0.1512	19.21	4.22	1.29	**0.50**	20.35	0.1510	18.95	3.89	0.98	0.26	20.19
5000	0.1666	18.54	4.21	1.27	0.48	21.94	0.1684	18.54	4.09	1.18	0.38	21.61
6000	0.1454	**19.99**	**4.68**	1.28	0.47	22.20	0.1468	**19.84**	**4.69**	**1.40**	**0.50**	21.72
7000	0.1528	18.75	4.47	1.28	0.42	22.32	0.1540	18.56	4.08	1.07	0.23	21.96
8000	0.1651	17.91	4.09	1.09	0.23	20.75	0.1659	17.58	3.86	1.01	0.28	20.26
9000	**0.1354**	19.20	4.27	1.11	0.40	**23.05**	**0.1357**	19.07	3.98	1.00	0.29	**22.63**
10000	0.1539	18.02	4.19	1.13	0.42	20.01	0.1543	17.50	3.78	1.03	0.37	19.59
25000	0.1496	17.58	3.91	1.09	0.42	19.49	0.1496	17.28	3.60	0.96	0.32	19.20

Table 6.2: The results of translating from spoken language Text-to-Pose with different codebook vocabulary sizes on the MeineDGS Annotated (MeineDGS) dataset.

the larger domain of discourse, hence the larger gloss vocabulary of approximately 4500. At the optimal vocabulary, a reasonable score of 19.99 BlEU-1 and 22.20 ROUGE were achieved on the test set. However, the model showed limited performance on BLEU-2 to 4 metrics. This is due to the limits of the back-translation model. On the ground truth data, the model achieved only 0.8 BLEU-4 (shown in Table 6.6 row 1, GT). Relative to this theoretical target performance, the approach performs well.

The smaller the codebook size, the more data points map to a single token. As a result, tokens can suffer from regression to the mean, resulting in under-articulated signing. Hence, on both datasets, a lower DTW-MJE is observed at small codebook sizes as each token is more likely to contain a mean skeleton pose. As this metric does not assess the meaning of the transition, we do not use it to determine the best vocabulary size. Therefore, we choose to follow BLEU and ROUGE scores. On MeineDGS we also see a decrease in the DTW-MJE metric at a larger vocabulary size of 9,000, we propose this is due to the codebook containing more detailed motions.

On PHOENIX14**T** we find our best codebook used 3985 tokens to tokenise the training data, a 99.4% vocabulary usage. MeineDGS has a similar result, with 99.6% of tokens being used. We attribute this to the aggressive codebook replacement strategy, which effectively removes dead tokens.

Codebook Window Size

Next, we investigate the best window size for each codebook entry. The vocabulary is fixed to the optimum found in the previous experiment, 4,000 on PHOENIX14**T** and 6,000 on MeineDGS. On both datasets, we find the better window size to be 4 frames. As both datasets are sub-sampled to 12 fps, the optimal size corresponds to $1/3$ of a second of signing. From the gloss timing annotation in the MeineDGS dataset we find the average sign length to be approximately 0.26 seconds, suggesting the model is learning a sign level representation.

PHOENIX14**T**	TEST SET						DEV SET					
Window Size:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
32	**0.1000**	19.24	10.20	7.37	5.89	21.23	**0.0986**	18.68	9.79	6.89	5.41	21.16
24	0.1002	20.10	10.08	7.04	5.54	21.39	0.1005	19.57	9.85	6.79	5.28	21.33
16	0.1047	22.06	11.45	7.97	6.18	24.36	0.1031	21.95	11.19	7.52	5.71	24.17
12	0.1045	23.71	13.12	9.29	7.33	24.51	0.1044	23.91	12.64	8.72	6.76	23.75
8	0.1039	25.44	13.86	9.27	7.04	26.21	0.1040	24.67	13.25	8.88	6.70	25.79
4	0.1040	**30.57**	**18.94**	**13.65**	**10.76**	**31.14**	0.1032	**28.14**	**17.03**	12.29	9.58	29.59
2	0.1048	28.61	17.84	13.14	10.46	30.75	0.1047	27.62	16.88	**12.51**	**10.07**	**30.57**

Table 6.3: The results of translating from spoken language Text-to-Pose with different codebook window sizes on the RWTH-PHOENIX-Weather-2014**T** dataset.

MeineDGS	TEST SET						DEV SET					
Window Size:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
32	**0.0958**	15.19	3.03	0.76	0.00	15.55	**0.0952**	15.26	2.84	0.75	0.23	15.38
16	0.0996	18.05	4.16	1.28	0.45	19.06	0.0988	17.75	3.79	1.01	0.29	18.64
24	0.1113	17.72	3.62	0.98	0.33	17.82	0.1110	17.52	3.27	0.78	0.18	17.57
12	0.1258	18.03	4.04	1.11	0.42	20.33	0.1249	17.99	3.93	1.16	0.40	20.36
8	0.1291	19.56	4.52	1.28	0.41	21.36	0.1291	19.62	4.44	1.24	0.36	21.76
4	0.1528	**19.99**	**4.68**	1.28	0.47	22.20	0.1540	**19.84**	**4.69**	**1.40**	**0.50**	21.72
2	0.1628	19.24	4.65	**1.45**	**0.51**	**22.56**	0.1640	18.98	4.30	1.29	0.37	**22.18**

Table 6.4: The results of translating from spoken language Text-to-Pose with different codebook window sizes on the MeineDGS Annotated (MeineDGS) dataset.

Examination of both Table 6.4 and Table 6.3 reveals that increasing the window size beyond 4, decreased the BLEU and ROUGE scores while improving the DTW-MJE. The minimum DTW-MJE was observed at a window size of 32. Possibly due to the reduced number of tokens in the target sequence, which led to fewer discontinuities in the skeleton pose sequence.

Supervised Contrastive Loss

When training the codebook we apply an additional loss to the encoder of the model, which encourages sequences from the same gloss to have a similar embedding, while simultaneously pushing them away from sequences with a different ID. This helps the model to ignore the natural variation between signers, allowing it to focus on the core similarities and simultaneously, encouraging the model to learn a more discriminative representation based on the meaning of the motions. We evaluate the effectiveness of this approach on the MeineDGS dataset.

MeineDGS	TEST SET						DEV SET					
Vocabulary Size:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
1000	0.1528	20.35	4.91	1.39	0.47	22.19	0.1539	20.14	4.68	1.32	0.41	21.08
2000	**0.1298**	20.43	4.79	1.29	0.41	23.25	**0.1295**	20.32	4.66	1.25	0.36	22.97
3000	0.1498	21.11	5.18	1.45	0.48	23.18	0.1515	20.85	4.84	1.29	0.38	22.79
4000	0.1439	19.83	4.37	1.25	0.38	22.15	0.1456	19.60	4.22	1.15	0.36	21.98
5000	0.1371	**21.62**	**5.35**	1.42	0.45	**23.49**	0.1368	**21.31**	4.91	1.37	0.40	**23.11**
6000	0.1537	20.86	4.83	1.40	**0.53**	22.35	0.1530	20.74	4.66	1.31	0.40	22.27
7000	0.1751	20.77	5.16	1.50	0.51	22.79	0.1770	20.75	**5.06**	**1.46**	**0.47**	22.51
8000	0.1400	21.40	5.20	**1.51**	0.51	22.88	0.1414	21.27	4.80	1.31	0.42	22.67
9000	0.1463	19.49	4.36	1.17	0.32	22.58	0.1474	19.50	4.28	1.23	0.41	22.36
10000	0.1316	18.67	4.31	1.22	0.42	21.72	0.1326	19.08	4.23	1.14	0.36	20.34
25000	0.1768	17.94	4.07	1.21	0.38	20.39	0.1782	17.60	3.77	0.90	0.24	19.97

Table 6.5: The results of translating from spoken language Text-to-Pose with different codebook vocabulary sizes with a supervised contrastive loss on the MeineDGS Annotated (MeineDGS) dataset.

Comparing the Table 6.2 and 6.5 the incorporation of the loss improves the performance on all metrics. But now we find the optimum vocabulary size to be 5,000, a 1,000 token decrease. This is due to the model learning a better representation, allowing it to better distinguish between similar motions that have distinct meanings, while also reducing the number of meaningless tokens. The BLEU-1 and ROUGE scores also increased by 1.59 and 1.81, respectively. This is a significant improvement in performance, showing the effectiveness of the contrastive loss.

Ablation Study

We start by sharing the results of training the back-translation model (GT Table 6.6). The model achieves good BLEU-1 and ROUGE scores. However, the performance was limited on BLEU-2

to 4. The MeineDGS dataset is challenging with a spoken language lexicon of 29,275 (10 times that of PHOENIX14**T**), which limits the back translation results. As such, row 1, GT, should be considered the upper bound for the experiment on MeineDGS.

MeineDGS	TEST SET						DEV SET					
Approach:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
GT	0.0000	20.87	5.60	1.89	0.80	23.78	0.0000	20.75	5.43	1.81	0.76	23.41
Quantisation	0.0273	20.47	4.74	1.30	0.48	22.11	0.0252	19.93	4.54	1.20	0.39	21.73
PT	0.2291	6.11	0.94	0.21	0.05	8.36	0.2284	6.22	0.98	0.17	0.00	8.44
PT + GN	0.2245	7.18	1.48	0.40	0.01	8.38	0.2241	9.22	1.63	0.38	0.01	8.57
Codebook	0.1454	19.99	4.68	1.28	0.47	22.20	0.1468	19.84	4.69	1.40	0.50	21.72
Codebook + Stitching	0.1405	20.67	4.86	1.32	**0.58**	22.16	0.1413	20.43	4.96	**1.48**	**0.70**	21.82
Codebook + Contrastive	0.1371	21.62	5.35	1.42	0.45	**23.49**	0.1368	21.31	4.91	1.37	0.40	**23.11**
Codebook + Contrastive + Stitching	**0.1353**	**22.12**	**5.48**	**1.45**	0.26	23.23	**0.1348**	**21.72**	**5.13**	1.44	0.59	23.01
Chapter 5 Stitcher (ISO)	0.581	16.63	3.75	0.94	0.22	21.69	0.592	16.9	3.67	0.95	0.32	21.34
Chapter 5 Stitcher (CON)	0.637	18.64	4.17	1.07	0.39	21.80	0.637	18.27	4.07	1.19	0.43	21.25

Table 6.6: The results of translating from Text-to-Pose with different approaches on the MeineDGS Annotated (MeineDGS) dataset.

Tokenising a skeleton pose sequence with a codebook causes two quantisation errors. Firstly, the last frames are lost if the sequence is not a multiple of the window size, and secondly, an error in the pose is caused by selecting the closest codebook token. As shown in Table 6.6, the accumulation of these errors reduced the performance from the GT (row 1) to Quantised (row 2), a decrease in BLEU-1 to 4 of 0.40, 0.86, 0.59 and 0.32 on the test set, a relatively small decrease in performance.

For comparison, the following two rows (PT and PT + GN) of Table 6.6 show the results of training a Progressive Transformer on the same data with the same normalisation. In line with the original paper, adding Gaussian Noise (GN) with a standard deviation of 5 increased the BLEU-1 and ROUGE scores by 3.00 and 0.13, respectively on the Dev set (shown in row 4, PT + GN). Despite this augmentation, our baseline approach was shown to outperform both versions of the progressive transformer. Showing an impressive BLEU-1 increase of 12.81 and 10.62 on the Test and Dev sets.

The stitching module, described in 5.2.2, is applied to the predicted sequence to create smoother transitions between codebook tokens. Table 6.6 ”Codebook + Stitching” shows that the stitching module increased the back-translation BLEU-1 score by 0.68, while also improving the mean joint error. This also had qualitative improvements, reducing the number of discontinuities

in the predicted sequences, and as a result, the sequence was more realistic. To evaluate this experimentally, we average the velocity of the signer's skeleton and find that the original data has a standard deviation of 0.047. In contrast, the quantised sequence has a deviation of 0.059. By applying the stitching module to the output, the standard deviation moves closer to the original of 0.041.

We find the best results when combining both stitching and the contrastive loss. On the test set, we observe a 2.13 increase in BLEU-1 and a similar 1.88 increase on the development set. The improvement in the higher n-gram scores is also reflected in the DTW-MJE metric, showing that the proposed techniques help the model recreate the original data more accurately. In comparison to the previous chapter's work, we observe improvements on all metrics, with a significant improvement in the DTW-MJE score. This is understandable, given that the current approach is trained on the original data and therefore is more likely to produce sequences in the style and form of the original data. Overall, we find the best results when combining both techniques, demonstrating the effectiveness of the proposed methods.

Cross-Corpus Codebook Study

Here, we investigate whether a codebook trained on a high-resource dataset can be applied to another. We take a codebook trained on MeineDGS and use it to perform translation on the PHOENIX14**T** dataset. As discussed previously, the MeineDGS dataset is captured from two native deaf signers having a conversation, while the Phoenix dataset comes from TV weather broadcasts. The result is that the camera setups used are distinctly different. Signers from the MeineDGS dataset are looking at each other while the camera is off-centre. The result is a slight morphing in the face mesh shape. Given the 128 key points in the face (a large portion of the skeleton representation), the minor differences between the two face meshes have a distinct impact when selecting the closest codebook entry. Therefore, to mitigate these issues for this experiment, we reduce the face mesh to a subset of key points and retrain the codebooks and translation models.

For this experiment, we fix the hyperparameters to the best found in the previous experiments on the MeineDGS dataset, a vocabulary size of 6,000 and a window size of 4. We present a baseline model in row 1, followed by two codebooks trained on MeineDGS, a normal model

(MeineDGS Codebook) and an enhanced codebook with stitching plus contrastive learning (MeineDGS Codebook+).

PHOENIX14**T**	TEST SET			DEV SET		
Approach:	DTW-MJE	BLEU-1	ROUGE	DTW-MJE	BLEU-1	ROUGE
PHIX Codebook	**0.1014**	**24.68**	23.89	**0.0997**	**24.40**	**24.21**
MeineDGS Codebook	0.1108	22.51	23.38	0.1106	22.71	23.92
MeineDGS Codebook+	0.1064	24.32	**24.03**	0.1046	24.03	23.31

Table 6.7: The results of translating from spoken language Text-to-Pose with codebooks train on different datasets on the RWTH-PHOENIX-Weather-2014**T** dataset.

The results show codebooks can be shared across datasets, although with some reduction in performance, compared to training on the original data. Between Table 6.7 rows 1 and 2, we see a small decrease in BLEU-1 and ROUGE of 1.69 and 0.29, on the Dev set, respectively. However, applying the enhanced codebook to the PHOENIX14**T** dataset recovered the majority of the lost performance.

State-of-the-art Comparison

Finally, to compare against previous works we take our best-performing parameters found in Section 6.2.2 and apply them to the G2P task. Results for comparison are provided by [212]. We find our approach outperforms all previous methods on all metrics, including the progressive transformer [155], a non-autoregressive transformer [74] and a diffusion-based approach [65]. Compared to the next best model we achieve improvements of 11% and 89% in DTW-MJE and BLEU-1.

In line with the findings in Chapter 5, we find the best performance is when translating from text. Once again, we assume this is due to the additional context within the spoken language, that assists the model in producing an accurate translation. Overall, we find the BLEU-1 and 4 scores increase by 4.42 and 3.19 when translating from text.

PHOENIX14**T** Approach:	DTW-MJE	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE
GT	0.000	32.41	20.19	14.41	11.32	32.96
PT + GN[155]	0.191	11.45	7.08	5.08	4.04	-
NAT-AT [74]	0.177	14.26	9.93	7.11	5.53	-
NAT-EA [74]	0.146	15.12	10.45	7.99	6.66	-
PoseVQ-MP [212]	0.146	15.43	10.69	8.26	6.98	-
PoseVQ_Diffusion [212]	0.116	16.11	11.37	9.22	7.50	-
G2P Codebook	0.104	26.15	14.32	9.85	7.57	29.09
T2P Codebook	**0.102**	**30.57**	**18.94**	**13.65**	**10.76**	**31.14**

Table 6.8: The results of translating from Text-to-Pose and Gloss-to-Pose on the RWTH-PHOENIX-Weather-2014**T** dataset.

Publically Available Codebook

The corresponding code for this chapter can be found here[1] and is publicly available under the Apache 2.0 licence.

[1] `https://github.com/walsharry/Sign-VQ-Transformer`

6.2.3 Qualitative Evaluation

PCA Visualisation

To help evaluate the distribution of codebook tokens, we project the codebook's embedding and the original data to 2D using PCA. As shown in Figure 6.5.a the codebook token (shown in orange) are spread across the space, with no entries lying outside the original data distribution (shown in blue). Figure 6.5.b shows the effect of not applying the replacement strategy during training. As can be seen, the codebook tokens are not well distributed, and they have collapsed to a single point. This indicates that our aggressive codebook replacement strategies help reduce the number of dead tokens, resulting in a more representative codebook. Finally, Figure 6.5.c shows the effect of the contrastive loss. With the addition of the loss, the codebook finds a new arrangement in the embedding space.

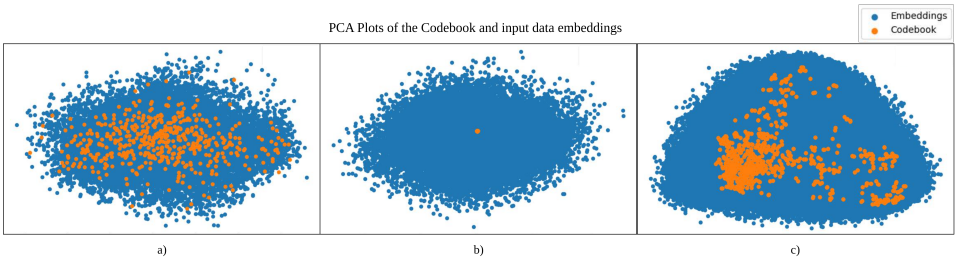


Figure 6.5: A visualisation of the codebook embedding space using PCA. Showing the effect; a) with the replacement strategy, b) without the replacement strategy, c) with the contrastive loss.

Codebook Size Effect

As observed in the experiments, an optimum vocabulary size exists. For the PHOENIX14**T** and MeineDGS datasets this was found to be 4,000 and 6,000, respectively. Here we visualise the effect of changing the size of the codebook. For each sequence, we find the closest matching codebook entry and plot it alongside the ground truth (gt). As shown in Figure 6.6, 6.7, and 6.8, the larger the codebook, the more detailed the skeleton pose. At larger codebook sizes, we suggest that the model fails to collapse the natural variation between signers effectively. Hence, this factor reduces performance, as multiple tokens are used to represent the same sequence

with the same meaning. As reflected in the results, we suggest that too small a codebook size can lead to under-articulated signing.

Figure 6.6: A visualisation of the effect of changing the codebook size on signed sequences.

Figure 6.7: A visualisation of the effect of changing the codebook size on signed sequences

Figure 6.8: A visualisation of the effect of changing the codebook size on signed sequences

Translation Examples

Figure 6.9, 6.10, and, 6.11 show translation examples from the PHOENIX14**T** dataset. The figure shows that the model is able to faithfully translate a given sentence. However, note that some details are missing in the hands caused by the quantisation error from the codebook. In addition, we share video outputs from our best models [2]. To ensure a realistic evaluation, we also share failure cases.

[2] https://github.com/walsharry/Representations-for-Sign-Language-Production/blob/master/ch6/README.md

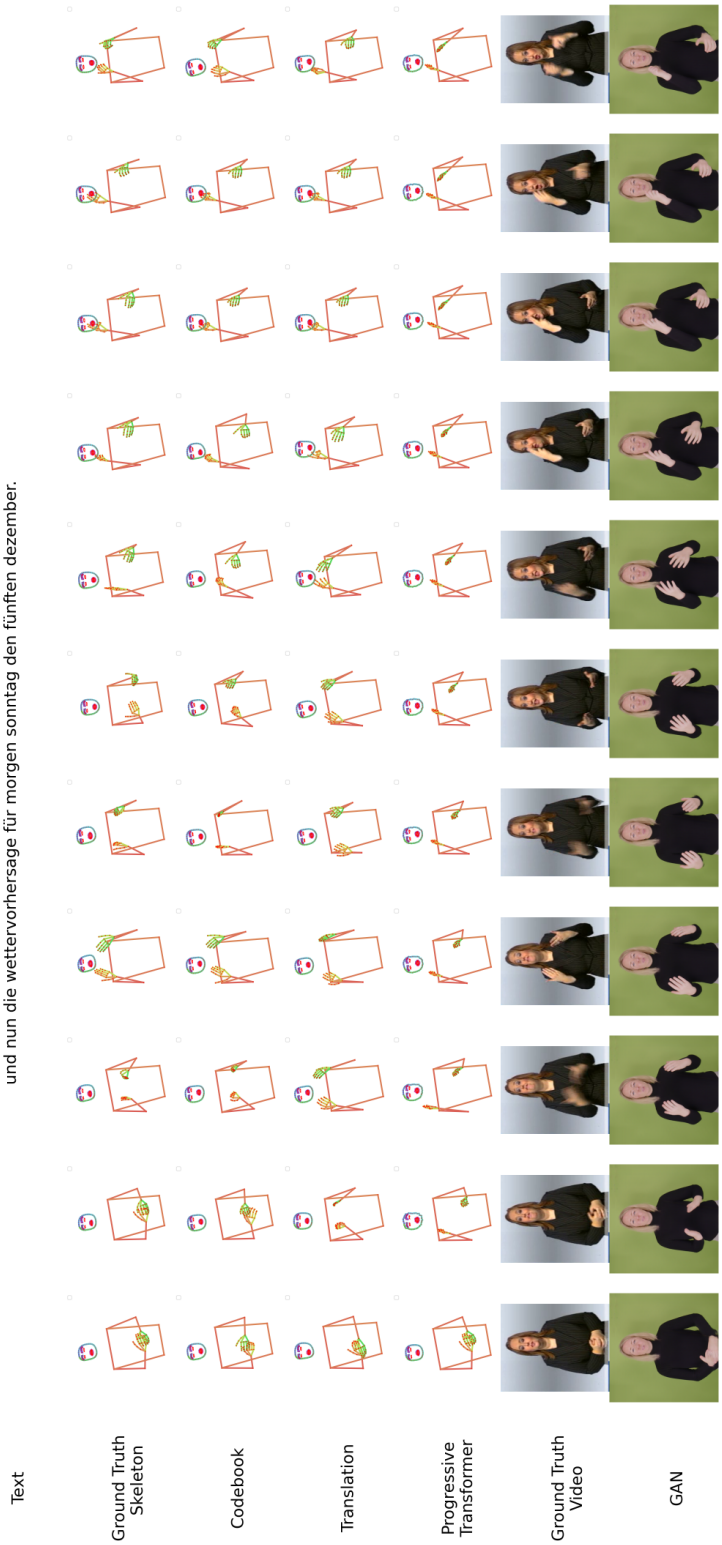


Figure 6.9: A Translation example produced by our best model, that uses a 4000 token codebook.

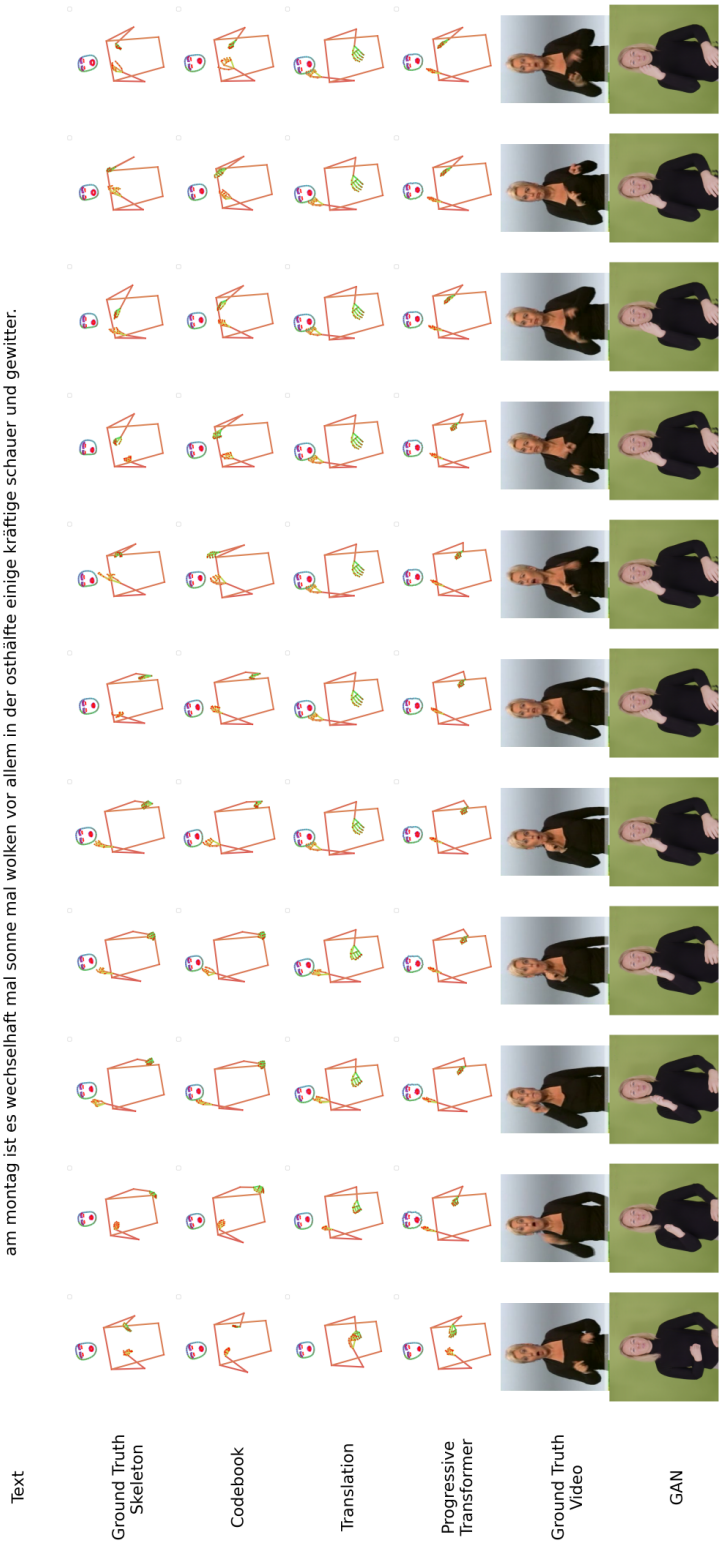


Figure 6.10: A Translation example produced by our best model, that uses a 4000 token codebook.

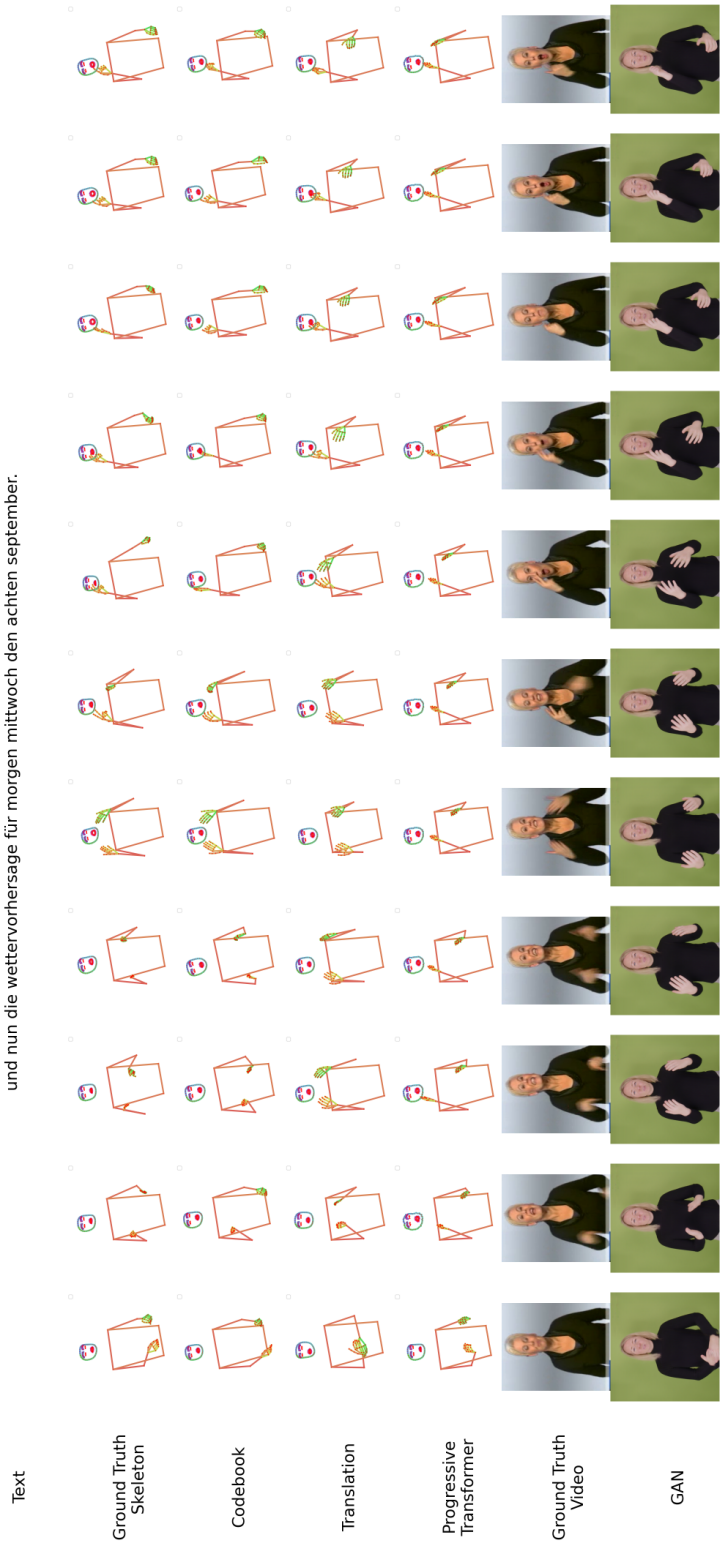


Figure 6.11: A Translation example produced by our best model, that uses a 4000 token codebook.

6.3 Conclusion

Previously, T2P translation was treated as a skeleton pose regression problem, where the goal was to synthesise a pose sequence directly from text or from a gloss intermediary. As a result, productions often suffered from regression to the mean [159]. The previous chapter's work made use of pre-recorded signs to overcome this limitation, but therefore, required a dictionary and linguistic annotation. A limiting factor when attempting to scale to large domains of discourse.

To address these limitations, we proposed a novel approach to T2P translation. Using a transformer NSVQ model, we learnt a new representation of sign language from the data. The model acts on both the spatial and temporal domains, allowing it to learn an expressive vocabulary of motions. This new representation is used as a substitute to gloss in the translation pipeline, helping to overcome the need for linguistic annotation. Each token can be mapped to a continuous motion, meaning the full translation can be achieved by a single model. Making it computationally efficient in comparison to the previous chapter's two-step approach. Using this method, we showed state-of-the-art back-translation scores on both the MeineDGS and PHOENIX14**T**.

When joining tokens together, we noted discontinuity in the motion. To address this, we leverage the stitching approach from the previous chapter. This allowed the approach to generate seamless transitions, and, therefore, more natural sequences. This improvement was reflected in the quantitative results. In addition, we showed how linguistic annotation can be leveraged to further improve the approach and, in cases where linguistic annotation is absent, we demonstrated the feasibility of sharing codebooks across datasets.

This is a promising direction for future SLP research. One main limitation of the approach is the fixed temporal window size. We observe in the data a large temporal variation with most signs, often changing greatly with context and individual person. Given that the transformer architecture is capable of working with variable length inputs, it would be interesting to see if a model could be trained to learn a temporal invariant representation. This would bring the codebook close to a gloss unit level. We hypothesise that this would further improve the quality of the generated sequences.

Chapter 7

Conclusions and Future Work

This thesis aimed to advance the field of Sign Language Production (SLP), by translating from spoken language sentences to a video of a photo-realistic signer. To achieve this, we focused on the representations that can be used in the translation pipeline. The work has the potential to reduce the communication barrier between the deaf and hearing communities, with applications in a variety of areas such as education, travel, news, social media, etc. In Chapter 2, we reviewed the existing literature and found many shortcomings. Mainly, regression to the mean, over reliance on gloss annotation and a lack of alternative representations. Now we summarise the contributions of each chapter and detail how they addressed the shortcomings of the existing literature.

While spoken languages have symbolic written representation, e.g. words, Sign languages have no clear substitute. This motivated the thesis to find the best representation of sign language for automatic machine translation. At the start of this thesis, few alternatives to gloss have been explored in the literature. In Chapter 3, the first contribution, we explored an alternative to gloss named the Hamburg Notation System (HamNoSys). This representation provides a more semantically rich description of sign. By introducing the task of T2H, we were able to compare against a traditional T2G approach, and by leveraging powerful techniques from the NLP domain such as work tokenisation, large language models and additional supervision, we showed improved performance. This helped to set a baseline for the following chapters to build on.

From initial findings, we were motivated to explore T2G translation. Analysis of the two data sets, PHOENIX14**T** and MeineDGS, revealed the large lexical overlap between the gloss notation systems and the corresponding spoken language. This observation allowed us to decompose the translation task into two distinct sub-problems. GS, the task of selecting the correct vocabulary, followed by GR, changing from spoken language order to sign order. We explored two methods for performing GR, a deep learning and a statistical approach. The statistical approach leveraged more linguistic features such as POS and word classes, which proved to be effective tools for reordering, despite the domain shift from spoken to signed languages. While the deep learning approach used the state-of-the-art transformer model to learn the reordering directly. Using the alignment, the S&R approach showed significant improvements in lower n-gram BLEU scores on all datasets, in comparison to a traditional T2G model.

The first two contributions focused on the translation to text-based intermediaries, showing improved performance in a variety of settings. The following contribution, Chapter 5, completed the translation pipeline. An enhanced gloss representation, termed Gloss++, was stitched together to create a skeleton pose sequence, followed by a production of a photo-realistic signer. Previous approaches have suffered from regression to the mean [216, 156, 153, 155], leading to under-articulated and incomprehensible signing. The proposed approach utilised dictionary examples, which are guaranteed to be expressive. The strategy focused on recreating sign prosody. In addition to including non-manual features (facial keypoints), which enhances the naturalness of the production, quantitative evaluation demonstrated the effectiveness of the approach in comparison to previous state-of-the-art works. Given the quality of the production, we were able to use the approach to synthesise synthetic skeleton data, which was then used to enhance recognition and translation performance in a CSLR model. Finally, the user evaluation agreed with the quantitative results, indicating a strong preference for the stitched sequences.

Creating intermediary representations that require specialist linguistic knowledge have become a limiting factor in scaling approaches to larger domains of discourse including this thesis. This motivated the final contribution of the thesis, Chapter 6, which overcomes the need for linguistic annotation, allowing us to learn a representation from any quantity of data in a self-supervised manner. Although when available, the linguistic annotation can still be used to enhance the learnt representation. Using a transformer encoder-decoder, the model learns a spatial-temporal representation of skeleton pose. Then, using vector quantisation, the space can be divided up

creating a codebook which represents the input data. By using real sign data, the codebook was guaranteed to be expressive, once again overcoming issues related to regression to the mean. The codebook representation acts as an alternative to gloss in the translation pipeline. However, unlike previous representations, the codebook can be directly mapped to the pose space, reducing the computational complexity during inference. Finally, the skeletons are mapped to a photo-realistic signer using the SignGAN architecture. The scalability of the approach is a significant milestone for SLP.

7.1 Future Work

This thesis has made significant progress in exploring representations for sign language machine translation. However, further work is needed to create productions which fully capture the complexity of the language. This includes the need for more data, as well as more sophisticated models. In Chapter 5 and 6 we proposed two distinct approaches to tackle the SLP problem. We see this as two promising directions for future work to build on.

Building on the work in Chapter 5, the approach could be improved in two ways:

1) Improving the quantity of lexical variants in the dictionary. Using a dictionary of isolated signs has the advantage of guaranteeing the expressiveness of the translation, but this also limits productions to specific versions of a sign. By collecting a diverse dataset with a large domain of discourse, more isolated signs can be added to the dictionary. This includes a number of signs that depend on the context of the sequence. This would help a SLP system capture features like sign placement, where a signer will place an object or person in the signing space so that they can be referred back to later or to aid in a description. Since LLMs [21, 180, 181] have been shown to be effective in understanding large context windows, these models could be leveraged to decide which lexical variant is the most appropriate given the context. This could improve the fluency and naturalness of the translation.

2) Simplifying the architecture and improving efficiency. The three-step approach allows each intermediary to be finely tuned to produce the best output while also allowing for more points of failure. In addition, each step requires more compute. Given the recent success with video generation models [72]. It would be interesting to see if these techniques could be used to

simplify the pipeline to two steps. Keeping the initial T2G translation. While joining steps 2 and 3 into a single video stitching model that would operate in the latent space to predict transitioning frames between two gloss videos. Such a model could be trained in a self-supervised manner. Plus, open source models, like Open SORA [221], could be employed to reduce the training cost of such an approach.

Given the success of using skeleton stitching to generate synthetic data, it would be interesting to see if a video stitching approach could be used to generate pseudo-sign language data. That could then be used to pre-train both SLP and SLT models. As sign will always struggle to collect a comparable quantity of data in comparison to spoken languages. In part, due to the number of users, and the increased storage size of video compared to text. This could be a promising direction for future work.

Building on the work in Chapter 6, we make the following suggestions:

Gloss notation is effective as it helps to collapse down spatial and temporal variations, which can vary drastically due to the natural variation within the data and the lexical variant seen between regions. Chapter 6 presented a method for automatically creating a representation. However, there is a disparity between the underlying signal and the temporal window the codebook uses. The method is good at collapsing spatial variants which are similar. However, it may struggle with large temporal variations. As a result, temporal variants can be encoded using a different set of tokens, possibly negatively affecting downstream tasks.

Given that the approach uses a transformer encoded-decoder it is capable of encoding variable length inputs. Therefore, future work could look at creating a temporal invariant codebook. This could be achieved by segmenting the input sequences based on some semantic information rather than a fixed frame length. Given the existing work in sign segmentation, we suggest a similar model could be incorporated [81, 123, 149].

Both chapters could benefit from recent developments in video generation and rendering. Both approaches utilise a GAN-based approach to generate a photo-realistic signer. However, two techniques, namely diffusion and Gaussian splatting, have shown promising results in this area. Diffusion-based models have shown impressive results and have already been leveraged for sign language tasks [139, 209]. However, they can suffer from hallucinations and require a large amount of data to train, so future work could explore minimising these issues. While

Gaussian splatting has been combined with human parametric models to render human avatars and has shown impressive results [192, 78]. Given that Gaussian splatting is built on top of a human parametric model, the rendering is guaranteed to produce a realistic human form, as it is constrained by the underlying mesh. However, the appearance is trained on a single individual, which makes customisation more difficult compared to a diffusion-based model. The latter has been shown to be controllable and therefore could be customised to the user's needs. Both techniques could be used to improve the quality of the photo-realistic signer.

Sign language research is where computer vision and natural language processing meet. Given the developments in both fields, it will be interesting to see how the wider knowledge can be applied to this specific domain. Future research should always focus on the end users of this technology and endeavour to incorporate them in its future development. Such research will excitingly lead to a more inclusive and accessible society for all.

Bibliography

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[2] Samuel Albanie, Gül Varol, Liliane Momeni, Hannah Bull, Triantafyllos Afouras, Himel Chowdhury, Neil Fox, Bencie Woll, Rob Cooper, Andrew McParland, et al. Bbc-oxford british sign language dataset. *arXiv preprint arXiv:2111.03635*, 2021.

[3] The Atlantic. Why Sign-Language Gloves Don't Help Deaf People. *https://www.theatlantic.com/technology/archive/2017/11/why-sign-language-gloves-dont-help-deaf-people/545441/*, 2017.

[4] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[6] Alberto Baldrati, Davide Morelli, Giuseppe Cartella, Marcella Cornia, Marco Bertini, and Rita Cucchiara. Multimodal garment designer: Human-centric latent diffusion models for fashion image editing. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 23393–23402, 2023.

[7] Srinivas Bangalore, Giuseppe Riccardi, et al. Finite-state models for lexical reordering in spoken language translation. In *INTERSPEECH*, pages 422–425, 2000.

[8] Andrew Bangham, SJ Cox, Ralph Elliott, John RW Glauert, Ian Marshall, Sanja Rankov, and Mark Wells. Virtual signing: Capture, animation, storage and transmission-an

overview of the visicast project. In *IEE Seminar on speech and language processing for disabled and elderly people (Ref. No. 2000/025)*, pages 6–1. IET, 2000.

[9] L Barnes. Basic BSL Signs. *https://www.scribd.com/document/168881865/Basic-BSL-Signs*, 2024.

[10] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[11] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3(null):1137–1155, mar 2003.

[12] Ankan Kumar Bhunia, Salman Khan, Hisham Cholakkal, Rao Muhammad Anwer, Jorma Laaksonen, Mubarak Shah, and Fahad Shahbaz Khan. Person image synthesis via denoising diffusion model. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5968–5976, 2023.

[13] Arianna Bisazza and Marcello Federico. A survey of word reordering in statistical machine translation: Computational models and language phenomena. *Computational linguistics*, 42(2):163–205, 2016.

[14] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[15] Yosra Bouzid and Mohamed Jemni. An avatar based approach for automatically interpreting a sign language notation. In *2013 IEEE 13th International Conference on Advanced Learning Technologies*, pages 92–94. IEEE, 2013.

[16] Danielle Bragg, Oscar Koller, Mary Bellard, Larwan Berke, Patrick Boudreault, Annelies Braffort, Naomi Caselli, Matt Huenerfauth, Hernisa Kacorri, Tessa Verhoef, et al. Sign language recognition, generation, and translation: An interdisciplinary perspective. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility*, pages 16–31, 2019.

[17] Diane Brentari. *A prosodic model of sign language phonology*. Mit Press, 1998.

[18] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024.

[19] Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Frederick Jelinek, John Lafferty, Robert L Mercer, and Paul S Roossin. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85, 1990.

[20] Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480, 1992.

[21] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[22] Access BSL. BSL Signs for Colours across the UK. *https://accessbsl.com/bsl-signs-for-colours/*, 2023.

[23] Jan Bungeroth and Hermann Ney. Statistical sign language translation. In *sign-lang@ LREC 2004*, pages 105–108. Citeseer, 2004.

[24] Stephen Butterworth et al. On the theory of filter amplifiers. *Wireless Engineer*, 7(6):536–541, 1930.

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

[26] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10023–10033, 2020.

[27] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields, 2019.

[28] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[29] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[30] Francisco Casacuberta and Enrique Vidal. Machine translation with inferred stochastic finite-state transducers. *Computational Linguistics*, 30(2):205–225, 2004.

[31] Branden Chan, Stefan Schweter, and Timo Möller. German's next language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6788–6796, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[32] Sheng Chen, Qingshan Wang, and Qi Wang. Semantic-driven diffusion for sign language production with gloss-pose latent spaces alignment. *Computer Vision and Image Understanding*, 246:104050, 2024.

[33] Yutong Chen, Ronglai Zuo, Fangyun Wei, Yu Wu, Shujie Liu, and Brian Mak. Two-stream network for sign language recognition and translation. *Advances in Neural Information Processing Systems*, 35:17043–17056, 2022.

[34] David Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (acl'05)*, pages 263–270, 2005.

[35] Michael Collins, Philipp Koehn, and Ivona Kučerová. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 531–540, 2005.

[36] Kearsy Cormier, Jordan Fenlon, Trevor Johnston, Ramas Rentelis, Adam Schembri, Katherine Rowley, Robert Adam, and Bencie Woll. From corpus to lexical database to online dictionary: Issues in annotation of the bsl corpus and the development of bsl

signbank. In *5th Workshop on the Representation of Sign Languages: Interactions between Corpus and Lexicon [workshop part of 8th International Conference on Language Resources and Evaluation, Turkey, Istanbul LREC 2012. Paris: ELRA. pp. 7–12*, 2012.

[37] Kearsy Cormier, Adam Schembri, David Vinson, and Eleni Orfanidou. First language acquisition differs from second language acquisition in prelingually deaf signers: Evidence from sensitivity to grammaticality judgement in british sign language. *Cognition*, 124(1):50–65, 2012.

[38] Stephen Cox, Michael Lincoln, Judy Tryggvason, Melanie Nakisa, Mark Wells, Marcus Tutt, and Sanja Abbott. Tessa, a system to aid communication with deaf people. In *Proceedings of the fifth international ACM conference on Assistive technologies*, pages 205–212, 2002.

[39] Carl Croneberg, W Stokoe, and D Casterline. A dictionary of american sign language. 1965.

[40] Vincent J Della Pietra. The mathematics of statistical machine translation: Parameter estimation. *Using Large Corpora*, page 223, 1994.

[41] Aashaka Desai, Maartje De Meulder, Julie A Hochgesang, Annemarie Kocab, and Alex X Lu. Systemic biases in sign language ai research: A deaf-led call to reevaluate research agendas. *arXiv preprint arXiv:2403.02563*, 2024.

[42] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[43] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North*, 2019.

[44] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

[45] Sander Dieleman, Aaron van den Oord, and Karen Simonyan. The challenge of realistic music generation: modelling raw audio at scale. *Advances in neural information processing systems*, 31, 2018.

[46] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[47] S Ebling and J Glauert. Exploiting the full potential of jasigning to build an avatar signing train announcements. In *Third International Symposium on Sign Language Translation and Avatar Technology*, October 2013.

[48] Sarah Ebling and Matt Huenerfauth. Bridging the gap between sign language machine translation and sign language animation using sequence classification. In Jan Alexandersson, Ercan Altinsoy, Heidi Christensen, Peter Ljunglöf, François Portet, and Frank Rudzicz, editors, *Proceedings of SLPAT 2015: 6th Workshop on Speech and Language Processing for Assistive Technologies*, pages 2–9, Dresden, Germany, September 2015. Association for Computational Linguistics.

[49] Eleni Efthimiou, Stavroula-Evita Fotinea, Thomas Hanke, John Glauert, Richard Bowden, Annelies Braffort, Christophe Collet, Petros Maragos, and François Lefebvre-Albaret. The dicta-sign wiki: Enabling web communication for the deaf. In *International Conference on Computers for Handicapped Persons*, pages 205–212. Springer, 2012.

[50] Hosni Mostafa El-dali. Towards an understanding of the distinctive nature of translation studies. *Journal of King Saud University-Languages and Translation*, 23(1):29–45, 2011.

[51] Oussama ElGhoul and Mohamed Jemni. Websign: A system to make and interpret signs using 3d avatars. In *Proceedings of the Second International Workshop on Sign Language Translation and Avatar Technology (SLTAT), Dundee, UK*, volume 23, 2011.

[52] Sen Fang, Chunyu Sui, Xuedong Zhang, and Yapeng Tian. Signdiff: Learning diffusion models for american sign language production, 2023.

[53] Sen Fang, Lei Wang, Ce Zheng, Yapeng Tian, and Chen Chen. Signllm: Sign languages production large language models. *arXiv preprint arXiv:2405.10718*, 2024.

[54] Sidney S Fels and Geoffrey E Hinton. Glove-talk: A neural network interface between a data-glove and a speech synthesizer. *IEEE transactions on Neural Networks*, 4(1):2–8, 1993.

[55] Jordan Fenlon, Kearsy Cormier, Ramas Rentelis, Adam Schembri, Katherine Rowley, Robert Adam, and Bencie Woll. Bsl signbank: A lexical database of british sign language (first edition), 2014.

[56] Department for Work and Pensions. The British Sign Language (BSL) report 2022. *https://www.gov.uk/government/publications/the-british-sign-language-bsl-report-2022/the-british-sign-language-bsl-report-2022*, 2022.

[57] Heidi Fox. Phrasal cohesion and statistical machine translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 304–3111, 2002.

[58] Dmitriy Genzel. Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 376–384, 2010.

[59] National Geographic. Sign Language. *https://education.nationalgeographic.org/resource/sign-language/*, 2024.

[60] Allen Gersho and Robert M Gray. *Vector quantization and signal compression*, volume 159. Springer Science & Business Media, 2012.

[61] Sylvie Gibet, François Lefebvre-Albaret, Ludovic Hamon, Rémi Brun, and Ahmed Turki. Interactive editing in french sign language dedicated to virtual signers: Requirements and challenges. *Universal Access in the Information Society*, 15:525–539, 2016.

[62] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010.

[63] Santiago Egea Gómez, Euan McGill, and Horacio Saggion. Syntax-aware transformers for neural machine translation: The case of text to sign gloss translation. In *Proceedings*

of the 14th workshop on building and using comparable corpora (BUCC 2021), pages 18–27, 2021.

[64] Gary J Grimes. Digital data entry glove interface device, November 8 1983. US Patent 4,414,537.

[65] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022.

[66] Zhengsheng Guo, Zhiwei He, Wenxiang Jiao, Xing Wang, Rui Wang, Kehai Chen, Zhaopeng Tu, Yong Xu, and Min Zhang. Unsupervised sign language translation and generation, 2024.

[67] Handspeak. Coarticulation in sign language. *https://www.handspeak.com/learn/101/*, 2024.

[68] Thomas Hanke. Hamnosys-representing sign language data in language resources and language processing contexts. In *LREC*, 2004.

[69] Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing, 2021.

[70] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*, 2021.

[71] Alexis Heloir and Sylvie Gibet. A qualitative and quantitative characterisation of style in sign language gestures. In *International Gesture Workshop*, pages 122–133. Springer, 2007.

[72] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

[73] S Hochreiter. Long short-term memory. *Neural Computation MIT-Press*, 1997.

[74] Wencan Huang, Wenwen Pan, Zhou Zhao, and Qi Tian. Towards fast and high-quality sign language production. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3172–3181, 2021.

[75] Matt Huenerfauth. A multi-path architecture for machine translation of english text into american sign language animation. In *Proceedings of the Student Research Workshop at HLT-NAACL 2004*, HLT-SRWS '04, page 25–30, USA, 2004. Association for Computational Linguistics.

[76] Eui Jun Hwang, Huije Lee, and Jong C Park. Autoregressive sign language production: A gloss-free approach with discrete representations. *arXiv preprint arXiv:2309.12179*, 2023.

[77] Maksym Ivashechkin, Oscar Mendez, and Richard Bowden. Improving 3d pose estimation for sign language. In *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, pages 1–5, 2023.

[78] Maksym Ivashechkin, Oscar Mendez, and Richard Bowden. Signsplat: Rendering sign language via gaussian splatting. *arXiv preprint arXiv:2505.02108*, 2025.

[79] Zora Jachova, Olivera Kovacheva, and Aleksandra Karovska. Differences between american sign language (asl) and british sign language (bsl). *Journal of Special Education and Rehabilitation*, 9(1-2):41–54, 2008.

[80] Elena Jahn, Reiner Konrad, Gabriele Langer, Sven Wagner, and Thomas Hanke. Publishing dgs corpus data: Different formats for different needs. In *Proceedings of the Workshop on the Representation and Processing of Sign Languages at LREC*, 2018.

[81] Low Jian He, Harry Walsh, Ozge Mercanoglu Sincan, and Richard Bowden. Hands-on: Segmenting individual signs from continuous sequences. In *Proceedings of the 18th International Conference on Automatic Face and Gesture Recognition (FG 2024)*. Institute of Electrical and Electronics Engineers (IEEE), 2024.

[82] Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Fu. Skeleton aware multi-modal sign language recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3413–3423, 2021.

[83] Zifan Jiang, Amit Moryossef, Mathias Müller, and Sarah Ebling. Machine translation between spoken languages and signed languages represented in signwriting. *arXiv preprint arXiv:2210.05404*, 2022.

[84] T. Johnston and A. Schembri. *Australian Sign Language (Auslan): An introduction to sign language linguistics*. Cambridge University Press, 2007.

[85] Xuan Ju, Ailing Zeng, Chenchen Zhao, Jianan Wang, Lei Zhang, and Qiang Xu. Humansd: A native skeleton-guided diffusion model for human image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15988–15998, 2023.

[86] Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. Fast decoding in sequence models using discrete latent variables. In *International Conference on Machine Learning*, pages 2390–2399. PMLR, 2018.

[87] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013.

[88] Jakub Kanis, Jiří Zahradil, Filip Jurčíček, and Luděk Müller. Czech-sign speech corpus for semantic based machine translation. In Petr Sojka, Ivan Kopeček, and Karel Pala, editors, *Text, Speech and Dialogue*, pages 613–620, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[89] Khushdeep Kaur and Parteek Kumar. Hamnosys to sigml conversion system for sign language automation. *Procedia Computer Science*, 2016.

[90] Rupinder Kaur and Parteek Kumar. Hamnosys generation system for sign language. In *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2014.

[91] Dilek Kayahan and Tunga Güngör. A hybrid translation system from turkish spoken language to turkish sign language. In *2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*, pages 1–6, 2019.

[92] Richard Kennaway. Avatar-independent scripting for real-time gesture animation. *arXiv preprint arXiv:1502.02961*, 2015.

[93] G.D. Kennedy. *A Concise Dictionary of New Zealand Sign Language*. Bridget Williams Books, 2002.

[94] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020.

[95] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[96] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[97] Michael Kipp, Quan Nguyen, Alexis Heloir, and Silke Matthes. Assessing the deaf user perspective on sign language avatars. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*, pages 107–114, 2011.

[98] Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*, 2017.

[99] Philipp Koehn, Franz J Och, and Daniel Marcu. Statistical phrase-based translation. Technical report, University of Southern California Marina Del Rey Information Sciences Inst, 2003.

[100] Oscar Koller, Jens Forster, and Hermann Ney. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *Computer Vision and Image Understanding*, 141:108–125, 2015. Pose & Gesture.

[101] Reiner Konrad, Thomas Hanke, Gabriele Langer, Dolly Blanck, Julian Bleicken, Ilona Hofmann, Olga Jeziorski, Lutz König, Susanne König, Rie Nishio, Anja Regen, Uta

Salden, Sven Wagner, Satu Worseck, Oliver Böse, Elena Jahn, and Marc Schulder. Meine dgs – annotiert. öffentliches korpus der deutschen gebärdensprache, 3. release / my dgs – annotated. public corpus of german sign language, 3rd release, 2020.

[102] Dimitris Kouremenos, Klimis S. Ntalianis, Georgios Siolas, and Andreas Stafylopatis. Statistical machine translation for greek to greek sign language using parallel corpora produced via rule-based machine translation. In *CIMA@ICTAI*, 2018.

[103] Julia Kreutzer, Jasmijn Bastings, and Stefan Riezler. Joey nmt: A minimalist nmt toolkit for novices. *arXiv:1907.12484*, 2019.

[104] John Laver. Linguistic phonetics. *The handbook of linguistics*, pages 150–179, 2001.

[105] M Lewis. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[106] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1459–1469, 2020.

[107] Dongxu Li, Chenchen Xu, Liu Liu, Yiran Zhong, Rong Wang, Lars Petersson, and Hongdong Li. Transcribing natural languages for the deaf via neural editing programs. *arXiv preprint arXiv:2112.09600*, 2021.

[108] Dongxu Li, Chenchen Xu, Liu Liu, Yiran Zhong, Rong Wang, Lars Petersson, and Hongdong Li. Transcribing natural languages for the deaf via neural editing programs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11991–11999, 2022.

[109] Rung-Huei Liang and Ming Ouhyoung. A real-time continuous gesture recognition system for sign language. In *Proceedings third IEEE international conference on automatic face and gesture recognition*, pages 558–567. IEEE, 1998.

[110] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[111] Zehui Lin, Xiao Pan, Mingxuan Wang, Xipeng Qiu, Jiangtao Feng, Hao Zhou, and Lei Li. Pre-training multilingual neural machine translation by leveraging alignment information. *arXiv preprint arXiv:2010.03142*, 2020.

[112] Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[113] Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation, 2020.

[114] Adam Lopez. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):1–49, 2008.

[115] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, 2019.

[116] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In Lluís Màrquez, Chris Callison-Burch, and Jian Su, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[117] Evie Malaia, Joshua D Borneman, and Ronnie B Wilbur. Information transfer capacity of articulators in american sign language. *Language and speech*, 61(1):97–112, 2018.

[118] Chloë R Marshall and Angela Hobsbaum. Sign-supported english: is it effective at teaching vocabulary to young children with english as an additional language? *International journal of language & communication disorders*, 50(5):616–628, 2015.

[119] Tomas Mikolov. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[120] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 2013.

[121] Donna A Morere and Rachel Roberts. Fingerspelling. In *Assessing literacy in deaf individuals: Neurocognitive measurement and predictors*, pages 179–189. Springer, 2012.

[122] Masahiro Mori, Karl F MacDorman, and Norri Kageki. The uncanny valley [from the field]. *IEEE Robotics & automation magazine*, 19(2):98–100, 2012.

[123] Amit Moryossef, Zifan Jiang, Mathias Müller, Sarah Ebling, and Yoav Goldberg. Linguistically motivated sign language segmentation. *arXiv preprint arXiv:2310.13960*, 2023.

[124] Amit Moryossef, Kayo Yin, Graham Neubig, and Yoav Goldberg. Data augmentation for sign language gloss translation. *arXiv:2105.07476*, 2021.

[125] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, and Ying Shan. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 4296–4304, 2024.

[126] Tetsuji Nakagawa. Efficient top-down BTG parsing for machine translation preordering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 208–218, Beijing, China, July 2015. Association for Computational Linguistics.

[127] Donna Jo Napoli and Rachel Sutton-Spence. Order of the major constituents in sign languages: Implications for all language. *Frontiers in psychology*, 5:376, 2014.

[128] NDC National Deaf Center. Attitudes and Biases as Barriers for Deaf People: New Online Course Explores Their Impact and Ways to Take Action. *https://nationaldeafcenter.org/news-items/attitudes-and-biases-barriers-deaf-people-new-online-course-explores-their-impact-and-ways-take/*, 2021.

[129] NHS National Health Service. Severe / profound deafness (audiology). *https://www.cuh.nhs.uk/clinics/severe-profound-deafness/*, 2024.

[130] Graham Neubig, Taro Watanabe, and Shinsuke Mori. Inducing a discriminative parser to optimize machine translation reordering. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 843–853, 2012.

[131] nrcpd. Registration figures. *https://www.nrcpd.org.uk/registration-figures*, 2024.

[132] Franz Josef Och and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual meeting of the Association for Computational Linguistics*, pages 295–302, 2002.

[133] Stein Erik Ohna. Open your eyes: deaf studies talking, 2010.

[134] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2017.

[135] Achraf Othman and Mohamed Jemni. Statistical sign language machine translation: from english written text to american sign language gloss, 2011.

[136] Carol A Padden and Darline Clark Gunsauls. How the alphabet came to be used in a sign language. *Sign Language Studies*, pages 10–33, 2003.

[137] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[138] Ankita Pasad, Felix Wu, Suwon Shon, Karen Livescu, and Kyu Han. On the use of external data for spoken named entity recognition. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 724–737, Seattle, United States, July 2022. Association for Computational Linguistics.

[139] Anton Pelykh, Ozge Mercanoglu Sincan, and Richard Bowden. Giving a hand to diffusion models: a two-stage approach to improving conditional human image generation, 2024.

[140] Xiangyu Peng, Zangwei Zheng, Chenhui Shen, Tom Young, Xinying Guo, Binluo Wang, Hang Xu, Hongxin Liu, Mingyan Jiang, Wenjun Li, Yuhui Wang, Anbang Ye, Gang Ren, Qianran Ma, Wanying Liang, Xiang Lian, Xiwen Wu, Yuting Zhong, Zhuangyan Li, Chaoyu Gong, Guojun Lei, Leijun Cheng, Limin Zhang, Minghao Li, Ruijie Zhang, Silan Hu, Shijie Huang, Xiaokang Wang, Yuanheng Zhao, Yuqi Wang, Ziang Wei, and Yang You. Open-sora 2.0: Training a commercial-level video generation model in 200k. *arXiv preprint arXiv:2503.09642*, 2025.

[141] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[142] Roland Pfau, Josep Quer, et al. *Nonmanuals: their grammatical and prosodic roles*. na, 2010.

[143] Maja Popović. chrF++: words helping character n-grams. In Ondřej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, and Julia Kreutzer, editors, *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[144] Siegmund Prillwitz and Hamburg Zentrum für Deutsche Gebärdensprache und Kommunikation Gehörloser. *HamNoSys: Version 2.0; Hamburg notation system for sign languages; an introductory guide*. Signum-Verlag, 1989.

[145] David Quinto-Pozos and Robert Adam. 379 sign language contact. In *The Oxford Handbook of Sociolinguistics*. Oxford University Press, 02 2013.

[146] A Radford. Improving language understanding by generative pre-training. 2018.

[147] Razieh Rastgoo, Kourosh Kiani, Sergio Escalera, and Mohammad Sabokrou. Sign language production: a review. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3451–3461, 2021.

[148] Judy S Reilly, Marina L McIntire, and Howie Seago. Affective prosody in american sign language. *Sign Language Studies*, pages 113–128, 1992.

[149] Katrin Renz, Nicolaj C Stache, Samuel Albanie, and Gül Varol. Sign language segmentation with temporal convolutional networks. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2135–2139. IEEE, 2021.

[150] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[151] RNID Royal National Institute for Deaf People. Not 12 million, but 18 million: why the number of people classed as having hearing loss in the UK has increased). *https://rnid.org.uk/2024/06/why-the-number-of-people-with-hearing-loss-has-increased/*, 2024.

[152] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, nov 1975.

[153] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. Adversarial training for multi-channel sign language production. *arXiv preprint arXiv:2008.12405*, 2020.

[154] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. Everybody sign now: Translating spoken language to photo realistic sign language video. *arXiv preprint arXiv:2011.09846*, 2020.

[155] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. Progressive transformers for end-to-end sign language production. In *European Conference on Computer Vision*, 2020.

[156] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. Continuous 3d multi-channel sign language production via progressive transformers and mixture density networks. *International journal of computer vision*, 129(7):2113–2135, 2021.

[157] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. Mixed signals: Sign language production via a mixture of motion primitives. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[158] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. Signing at scale: Learning to co-articulate signs for large-scale photo-realistic sign language production. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[159] Benjamin Saunders. *Photo-Realistic Sign Language Production*. PhD thesis, University of Surrey, 2024.

[160] Adam Schembri. British sign language corpus project: Open access archives and the observer's paradox. In *sign-lang@ LREC 2008*, pages 165–169. European Language Resources Association (ELRA), 2008.

[161] Adam Schembri, Jordan Fenlon, Ramas Rentelis, and Kearsy Cormier. British sign language corpus project: A corpus of digital video data and annotations of british sign language 2008-2017 (third edition). `https://www.bslcorpusproject.org`, 2017.

[162] Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal processing (ICASSP)*, 2012.

[163] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909, 2015.

[164] Bowen Shi, Diane Brentari, Greg Shakhnarovich, and Karen Livescu. Open-domain sign language translation learned from online video. *arXiv preprint arXiv:2205.12870*, 2022.

[165] BSL Signbank. Fingerspelling. *https://bslsignbank.ucl.ac.uk/*, 2024.

[166] Ozge Mercanoglu Sincan, Necati Cihan Camgoz, and Richard Bowden. Is context all you need? scaling neural sign language translation to large domains of discourse. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1955–1965, 2023.

[167] Shashi Pal Singh, Ajai Kumar, Hemant Darbari, Lenali Singh, Anshika Rastogi, and Shikha Jain. Machine translation using deep learning: An overview. In *2017 international conference on computer, communications and electronics (comptelix)*, pages 162–167. IEEE, 2017.

[168] Nils Skotara, Uta Salden, Monique Kügow, Barbara Hänel-Faulhaber, and Brigitte Röder. The influence of language deprivation in early childhood on l2 processing: An erp comparison of deaf native signers and deaf signers with a delayed language acquisition. *BMC neuroscience*, 13:1–14, 2012.

[169] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[170] Sign Solutions. How many people use BSL in the UK? *https://www.signsolutions.uk.com/how-many-people-use-bsl-in-the-uk/*, 2023.

[171] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 2014.

[172] William C Stokoe. Sign language structure. *Annual Review of Anthropology*, 1980.

[173] Stephanie Stoll, Necati Cihan Camgöz, Simon Hadfield, and R. Bowden. Sign language production using neural machine translation and generative adversarial networks. In *British Machine Vision Conference*, 2018.

[174] Stephanie Stoll, Necati Cihan Camgoz, Simon Hadfield, and Richard Bowden. Text2sign: Towards sign language production using neural machine translation and generative adversarial networks. *Int. J. Comput. Vision*, 128(4):891–908, apr 2020.

[175] Valerie Sutton. *Lessons in SignWriting*. SignWriting Press, 2022.

[176] Valerie J Sutton. Signwriting basics. 2009.

[177] Rachel Sutton-Spence and Bencie Woll. *The linguistics of British Sign Language: an introduction*. Cambridge University Press, 1999.

[178] Shinichi Tamura and Shingo Kawasaki. Recognition of sign language motion images. *Pattern recognition*, 21(4):343–353, 1988.

[179] Shengeng Tang, Feng Xue, Jingjing Wu, Shuo Wang, and Richang Hong. Gloss-driven conditional diffusion models for sign language production. *ACM Trans. Multimedia Comput. Commun. Appl.*, may 2024. Just Accepted.

[180] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[181] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[182] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018.

[183] Don Tuggener. *Incremental coreference resolution for German*. PhD thesis, University of Zurich, 2016.

[184] UCL University Collage London. Marriage Certificate of Thomas Tillsye. *https://www.ucl.ac.uk/british-sign-language-history/beginnings/marriage-certificate-thomas-tillsye*, 2024.

[185] Dot Sign language Unversity of Surrey. Signing Space. *https://bsl.surrey.ac.uk/principles/g-signing-space*, 2024.

[186] Dave Uthus, Garrett Tanzer, and Manfred Georg. Youtube-asl: A large-scale, open-domain american sign language-english parallel corpus. *Advances in Neural Information Processing Systems*, 36, 2024.

[187] Mohammad Hassan Vali and Tom Bäckström. Nsvq: Noise substitution in vector quantization for machine learning. *IEEE Access*, 10:13598–13610, 2022.

[188] Clayton Valli and Ceil Lucas. *Linguistics of American sign language: An introduction*. Gallaudet University Press, 2000.

[189] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[190] Lucas Ventura, Amanda Duarte, and Xavier Giró-i Nieto. Can everybody sign now? exploring sign language video generation from 2d poses. *arXiv preprint arXiv:2012.10941*, 2020.

[191] Harry Walsh, Ed Fish, Ozge Mercanoglu Sincan, Mohamed Ilyes Lakhal, Richard Bowden, Neil Fox, Kearsy Cormier, Bencie Woll, Kepeng Wu, Zecheng Li, Weichao Zhao, Haodong Wang, Wengang Zhou, Houqiang Li, Shengeng Tang, Jiayi He, Xu Wang, Ruobei Zhang, Yaxiong Wang, Lechao Cheng, Meryem Tasyurek, Tugce Kiziltepe, and Hacer Yalim Keles. Slrtp2025 sign language production challenge: Methodology, results, and future work. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[192] Harry Walsh, Maksym Ivashechkin, and Richard Bowden. Using sign language production as data augmentation to enhance sign language translation. In *Adjunct Proceedings of the 25th ACM International Conference on Intelligent Virtual Agents*, IVA Adjunct '25, New York, NY, USA, 2025. Association for Computing Machinery.

[193] Harry Walsh, Abolfazl Ravanshad, Mariam Rahmani, and Richard Bowden. A data-driven representation for sign language production. In *Proceedings of the 18th International Conference on Automatic Face and Gesture Recognition (FG 2024)*. Institute of Electrical and Electronics Engineers (IEEE), 2024.

[194] Harry Walsh, Ben Saunders, and Richard Bowden. Changing the representation: Examining language representation for neural sign language production. In *LREC 2022 Workshop Language Resources and Evaluation Conference 24 June 2022*, page 117, 2022.

[195] Harry Walsh, Ben Saunders, and Richard Bowden. Select and reorder: A novel approach for neural sign language production. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 14531–14542, 2024.

[196] Harry Walsh, Ben Saunders, and Richard Bowden. Sign stitching: A novel approach to sign language production. In *The 35th British Machine Vision Conference (BMVC)*, 2024.

[197] Harry Walsh, Ozge Mercanoglu Sincan, Ben Saunders, and Richard Bowden. Gloss alignment using word embeddings. In *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, pages 1–5. IEEE, 2023.

[198] WHO. Deafness and hearing loss, 2021.

[199] Ronnie B Wilbur. Stress in a sl: Empirical evidence and linguistic i ssues. *Language and speech*, 42(2-3):229–250, 1999.

[200] Ronnie B Wilbur. Effects of varying rate of signing on asl manual signs and nonmanual markers. *Language and speech*, 52(2-3):245–285, 2009.

[201] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[202] Rosalee Wolfe, John C McDonald, Thomas Hanke, Sarah Ebling, Davy Van Landuyt, Frankie Picron, Verena Krausneker, Eleni Efthimiou, Evita Fotinea, and Annelies Braffort. Sign language avatars: A question of representation. *Information*, 13(4):206, 2022.

[203] Ryan Wong, Necati Cihan Camgoz, and Richard Bowden. Learnt contrastive concept embeddings for sign recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1945–1954, 2023.

[204] Ryan Wong, Necati Cihan Camgoz, and Richard Bowden. Sign2gpt: Leveraging large language models for gloss-free sign language translation. *arXiv preprint arXiv:2405.04164*, 2024.

[205] Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403, 1997.

[206] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural

machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[207] 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, Łukasz 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. Google's neural machine translation system: Bridging the gap between human and machine translation, 2016.

[208] Fei Xia and Michael C McCord. Improving a statistical mt system with automatically learned rewrite patterns. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 508–514, 2004.

[209] Zhaoyang Xia, Carol Neidle, and Dimitris N. Metaxas. Diffslva: Harnessing diffusion models for sign language video anonymization, 2023.

[210] Qinkun Xiao, Minying Qin, and Yuting Yin. Skeleton-based chinese sign language recognition and generation for bidirectional communication between deaf and hearing people. *Neural Networks*, 125:41–55, 2020.

[211] Pan Xie, Taiyi Peng, Yao Du, and Qipeng Zhang. Sign language production with latent motion transformer, 2023.

[212] Pan Xie, Qipeng Zhang, Zexian Li, Hao Tang, Yao Du, and Xiaohui Hu. Vector quantized diffusion model with codeunet for text-to-sign pose sequences generation. *arXiv preprint arXiv:2208.09141*, 2022.

[213] Huijie Yao, Wengang Zhou, Hao Zhou, and Houqiang Li. Semi-supervised spoken language glossification. *arXiv preprint arXiv:2406.08173*, 2024.

[214] Aoxiong Yin, Haoyuan Li, Kai Shen, Siliang Tang, and Yueting Zhuang. T2s-gpt: Dynamic vector quantization for autoregressive sign language production from text. *arXiv preprint arXiv:2406.07119*, 2024.

[215] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

[216] Jan Zelinka and Jakub Kanis. Neural sign language synthesis: Words are our glosses. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 3384–3392, 2020.

[217] Biao Zhang, Mathias Müller, and Rico Sennrich. Sltunet: A simple unified model for sign language translation. *arXiv preprint arXiv:2305.01778*, 2023.

[218] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023.

[219] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[220] Xuan Zhang and Kevin Duh. Approaching sign language gloss translation as a low-resource machine translation task. In *Proceedings of the 1st International Workshop on Automatic Translation for Signed and Spoken Languages (AT4SSL)*, pages 60–70, 2021.

[221] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, March 2024.

[222] Benjia Zhou, Zhigang Chen, Albert Clapés, Jun Wan, Yanyan Liang, Sergio Escalera, Zhen Lei, and Du Zhang. Gloss-free sign language translation: Improving from visual-language pretraining. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20871–20881, 2023.

[223] Ronglai Zuo, Fangyun Wei, and Brian Mak. Natural language-assisted sign language recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14890–14900, 2023.

[224] Inge Zwitserlood, Margriet Verlinden, Johan Ros, Sanny Van Der Schoot, and T Netherlands. Synthetic signing for the deaf: Esign. In *Proceedings of the conference and workshop on assistive technologies for vision and hearing impairment (CVHI)*, 2004.