Multi-channel Transformers for Multi-articulatory Sign Language Translation

Necati Cihan Camgoz[1], Oscar Koller[2], Simon Hadfield[1], and Richard Bowden[1]

[1] CVSSP, University of Surrey, UK, {n.camgoz, s.hadfield, r.bowden}@surrey.ac.uk,
[2] Microsoft, Munich, Germany, oscar.koller@microsoft.com

Abstract. Sign languages use multiple asynchronous information channels (articulators), not just the hands but also the face and body, which computational approaches often ignore. In this paper we tackle the multi-articulatory sign language translation task and propose a novel multi-channel transformer architecture. The proposed architecture allows both the inter and intra contextual relationships between different sign articulators to be modelled within the transformer network itself, while also maintaining channel specific information. We evaluate our approach on the RWTH-PHOENIX-Weather-2014T dataset and report competitive translation performance. Importantly, we overcome the reliance on gloss annotations which underpin other state-of-the-art approaches, thereby removing the need for expensive curated datasets.

Keywords: sign language translation, multi-channel, sequence-to-sequence

1 Introduction

Sign languages are the main medium of communication of the Deaf. Every country typically has its own sign language and although some grammatical structures are shared, as are signs that rely upon heavy iconicity, different sign language have unique vocabularies [57, 59]. Contrary to spoken and written languages, sign languages are visual. This makes automatic sign language understanding a novel research field where computer vision and natural language processing meet with a view to understanding and translating the spatio-temporal linguistic constructs of sign [7].

Signers use multiple channels to convey information [61]. These channels, also known as articulators in linguistics [46], can be grouped under two main categories with respect to their role in conveying information, namely manual and non-manual features [8]. Manual features include the hand shape and its motion. Although manual features can be considered as the dominant part of the sign morphology, they alone do not encapsulate the full context of the conveyed information. To give clarity, emphasis and additional meaning, signers use non-manual features, such as facial expressions, mouth gestures, mouthings[3] and body pose. Furthermore, both manual and non-manual features effect each other's meaning when used together.

[3] Mouthings are lip patterns that accompany a sign.

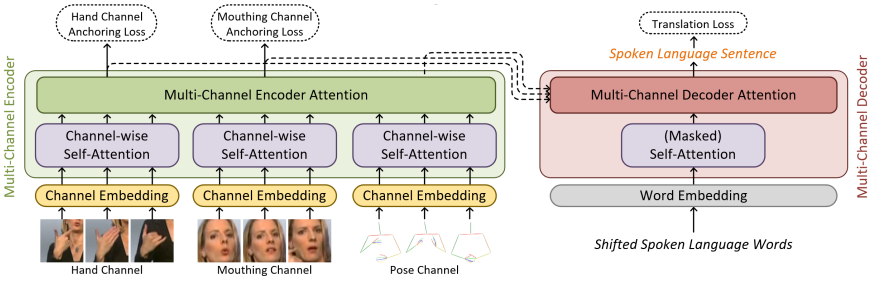

Fig. 1. An overview of the proposed Multi-channel Transformer architecture applied to the multi-articulatory SLT task.

To date, the literature in the field has predominantly focused on using the manual features to realize sign language recognition and translation [50, 39, 13], thus ignoring the rich and essential information contained in the non-manual features. This focus on the manual features is partially responsible for the common misconception that sign language recognition and translation problems are special sub-tasks of the gesture recognition field [54]. Sign language is as rich and complex as any spoken language. However, the multi-channel nature adds additional complexity as channels are not synchronised.

In contrast to much of the existing literature, in this paper we model sign language by incorporating both manual and non-manual features into Sign Language Translation (SLT). To achieve this, we utilize multiple channels which correspond to the articulatory subunits of the sign, namely hand shape, upper body pose and mouthings. We explore several approaches to combine the information present in these channels using both early and late fusion in a transformer architecture. Based on these findings we then introduce a novel deep learning architecture, the Multi-channel Transformer. This approach incorporates both inter and intra channel contextual relationships to learn meaningful spatio-temporal representations of asynchronous sign articulators, while also preserving channel specific information by using anchoring losses. Although this approach was designed specifically for SLT, we believe it can also be used to tackle other multi-channel sequence-to-sequence learning tasks, such as audio-visual speech recognition [1]. An overview of the Multi-channel Transformer in the context of SLT can be seen in Figure 1.

We evaluate our approach on the challenging RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**) dataset which provides both sign gloss[4] annotations and spoken language translations. Previous approaches [13, 12] on PHOENIX14**T** heavily relied upon sign gloss annotations, which are labor intensive to obtain. We aim to remove this dependency on gloss annotation, by utilizing channel specific features obtained from related tasks, such as human pose estimation approaches [14, 27] to represent upper body pose channel or lip reading features [19, 3] to represent mouthings [38, 37]. Removing the dependency on manual annota-

[4] Glosses can be considered as the minimal lexical items of the sign languages.

tion allows our approach to be scaled beyond what is possible with previous techniques, potentially using huge collections of un-annotated data. We empirically show that by integrating multiple articulator channels into our multi-channel transformer, it is possible to achieve competitive SLT performance which is *on par* with models trained using additional gloss annotation.

The contributions of the paper can be summarized as: (1) We overcome the need for expensive gloss-level supervision by combining multiple articulatory channels with anchoring losses to achieve competitive continuous SLT performance on the PHOENIX14**T** dataset. (2) We propose a novel multi-channel transformer architecture that supports multi-channel, asynchronous, sequence-to-sequence learning and (3) We use this to introduce the first successful approach to multi-articulatory SLT, which models the inter and intra relationship of manual and non-manual channels.

2 Literature Review

Computational processing of sign languages is an important field and expected to have tremendous impact on language deprivation of Deaf children, accessibility, sign linguistics and human-computer interaction in general. Its first attempt dates back more than thirty years: A patent describing a hardwired electronic glove that recognized American Sign Language (ASL) finger spelling from hand configurations [24]. In the early days the field moved slowly, focusing first on isolated [60], then continuous sign language recognition [55]. With the rise of deep learning, enthusiasm was revived and accelerated the field [10, 39, 41]. The recognition of limited domain but continuous real-life sign language became feasible [11, 20, 28, 40, 21]. Driven by linguistic evidence [5, 65, 52], the field realized that sign language recognition needs to focus on more than just the hands. Earlier works looked at several modalities separately, such as the face in general [63, 36], head pose [45], the mouth [2, 37, 38], eye-gaze [15], and body pose [53, 17]. More recently, multi-stream architectures showed strong performance [35, 68].

Nevertheless, sign recognition only addresses part of the communication barrier between Deaf and hearing people. Sign languages follow a distinct grammar and are not word by word translations of spoken languages. After successful recognition, reordering and mapping into the target spoken language complete the communication pipeline. In early works, recognition and translation were treated as two independent processing steps. Isolated single signs were recognized and subsequently translated [16]. Often, existing work exclusively considered the problem as a text-to-text translation problem [9, 56], despite the visual nature of sign language and the lack of a written representation.

Generally speaking, much of the available sign translation literature falsely declares sign recognition as sign translation [22, 64, 26, 25]. Camgoz *et al.* [13] were the first to release a joint recognition and translation corpus with videos, glosses and translations to spoken language, covering real-life sign language, recorded from the broadcast news. They proposed to tackle the task based on a Neural Machine Translation (NMT) framework relying on input tokenization of

the videos and subsequent sequence-to-sequence networks with attention. Their best performing tokenization method was based on strong sign recognition models trained using gloss annotations with full video frames and achieved an 18.1 BLEU-4 score, while a simple tokenization scheme (not trained with glosses) only reached 9.6 BLEU-4 on the test set of PHOENIX14**T**. Orbay and Akarun [48] investigated different tokenization methods on the same corpus and showed again that a pretrained hand shape recognizer [39] outperforms simpler approaches and reaches 14.6 BLEU-4. While they also investigated transformer architectures and multiple hands as input, the results underperformed. Ko *et al.* [34] describe a non-public dataset covering sign language videos, gloss annotation and translation. Their method relies on detected body keypoints only. It hence misses the important appearance based characteristics of sign. More recently, Camgoz *et al.* [12] proposed Sign Language Transformers, a joint end-to-end sign language recognition and translation approach. They used pre-trained gloss representations as inputs to their networks and trained transformer encoders using gloss annotations to learn meaningful spatio-temporal representations for SLT. Their approach is the current state-of-the-art on PHOENIX14**T**. They report 20.2 BLEU-4 for pre-trained gloss features to spoken language translation, and 21.3 BLEU-4 with the additional gloss recognition supervision.

Overall, previous work in the space of SLT has two major short-comings, which we intend to address with this paper: (1) The beauty of translations is the abundance of available training data, as they can be created in real-time by interpreters. Glosses are expensive to create and limit data availability. No previous work was able to achieve competitive performance while not relying on glosses. (2) So far SLT has never considered multiple articulators.

3 Background on Neural Machine Translation

The objective of machine translation is to learn the conditional probability $p(\mathcal{Y}|\mathcal{X})$ where $\mathcal{X} = (x_1, ..., x_T)$ is a sentence from the source language with T tokens and $\mathcal{Y} = (y_1, ..., y_U)$ is the desired corresponding translation of said sentence in the target language. To learn this mapping using neural networks, Kalchbrenner *et al.* [32] proposed using an *encoder-decoder* architecture, where the source sentence is encoded into a fixed sized "context" vector which is then used to decode the target sentence. Cho *et al.* [18] and Sutzeker *et al.* [58] further improved this approach by assigning the encoding and decoding stages of translation to individual specialized Recurrent Neural Networks (RNNs).

The main drawback of RNN-based approaches are long term dependency issues. Although there have been practical solutions to this, such as source sentence reversing [58], the context vector is still of fixed size, and thus cannot perfectly encode arbitrarily long input sequences. To overcome the information bottleneck imposed by using the last hidden state of the RNN as the context vector, Bahdanau *et al.* [4] proposed an attention mechanism, which was a breakthrough in the field of NMT. The idea behind the attention mechanism is to use a soft-search over the encoder outputs at each step of target sentence decoding. This

was realized by conditioning target word prediction on a context vector which is a weighted sum of the source sentence representations. The weighting in turn is done by a learnt scoring function which measures the relevance of the decoders current hidden state and the encoder outputs. Luong *et al.* [44] further improved this approach by proposing a dot product attention (scoring) function as:

$$\text{context} = \text{softmax}\left(QK^T\right)V \tag{1}$$

where queries, Q, correspond to the hidden state of the decoder at a given time step, and keys, K, and values, V, represent the encoder outputs.

More recently, Vaswani *et al.* [62] introduced self-attention mechanisms, which refine the source and target token representations by looking at the context they have been used in. Combining encoder and decoder self-attention layers with encoder-decoder attention, Vaswani *et al.* proposed Transformer networks, a fully connected network (as opposed to being RNN-based) which has revolutionized the field of machine translation. In contrast to RNN-based models, transformers obtain Q, K and V values by using individually learnt linear projection matrices at each attention layer. Vaswani *et al.* also introduced "scaled" dot-product attention as:

$$\text{context} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_m}}\right)V \tag{2}$$

where d_m is the number of hidden units of the model. The motivation behind the scaling operation is to counteract the effect of gradients becoming extremely small in cases where the number of hidden units is high and in-turn, the dot products grow large [62].

In this work we extend the transformer network architecture and adapt it to the task of multi-channel sequence-to-sequence learning. We propose a multi-channel attention layer to refine the representations of each source channel in the context of other source channels, while maintaining channel specific information using anchoring losses. We also adapt the encoder-decoder attention layer to be able to use multiple source channel representations.

4 Multi-channel Transformers

In this section we introduce Multi-channel Transformers, a novel architecture for sequence-to-sequence learning problems where the source information is embedded across several asynchronous channels. Given source sequences $\mathcal{X} = (X_1, ..., X_N)$, where X_i is the i^{th} source channel with a cardinality of T_i, our objective is to learn the conditional probability $p(\mathcal{Y}|\mathcal{X})$, where $\mathcal{Y} = (y_1, ..., y_U)$ is the target sequence with U tokens. In the application domain of SLT, these channels correspond to representations of the manual and non-manual features of the sign. An overview of the multi-channel transformer can be seen in Figure 1, while individual attention modules introduced in this paper are visualized in Figure 2. To keep the formulation simple, and to focus the readers attention on the differentiating factors of our architecture, we omit the multi-headed attention, layer normalization and residual connections from our equations, which are the same as the original transformer networks [62].

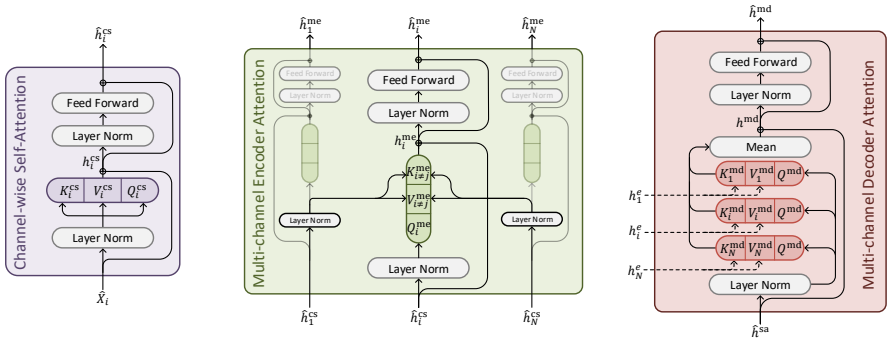

Fig. 2. A detailed overview of the introduced attention modules: (left) Channel-wise Self-Attention, (middle) Multi-channel Encoder Attention, and (right) Mult-channel Decoder Attention

4.1 Channel and Word Embeddings

As with other machine translation tasks, we start by projecting both the source channel features and the one-hot word vectors into a denser embedding where similar inputs lie close to one-another. To achieve this we use linear layers. We employ normalization and activation layers to change the scale of the embedded channel features and give additional non-linear representational capability to the model. The transformer networks do not have an implicit structure to model the position of a token within the sequence. To overcome this, we employ positional encoding [62] to add temporal ordering to the embedded representations. The embedding process for an input feature $x_{i,t}$ coming from the i^{th} channel at time t can be formalized as:

$$\hat{x}_{i,t} = \text{Activ}\left(\text{Norm}\left(x_{i,t}W_i^{\text{ce}} + b_i^{\text{ce}}\right)\right) + \text{PosEnc}(t) \tag{3}$$

where W_i^{ce} and b_i^{ce} are channel specific learnt parameters of the linear projection layers. Similarly, the word embedding is as follows:

$$\hat{y}_u = y_u W^{\text{we}} + b^{\text{we}} + \text{PosEnc}(u) \tag{4}$$

where W^{we} and b^{we} are the weights of a linear layer which are either learned from scratch or pretrained on a large corpus [6, 31].

4.2 Multi-Channel Encoder Layer

Channel-wise Self Attention (cs): Each multi-channel encoder layer starts by learning the contextual relationships within a single channel by utilizing individual self-attention layers (See Figure 2 (left)). As per the original transformer implementation, we use the scaled dot product scoring function in the attention mechanisms. Given embedded source channel representations, $\hat{X}_i$, we obtain

*Q*ueries, *K*eys and *V*alues for the channel i as[5]:

$$\begin{aligned} Q_i^{\text{cs}} &= \hat{X}_i W_i^{\text{cs,q}} + b_i^{\text{cs,q}} \\ K_i^{\text{cs}} &= \hat{X}_i W_i^{\text{cs,k}} + b_i^{\text{cs,k}} \\ V_i^{\text{cs}} &= \hat{X}_i W_i^{\text{cs,v}} + b_i^{\text{cs,v}} \end{aligned} \tag{5}$$

which are then passed to the channel-wise self attention function to have their intra channel contextual relationship modeled as:

$$h_i^{\text{cs}} = \text{softmax}\left(\frac{Q_i^{\text{cs}} \left(K_i^{\text{cs}}\right)^T}{\sqrt{d_m}}\right) V_i^{\text{cs}} \tag{6}$$

where h_i^{CS} is the spatio-temporal representation of the i^{th} source channel and d_m is the hidden size of the model. We also utilize individual feed forward layers as described in [62] for each channel as:

$$\text{FF}(x) = \max\left(0, xW_1^{\text{ff}} + b_1\right) W_2^{\text{ff}} + b_2 \tag{7}$$

By feeding the contextually modeled channel representations through feed forward layers, we obtain the final outputs of the channel-wise attention layer of our multi-channel encoder layer as:

$$\hat{h}_i^{\text{cs}} = \text{FF}_i^{\text{cs}}(h_i^{\text{cs}}) \tag{8}$$

Multi-channel Encoder Attention (me): We now introduce the multi-channel encoder attention, which learns the contextual relationship between the self-attended channel representations (See Figure 2 (middle)). As we are using dot product attention, we start by obtaining Q, K and V for each source as:

$$\begin{aligned} Q_i^{\text{me}} &= \hat{h}_i^{\text{cs}} W_i^{\text{me,q}} + b_i^{\text{me,q}} \\ K_i^{\text{me}} &= \hat{h}_i^{\text{cs}} W_i^{\text{me,k}} + b_i^{\text{me,k}} \\ V_i^{\text{me}} &= \hat{h}_i^{\text{cs}} W_i^{\text{me,v}} + b_i^{\text{me,v}} \end{aligned} \tag{9}$$

These values are then passed to the multi-channel attention layers where the *Q*ueries of each channel are used to estimate the scores over the concatenated *K*eys of the other channels. These scores are then used to calculate the channel-fused representations by taking a weighted sum over the other channels' concatenated *V*alues. More formally, multi-channel attention can be defined as:

$$h_i^{\text{me}} = \text{softmax}\left(\frac{Q_i^{\text{me}} \left(\left[\forall K_j^{\text{me}} \text{ where } j \neq i\right]\right)^T}{\sqrt{d_m}}\right) \left[\forall V_j^{\text{me}} \text{ where } j \neq i\right] \tag{10}$$

We would like to note that, the concatenation operation ([ ]) is performed over the time axis, thus making our approach applicable to tasks where the

[5] Note that we use a vectorized formulation in our equations. All softmax and bias addition operations are done row-wise.

source channels have a different numbers of tokens. We then pass multi-channel attention outputs to individual feed forward layers to obtain the final outputs of the multi-channel encoder layer as:

$$\hat{h}_i^{\text{me}} = \text{FF}_i^{\text{me}}(h_i^{\text{me}}) \tag{11}$$

Several multi-channel encoder layers can be stacked to form the encoder network with the aim of learning more complex multi-channel contextual representations, $h^e = (h_1^e, ..., h_N^e)$, where h_i^e is the output corresponding to the i^{th} source channel.

4.3 Multi-channel Decoder Layer

Transformer networks utilize a masked self attention and an encoder-decoder attention in each decoder layer. The subsequent masking on self-attention is essential, as the target tokens' successors will not be available at inference time. In our approach, we also employ the masked self-attention to model the contextual relationship between target tokens' and its predecessors. However, we replace encoder-decoder attention with multi-channel decoder attention, which is modified to work with multiple source channel representations (See Figure 2 (right)). Given the word embeddings $\hat{\mathcal{Y}}$ of a sentence $\mathcal{Y}$, we first obtain the masked self-attention (sa) outputs $\hat{h}^{\text{sa}}$ using the generic approach [62], which are then in turn passed to our multi-channel decoder attention.

Multi-channel Decoder Attention (md): In generic transformers, encoder-decoder attention *Q*ueries are obtained from the decoder self-attention estimates, $\hat{h}^{\text{sa}}$, while *K*eys and *V*alues are calculated from the final encoder layer outputs, h_e. In order to incorporate information coming from multiple channels using transformer models, we propose the multi-channel decoder attention module. We first obtain the Q, K and V as:

$$\begin{aligned} Q^{\text{md}} &= \hat{h}^{\text{sa}} W^{\text{md,q}} + b^{\text{md,q}} \\ K_i^{\text{md}} &= h_i^e W_i^{\text{md,k}} + b_i^{\text{md,k}} \\ V_i^{\text{md}} &= h_i^e W_i^{\text{md,v}} + b_i^{\text{md,v}} \end{aligned} \tag{12}$$

Note that each source channel i has their own learned *K*ey and *V*alue matrices, $W_i^{md,k}$ and $W_i^{md,v}$ respectively.

These are then passed to the multi-channel decoder attention module where the *Q*ueries of each target token are scored against all channel *K*eys. Channel scores are then used to calculate the weighted average of their respective *V*alues. Individual channel outputs are averaged to obtain the final output of the multi-channel decoder attention module. This process can be formalized as:

$$h^{\text{md}} = \frac{1}{N} \sum_{i=1}^{N} \left(\text{softmax} \left(\frac{Q^{\text{md}} \left(K_i^{\text{md}}\right)^T}{\sqrt{d_m}} \right) V_i^{\text{md}} \right) \tag{13}$$

The attention module outputs are then passed through a feed forward layer to obtain the final representations of the multi-channel decoder layer as:

$$\hat{h}^{\mathrm{md}} = \mathrm{FF}^{\mathrm{md}}(h^{\mathrm{md}}) \tag{14}$$

Like the multi-channel encoder layer, multiple decoder layers can be stacked to improve representation capabilities of the decoder network. The output of the stacked decoder is denoted as $h^d = (h_1, ..., h_U)$ which is used to condition target token generation.

4.4 Loss Functions

We propose training multi-channel transformers using two types of loss function, namely a Translation Loss, which is commonly used in machine translation, and a Channel Anchoring Loss, which aims to preserve channel specific information during encoding.

Translation Loss: Although different loss functions have been used to train translation models, such as a mixture-of-softmaxes [66], token level cross-entropy loss is the most common approach to learn network parameters. Given a source-target pair, the translation loss, $\mathcal{L}_T$, is calculated as the accumulation of the error at each decoding step u, which is estimated using a classification loss over the target vocabulary as:

$$\mathcal{L}_T = 1 - \prod_{u=1}^{U} \sum_{g=1}^{G} p(y_u^g) p(\hat{y}_u^g) \tag{15}$$

where $p(y_u^g)$ and $p(\hat{y}_u^g)$ represent the ground truth and the generation probabilities of the target y^g at decoding step u, respectively, and G is the target language vocabulary size. In our networks, the probability of generating target token y_u at the decoding step u is conditioned on the hidden state of the decoder network h_u^d at the corresponding time step, $p(\hat{y}_u) = p(\hat{y}_u|h_u^d)$. Softmaxed linear projection of h_u^d is used to model the probability of producing tokens over the whole target vocabulary as:

$$p(\hat{y}_u|h_u^d) = \mathrm{softmax}(h_u^d W^o + b^o) \tag{16}$$

where W^o and b^o are the trainable parameters of a linear layer.

Channel Anchoring Loss: For source channels, where we have access to a relevant classifier, we use an anchoring loss to preserve channel specific information. Predictions of these classifiers are used as ground truth to calculate token level cross entropy losses in the same manner as the translation loss. Given the classifier outputs corresponding to the i^{th} channel, $C_i = (c_{i,1}, ..., c_{i,T_i})$, and the hidden state of the encoder, h^e, we first calculate the prediction probabilities over the target channel classes as:

$$p(\hat{c}_{i,t}|h^e) = \mathrm{softmax}(h^e W_i^o + b_i^o) \tag{17}$$

where $p(\hat{c}_{i,t}|h^e)$ represent the prediction probabilities over the i^{th} channel's classifier vocabulary, while W_i^o and b_i^o are the weights and biases of the linear layer used for the i^{th} channel, respectively. We then use a modified version of Equation 15 to calculate the i^{th} channel's anchoring loss, $\mathcal{L}_{A,i}$, as:

$$\mathcal{L}_{A,i} = 1 - \prod_{t=1}^{T} \sum_{g=1}^{G_i} p(c_{i,t}^g) p(\hat{c}_{i,t}^g | h_t^e) \tag{18}$$

where $p(c_{i,t}^g)$ and $p(\hat{c}_{i,t}^g)$ represent the classifier output and the predicted probabilities of the class c_i^d at the encoders t^{th} step, respectively, while G_i is the number of target classes of the classifier corresponding to channel i. For example, one can use a hand shape classifier's convolutional layer as as input channel and the same classifier's predictions as ground truth for hand channel anchoring loss to preserve the hand shape information, as well as to regularize the translation loss.

Total Loss: We use a weighed combination of Translation loss, $\mathcal{L}_T$, and Anchoring losses, $\mathcal{L}_A = (\mathcal{L}_{A,1}, ..., \mathcal{L}_{A,N})$, during training as:

$$\mathcal{L} = \lambda_T \mathcal{L}_T + \lambda_A \sum_{i=1}^{N} \mathcal{L}_{A,i} \tag{19}$$

where λ_T and λ_A decide the importance of each loss function during training.

5 Implementation and Evaluation Details

Dataset: We evaluate our model on the challenging PHOENIX14**T** [13] dataset, which is currently the only publicly available large vocabulary continuous SLT dataset aimed at vision based sign language research.

Sign Channels: We use three different articulators/channels to represent the manual and non-manual features of the sign, namely hand shapes, mouthings and upper body pose. We employ the models proposed and used in [35] and from these networks extracted 1024 dimensional Convolutional Neural Network (CNN) features for each frame (last layer before the fully connected layer) for hand shape and mouthing channels. We use the class prediction from the same network to anchor the channel representations. Although these networks were trained on 61 and 40 hand shapes and mouthing classes respectively (including transition/silence class), the predictions only contained 52 and 36 classes. Hence, our anchoring losses are calculated over the predicted number of classes.

To represent the upper body pose of the signers, we extract 2D skeletal pose information using the OpenPose library [14]. We then employ a 2D-to-3D lifting approach designed specifically for sign language production to obtain the final 3D joint positions of 50 upper body pose joints [67]. As there were no prior subunit classes for the upper body pose on PHOENIX14**T**, we do not utilize an anchoring loss on the pose channel.

Training and Network Details: Our networks are trained using the PyTorch framework [51] with a modified version of the JoeyNMT library [42]. We use Adam [33] optimizer with a batch size of 32, a learning rate of 10^{-3} ($\beta_1 = 0.9, \beta_2 = 0.998$) and a weight decay of 10^{-3}. We utilize Xavier [23] initialization and train all networks from scratch. We do not apply dropout and only use a single headed scaled dot-product attention to reduce the number of hyper-parameters in our experiments.

Decoding: During training we use a greedy search to evaluate development set translation performance. At inference, we employ beam search decoding with the beam width ranging from 0 to 10. We also employ a length penalty as proposed by [30] with α values ranging from 0 to 5. We use the development set to find the best performing beam width and α, and use these during test set inference for final results.

Performance Metrics: We use BLEU [49] and ROUGE [43] scores to measure the translation performance. To give the reader a better understanding of the networks behaviour, we repeat each experiment 10 times and report mean and standard deviation of BLEU-4 and ROUGE scores on both development and test sets. We also report our best result for every setup based on the BLEU-4 score as per the development set. BLEU-4 score is also used as the validation score for our learning scheduler and for early stopping.

6 Experiment Results

In this section we propose several multi-channel SLT experimental setups and report our quantitative results. We start by sharing single channel SLT performance using different network architectures, varying the number of hidden units, both to set a baseline for our multi-channel approaches and to find the optimal network size. After that, we propose two naive channel fusion approaches, namely early fusion and late fusion, to set a fusion benchmark for our novel *Multi-channel Transformer* architecture. Finally, we report the performance of the multi-channel transformer approach with and without the channel anchoring losses and compare our results against the state-of-the-art.

We apply batch normalization [29] and a soft-sign activation function [47] to input channel embeddings before passing them to our networks. See supplementary material for empirical justification for this choice.

6.1 Single Channel Baselines

In the first set of experiments, we train single channel SLT models. The main objective of these experiments is to set translation baselines for all future multi-channel fusion models. However, we would also like to examine the relative information presented in each channel by comparing their translation performance against one another. In addition, we wish to identify the optimal network setup for each channel to guide the future experiments. Therefore, we conduct experiments with four network setups for all three articulators with sizes varying from

Table 1. Single channel SLT baselines using different network architectures.

		Dev Set				Test Set			
		BLEU-4		ROUGE		BLEU-4		ROUGE	
Channel	HSxFF	Best	mean ± std	-	mean ± std	-	mean ± std	-	mean ± std
Hand	32x64	14.54	14.05 ± 0.42	38.47	38.49 ± 0.45	13.88	13.80 ± 0.63	38.05	38.04 ± 0.49
Hand	64x128	**16.44**	15.70 ± 0.41	40.79	40.45 ± 0.64	16.18	15.63 ± 0.65	40.62	40.07 ± 0.80
Hand	128x256	16.32	**15.91** ± 0.43	**41.87**	**41.08** ± 0.73	**16.76**	**16.02** ± 0.88	**41.85**	**40.67** ± 0.93
Hand	256x512	16.06	15.41 ± 0.46	41.46	40.13 ± 0.83	15.43	15.57 ± 0.60	40.48	39.88 ± 0.71
Mouthing	32x64	11.70	11.24 ± 0.34	33.26	32.84 ± 0.60	10.77	11.01 ± 0.34	33.51	33.05 ± 0.34
Mouthing	64x128	12.91	12.55 ± 0.30	36.22	35.17 ± 0.71	12.83	12.62 ± 0.50	35.30	35.04 ± 0.72
Mouthing	128x256	**13.74**	**13.08** ± 0.41	**37.20**	**35.96** ± 0.79	**13.77**	**13.50** ± 0.37	**37.24**	**36.60** ± 0.72
Mouthing	256x512	12.86	12.40 ± 0.42	34.13	34.63 ± 0.64	13.25	12.34 ± 0.51	35.53	34.83 ± 0.47
Pose	32x64	9.64	8.91 ± 0.42	30.27	29.92 ± 0.67	9.64	8.55 ± 0.62	30.03	29.13 ± 0.79
Pose	64x128	10.64	10.28 ± 0.26	31.06	31.31 ± 0.47	9.97	9.88 ± 0.18	29.63	30.44 ± 0.60
Pose	128x256	**11.02**	**10.52** ± 0.31	**32.22**	**31.85** ± 0.42	**10.26**	**10.03** ± 0.46	**30.44**	**30.79** ± 0.82
Pose	256x512	10.06	9.50 ± 0.51	31.03	30.11 ± 0.75	9.51	8.62 ± 0.70	29.92	28.65 ± 0.99
Gloss	32x64	17.21	16.03 ± 0.49	42.20	41.26 ± 0.64	15.45	15.68 ± 0.43	41.21	40.71 ± 0.42
Gloss	64x128	18.50	18.16 ± 0.23	44.99	43.87 ± 0.68	18.14	17.89 ± 0.56	43.57	43.02 ± 0.69
Gloss	128x256	19.43	**19.14** ± 0.36	**46.10**	**45.17** ± 0.63	19.52	**19.08** ± 0.48	**45.32**	**44.52** ± 0.80
Gloss	256x512	**19.52**	18.36 ± 0.50	45.97	44.16 ± 0.79	**19.61**	18.60 ± 0.63	45.29	43.92 ± 0.98

32x64 to 256x512 (hidden size (HS) x number of feed forward (FF) units). All networks were built using two encoder and decoder layers.

As can be seen in Table 1, **H**and is the best performing channel in all network setups. Furthermore, using a network setup of 128x256 outperforms all of the alternatives. We believe this is closely related to the limited number of training samples we have and the over-fitting issues that come with it. Therefore, for the rest of our experiments we use 128x256 parameters for each channel.

We further train a **G**loss single channel network to set a baseline for our multi-channel approaches to compare against. As shown in Table 1, using CNN features that were trained using gloss level annotations outperforms all single sign articular based models (19.52 *vs.* 16.44 dev BLEU-4 score). Although the 256x512 network setup obtained the best individual development and test set translation performances, the mean performance of the 128x256 network was better, encouraging us to utilize this setup going forward.

6.2 Early and Late Fusion of Sign Channels

To set another benchmark for our multi-channel transformer, we propose two naive multi-channel fusion approaches, namely early and late fusion. In the early fusion setup, features from different channels are concatenated to create a fused representation of each frame. These representations are then used to train SLT models, as if they were features coming from a single channel. Hence, the contextual relationship is performed in an implicit manner by the transformer architecture. In our second, late fusion setup, individual SLT models are built which are then fused at the decoder output level, *i.e.* h^d, by concatenation. The fused representation is then used to generate target tokens using a linear projection layer. Compared to early fusion, this approach's capability to learn more abstract relationships is limited as the fusion is only done by a single linear layer. We examine all four possible fusions of the three channels. Network setup is set to linearly scale with respect to the number of channels that are fused together with a factor of 128x256 per channel.

Table 2. SLT performance of early and late channel fusion approaches.

			Dev Set				Test Set			
			BLEU-4		ROUGE		BLEU-4		ROUGE	
Fusion	Channels	HSxFF	Best	mean ± std	-	mean ± std	-	mean ± std	-	mean ± std
Early	**H** + **M**	2*(128x256)	**17.25**	**16.73** ± 0.57	**42.04**	**41.72** ± 0.61	**17.37**	**16.73** ± 0.82	**42.35**	**41.76** ± 0.80
Early	**H** + **P**	2*(128x256)	16.17	15.70 ± 0.32	40.51	40.28 ± 0.46	15.75	15.83 ± 0.30	40.54	40.34 ± 0.70
Early	**M** + **P**	2*(128x256)	13.57	12.91 ± 0.30	36.43	35.88 ± 0.40	13.23	13.02 ± 0.42	35.66	36.01 ± 0.72
Early	**H** + **M** + **P**	3*(128x256)	15.69	15.08 ± 0.55	39.78	39.38 ± 0.73	15.19	15.19 ± 0.49	39.99	39.43 ± 0.80
Late	**H** + **M**	2*(128x256)	**17.03**	**16.36** ± 0.48	41.69	**41.58** ± 0.79	16.81	**16.67** ± 0.49	41.69	**41.69** ± 0.54
Late	**H** + **P**	2*(128x256)	16.61	16.16 ± 0.33	41.54	41.18 ± 0.58	15.90	16.07 ± 0.76	40.81	40.50 ± 0.99
Late	**M** + **P**	2*(128x256)	14.22	13.55 ± 0.31	36.44	37.03 ± 0.64	14.11	13.65 ± 0.49	36.25	36.95 ± 0.65
Late	**H** + **M** + **P**	3*(128x256)	17.00	16.35 ± 0.38	**42.09**	41.29 ± 0.42	**16.95**	16.50 ± 0.47	**42.12**	41.53 ± 0.52

As can be seen in Table 2, fusion of **H**ands and **M**outh yields slightly better results than single channel translation models (excluding gloss). However, unlike late fusion, which saw improvement in all scenarios, early fusion's performance gets worse as more features are added to the network. As this means having more parameters in our networks, we believe this is due to the natural propensity of the transformers to over-fit on small training datasets, like ours.

6.3 Multi-channel Transformers

In this set of experiments we examine the translation performance of the proposed multi-channel transformer architecture for multi-articulatory SLT. We first start by investigating the effects of the anchoring loss. We then compare our best performing method against other fusion options, gloss based translation and other state-of-the-art methods. As with other fusion experiments, we examine all possible fusion combinations. In addition to using the 128x256 network setup, we also evaluate having a larger network to see if the additional anchoring losses help with over-fitting by regularizing the translation loss.

As can be seen in the first row of Table 3, while using the same number of parameters as the early and late fusion setups, our proposed *Multi-Channel Transformer* approach outperforms both configurations. However, doubling the network size does effect the direct application of multi-channel attention negatively. To counteract this issue and to examine the effects of the anchoring loss, we run experiments with both 128x256 and 256x512 setups. We normalize our losses on the sequence level instead of token level and we set the anchoring loss weight, λ_A, to 0.15 to counteract different source (video) and target (sentence) sequence lengths. Using the anchoring losses not only improves the performance of the 128x256 models but also allows the 256x512 networks to achieve similar translation performance to using gloss features. We believe this is due to two main factors. Firstly, the anchoring loss forces the encoder channels to preserve the channel specific information while being contextually modeled against other articulators. Secondly, it acts as a regularizer for the translation loss and counteracts the over-fitting previously discussed.

Compared to the state-of-the-art, our best multi-channel transformer model surpasses the performance of several previous models [48, 13], some of which are heavily reliant on gloss annotation. Furthermore, our multi-channel models perform *on par* with our single gloss channel model, and yields competitive

Table 3. Multi-channel Transformer based multi-articulatory SLT results.

			Dev Set				Test Set			
			BLEU-4		ROUGE		BLEU-4		ROUGE	
Channels	Anchoring Loss	HSxFF	Best	mean ± std	-	mean ± std	-	mean ± std	-	mean ± std
H + M	✗	2*(128x256)	17.71	16.97 ± 0.53	43.43	42.02 ± 0.86	17.72	17.19 ± 0.73	42.70	41.95 ± 0.85
H + P	✗	2*(128x256)	17.20	16.36 ± 0.58	42.15	41.23 ± 0.46	16.41	16.25 ± 0.66	40.56	40.87 ± 0.67
M + P	✗	2*(128x256)	14.17	13.50 ± 0.40	36.82	36.62 ± 0.52	13.43	13.93 ± 0.44	37.03	37.43 ± 0.65
H + M + P	✗	3*(128x256)	17.98	16.89 ± 0.59	44.01	41.85 ± 0.93	17.15	16.85 ± 0.65	42.38	41.83 ± 0.85
H + M	✗	2*(256x512)	15.95	15.46 ± 0.34	41.01	40.31 ± 0.50	15.87	15.80 ± 0.36	40.30	40.40 ± 0.70
H + P	✗	2*(256x512)	15.41	14.95 ± 0.33	40.10	39.22 ± 0.53	15.91	15.14 ± 0.59	40.24	39.11 ± 0.69
M + P	✗	2*(256x512)	13.39	12.60 ± 0.49	35.70	35.01 ± 0.73	13.38	12.74 ± 0.48	36.89	35.39 ± 0.79
H + M + P	✗	3*(256x512)	15.87	14.97 ± 0.51	40.53	39.65 ± 0.79	16.02	15.17 ± 0.81	40.15	39.61 ± 1.12
H + M	✓	2*(128x256)	18.52	17.93 ± 0.39	44.56	43.25 ± 0.57	17.93	17.76 ± 0.49	43.21	42.91 ± 0.49
H + P	✓	2*(128x256)	17.70	16.53 ± 0.67	43.19	41.26 ± 0.99	16.93	16.41 ± 0.48	42.62	40.80 ± 0.82
M + P	✓	2*(128x256)	15.14	14.60 ± 0.32	38.57	38.45 ± 0.45	15.32	15.05 ± 0.72	38.47	38.66 ± 0.76
H + M + P	✓	3*(128x256)	18.80	17.81 ± 0.68	44.24	43.17 ± 0.81	18.30	17.75 ± 0.58	43.65	42.90 ± 0.62
H + M	✓	2*(256x512)	19.05	18.07 ± 0.44	45.04	43.76 ± 0.82	**19.21**	17.71 ± 0.72	**45.05**	43.29 ± 0.99
H + P	✓	2*(256x512)	16.80	16.29 ± 0.36	41.15	40.86 ± 0.42	16.68	16.29 ± 0.48	41.34	40.68 ± 0.76
M + P	✓	2*(256x512)	15.14	14.60 ± 0.26	39.45	38.47 ± 0.62	15.36	15.13 ± 0.44	40.06	39.09 ± 0.62
H + M + P	✓	**3*(256x512)**	**19.51**	**18.66** ± 0.52	**45.90**	**44.30** ± 0.92	18.51	**18.31** ± 0.57	43.57	**43.75** ± 0.63
Gloss	–	128x256	19.43	**19.14** ± 0.36	**46.10**	**45.17** ± 0.63	19.52	**19.08** ± 0.48	**45.32**	**44.52** ± 0.80
Gloss	–	**256x512**	**19.52**	18.36 ± 0.50	45.97	44.16 ± 0.79	**19.61**	18.60 ± 0.63	45.29	43.92 ± 0.98
Orbay *et al.* [48]	–	–	–	–	–	–	14.56	–	38.05	–
Sign2Gloss→Gloss2Text [13]	–	–	17.89	–	43.76	–	17.79	–	43.45	–
Sign2Gloss2Text [13]	–	–	18.40	–	44.14	–	18.13	–	43.80	–
(Gloss) Sign2Text [12]	–	**3x(512x2048)**	**20.69**	–	–	–	**20.17**	–	–	–
(Gloss) Sign2(Gloss+Text) [12]	–	**3x(512x2048)**	**22.38**	–	–	–	**21.32**	–	–	–

translation performance to the state-of-the-art transformer based approaches [12], which utilize larger models and uses gloss supervision on several levels (pre-trained Gloss CNN features and transformer encoder supervision). However, due to their dependence on gloss annotations, such models [13, 12] can not be scaled to larger un-annotated datasets, which is not a limiting factor for the proposed multi-channel transformer approach. See supplementary material for qualitative translation examples from our best multi-articulatory translation model.

7 Conclusion

This paper presented a novel approach to Neural Machine Translation in the context of sign language. Our novel multi-channel transformer architecture allows both the inter and intra contextual relationship between different asynchronous channels to be modelled within the transformer network itself. Experiments on RWTH-PHOENIX-Weather-2014T dataset demonstrate the approach achieves *on par* or competitive performance against the state-of-the-art. More importantly, we overcome the reliance on gloss information which underpins other state-of-the-art approaches. Now we have broken the dependency upon gloss information, future work will be to scale learning to larger dataset where gloss information is not available, such as broadcast footage.

Acknowledgements

This work received funding from the SNSF Sinergia project 'SMILE' (CRSII2_160811), the European Union's Horizon2020 research and innovation programme under grant agreement no. 762021 'Content4All' and the EPSRC project 'ExTOL' (EP/R03298X/1). This work reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains. We would also like to thank NVIDIA Corporation for their GPU grant.

References

1. Afouras, T., Chung, J.S., Senior, A., Vinyals, O., Zisserman, A.: Deep Audio-visual Speech Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2018)
2. Antonakos, E., Pitsikalis, V., Rodomagoulakis, I., Maragos, P.: Unsupervised Classification of Extreme Facial Events Using Active Appearance Models Tracking for Sign Language Videos. In: Proceedings of the IEEE International Conference on Image Processing (ICIP) (2012)
3. Assael, Y.M., Shillingford, B., Whiteson, S., De Freitas, N.: Lipnet: End-to-end Sentence-level Lipreading. In: GPU Technology Conference (2017)
4. Bahdanau, D., Cho, K., Bengio, Y.: Neural Machine Translation by Jointly Learning to Align and Translate. In: Proceedings of the International Conference on Learning Representations (ICLR) (2015)
5. Bellugi, U., Fischer, S.: A comparison of sign language and spoken language. Cognition **1**(2) (1972)
6. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics (ACL) **5** (2017)
7. Bragg, D., Koller, O., Bellard, M., Berke, L., Boudrealt, P., Braffort, A., Caselli, N., Huenerfauth, M., Kacorri, H., Verhoef, T., Vogler, C., Morris, M.R.: Sign Language Recognition, Generation, and Translation: An Interdisciplinary Perspective. In: Proceedings of the International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS) (2019)
8. Brentari, D.: Sign Language Phonology. Cambridge University Press (2019)
9. Bungeroth, J., Ney, H.: Statistical Sign Language Translation. In: Proceedings of the Workshop on Representation and Processing of Sign Languages at International Conference on Language Resources and Evaluation (LREC) (2004)
10. Camgoz, N.C., Hadfield, S., Koller, O., Bowden, R.: Using Convolutional 3D Neural Networks for User-Independent Continuous Gesture Recognition. In: Proceedings of the IEEE International Conference on Pattern Recognition Workshops (ICPRW) (2016)
11. Camgoz, N.C., Hadfield, S., Koller, O., Bowden, R.: SubUNets: End-to-end Hand Shape and Continuous Sign Language Recognition. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
12. Camgoz, N.C., Hadfield, S., Koller, O., Bowden, R.: Sign Language Transformers: Joint End-to-end Sign Language Recognition and Translation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
13. Camgoz, N.C., Hadfield, S., Koller, O., Ney, H., Bowden, R.: Neural Sign Language Translation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
14. Cao, Z., Hidalgo, G.M., Simon, T., Wei, S., Sheikh, Y.: OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2019)
15. Caridakis, G., Asteriadis, S., Karpouzis, K.: Non-Manual Cues in Automatic Sign Language Recognition. Personal and ubiquitous computing **18**(1) (2014)
16. Chai, X., Li, G., Lin, Y., Xu, Z., Tang, Y., Chen, X., Zhou, M.: Sign Language Recognition and Translation with Kinect. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition (FG) (2013)

17. Charles, J., Pfister, T., Everingham, M., Zisserman, A.: Automatic and Efficient Human Pose Estimation for Sign Language Videos. International Journal of Computer Vision (IJCV) **110**(1) (2014)
18. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (2014)
19. Chung, J.S., Senior, A., Vinyals, O., Zisserman, A.: Lip Reading Sentences in the Wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
20. Cui, R., Liu, H., Zhang, C.: Recurrent Convolutional Neural Networks for Continuous Sign Language Recognition by Staged Optimization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
21. Cui, R., Liu, H., Zhang, C.: A Deep Neural Framework for Continuous Sign Language Recognition by Iterative Training. IEEE Transactions on Multimedia (2019)
22. Fang, B., Co, J., Zhang, M.: DeepASL: Enabling Ubiquitous and Non-Intrusive Word and Sentence-Level Sign Language Translation. In: Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys) (2017)
23. Glorot, X., Bengio, Y.: Understanding the Difficulty of Training Deep Feedforward Neural Networks. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (2010)
24. Grimes, G.J.: Digital Data Entry Glove Interface Device (1983), uS Patent 4,414,537
25. Guo, D., Wang, S., Tian, Q., Wang, M.: Dense Temporal Convolution Network for Sign Language Translation. In: Proceedings of the AAAI Conference on Artificial Intelligence (2019)
26. Guo, D., Zhou, W., Li, H., Wang, M.: Hierarchical LSTM for Sign Language Translation. In: Proceedings of the AAAI Conference on Artificial Intelligence (2018)
27. Hidalgo, G., Raaj, Y., Idrees, H., Xiang, D., Joo, H., Simon, T., Sheikh, Y.: Single-Network Whole-Body Pose Estimation. In: IEEE International Conference on Computer Vision (ICCV) (2019)
28. Huang, J., Zhou, W., Zhang, Q., Li, H., Li, W.: Video-based Sign Language Recognition without Temporal Segmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence (2018)
29. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: Proceedings of the International Conference on Machine Learning (ICML) (2015)
30. Johnson, M., Schuster, M., Le, Q.V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., et al.: Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. Transactions of the Association for Computational Linguistics **5** (2017)
31. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of Tricks for Efficient Text Classification. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (ACL) (2017)
32. Kalchbrenner, N., Blunsom, P.: Recurrent Continuous Translation Models. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (2013)
33. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In: Proceedings of the International Conference on Learning Representations (ICLR) (2014)
34. Ko, S.K., Kim, C.J., Jung, H., Cho, C.: Neural Sign Language Translation based on Human Keypoint Estimation. Applied Sciences **9**(13) (2019)

35. Koller, O., Camgoz, N.C., Bowden, R., Ney, H.: Weakly Supervised Learning with Multi-Stream CNN-LSTM-HMMs to Discover Sequential Parallelism in Sign Language Videos. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2019)
36. Koller, O., Forster, J., Ney, H.: Continuous Sign Language Recognition: Towards Large Vocabulary Statistical Recognition Systems Handling Multiple Signers. Computer Vision and Image Understanding (CVIU) **141** (2015)
37. Koller, O., Ney, H., Bowden, R.: Read My Lips: Continuous Signer Independent Weakly Supervised Viseme Recognition. In: European Conference on Computer Vision (ECCV) (2014)
38. Koller, O., Ney, H., Bowden, R.: Deep Learning of Mouth Shapes for Sign Language. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW) (2015)
39. Koller, O., Ney, H., Bowden, R.: Deep Hand: How to Train a CNN on 1 Million Hand Images When Your Data Is Continuous and Weakly Labelled. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
40. Koller, O., Zargaran, S., Ney, H.: Re-Sign: Re-Aligned End-to-End Sequence Modelling with Deep Recurrent CNN-HMMs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
41. Koller, O., Zargaran, S., Ney, H., Bowden, R.: Deep Sign: Hybrid CNN-HMM for Continuous Sign Language Recognition. In: Proceedings of the British Machine Vision Conference (BMVC) (2016)
42. Kreutzer, J., Bastings, J., Riezler, S.: Joey NMT: A minimalist NMT toolkit for novices. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations (2019)
43. Lin, C.Y.: ROUGE: A Package for Automatic Evaluation of Summaries. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics, Text Summarization Branches Out Workshop (2004)
44. Luong, M.T., Pham, H., Manning, C.D.: Effective Approaches to Attention-based Neural Machine Translation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (2015)
45. Luzardo, M., Karppa, M., Laaksonen, J., Jantunen, T.: Head Pose Estimation for Sign Language Video. Image Analysis (2013)
46. Malaia, E., Borneman, J.D., Wilbur, R.B.: Information Transfer Capacity of Articulators in American Sign Language. Language and Speech **61**(1) (2018)
47. Nwankpa, C., Ijomah, W., Gachagan, A., Marshall, S.: Activation Functions: Comparison of trends in practice and research for deep learning. arXiv:1811.03378 (2018)
48. Orbay, A., Akarun, L.: Neural Sign Language Translation by Learning Tokenization. arXiv:2002.00479 (2020)
49. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: A Method for Automatic Evaluation of Machine Translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL) (2002)
50. Parton, B.S.: Sign Language Recognition and Translation: A Multidisciplined Approach From the Field of Artificial Intelligence. The Journal of Deaf Studies and Deaf Education **11**(1) (2005)
51. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic Differentiation in PyTorch. In: Proceedings of the Advances in Neural Information Processing Systems Workshops (NIPSW) (2017)

52. Pfau, R., Quer, J.: Nonmanuals: Their Grammatical and Prosodic Roles. In: Sign Languages. Cambridge University Press (2010)
53. Pfister, T., Charles, J., Everingham, M., Zisserman, A.: Automatic and Efficient Long Term Arm and Hand Tracking for Continuous Sign Language TV Broadcasts. In: Proceedings of the British Machine Vision Conference (BMVC) (2012)
54. Pigou, L., Dieleman, S., Kindermans, P.J., Schrauwen, B.: Sign Language Recognition using Convolutional Neural Networks. In: European Conference on Computer Vision Workshops (ECCVW) (2014)
55. Starner, T., Weaver, J., Pentland, A.: Real-time American Sign Language Recognition using Desk and Wearable Computer Based Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **20**(12) (1998)
56. Stein, D., Schmidt, C., Ney, H.: Sign Language Machine Translation Overkill. In: International Workshop on Spoken Language Translation (2010)
57. Stokoe, W.C.: Sign Language Structure. Annual Review of Anthropology **9**(1) (1980)
58. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to Sequence Learning with Neural Networks. In: Proceedings of the Advances in Neural Information Processing Systems (NIPS) (2014)
59. Sutton-Spence, R., Woll, B.: The Linguistics of British Sign Language: An Introduction. Cambridge University Press (1999)
60. Tamura, S., Kawasaki, S.: Recognition of Sign Language Motion Images. Pattern Recognition **21**(4) (1988)
61. Valli, C., Lucas, C.: Linguistics of American Sign Language: An Introduction. Gallaudet University Press (2000)
62. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is All You Need. In: Proceedings of the Advances in Neural Information Processing Systems (NIPS) (2017)
63. Vogler, C., Goldenstein, S.: Facial Movement Analysis in ASL. Universal Access in the Information Society **6**(4) (2008)
64. Wang, S., Guo, D., Zhou, W.g., Zha, Z.J., Wang, M.: Connectionist Temporal Fusion for Sign Language Translation. In: Proceedings of the ACM International Conference on Multimedia (2018)
65. Wilbur, R.B.: Phonological and Prosodic Layering of Nonmanuals in American Sign Language. The Signs of Language Revisited: An Anthology to Honor Ursula Bellugi and Edward Klima (2000)
66. Yang, Z., Dai, Z., Salakhutdinov, R., Cohen, W.W.: Breaking the Softmax Bottleneck: A High-Rank RNN Language Model. In: Proceedings of the International Conference on Learning Representations (ICLR) (2018)
67. Zelinka, J., Kanis, J., Salajka, P.: NN-Based Czech Sign Language Synthesis. In: International Conference on Speech and Computer (2019)
68. Zhou, H., Zhou, W., Zhou, Y., Li, H.: Spatial-Temporal Multi-Cue Network for Continuous Sign Language Recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence (2020)

Multi-channel Transformers for Multi-articulatory Sign Language Translation: Supplementary Material

Necati Cihan Camgoz[1], Oscar Koller[2], Simon Hadfield[1], and Richard Bowden[1]

[1] CVSSP, University of Surrey, UK, {n.camgoz, s.hadfield, r.bowden}@surrey.ac.uk,
[2] Microsoft, Munich, Germany, oscar.koller@microsoft.com

In this supplementary material, we report our quantitative experiment results for finding the best channel feature and word embedding setup. We also share qualitative translation samples produced by our best performing Multi-channel transformer model.

1 Channel Feature and Word Embeddings

In this preliminary set of experiments we investigate different ways to embed Convolutional Neural Network (CNN) based channel features and one-hot word vectors. Machine translation orientated transformer implementations either use pretrained word embeddings or train a linear projection layer from scratch. Although not stated in the original paper [3], the official transformer implementation also utilizes *embedding scaling*, where the projected word representations are multiplied by a constant which is the square root of the hidden size[3].

Compared to the one-hot vectors which have a constant scale between $[0-1]$, CNN features can have an arbitrary scale. To see how important the input scale is and to examine the effects of the different embedding setups, we initially trained translation networks that only used hand channel features as input. Each network has two layers, with a hidden size of 64 and 128 position-wise feed forward units.

As can been in the first row of Table 1, the translation performance degrades drastically when we apply the commonly used embedding scaling on either of the embeddings. We have experimentally found that transformer networks are extremely sensitive to input scale and this is substantiated by these results. Thus, in our next set of experiments we investigate ways to normalize inputs to control their scale. To do so, we utilize batch normalization [1] and soft-sign activation [2]. While batch normalization scales the inputs between $[-3, 3]$ it has also been shown to improve convergence rate. On the other hand, soft-sign activation scales the inputs between $[-1, 1]$ while also enhancing the representation capability of the embedding due to its non-linear nature.

Individually, both batch norm and soft-sign significantly improve the translation performance when applied to the projected CNN features (see second and third rows of Table 1). We then investigate their combined use on both CNN features and word embeddings. Although there were several comparable setups, we

[3] github.com/tensorflow/models/blob/master/official/nlp/transformer/

concur that the joint application of batch norm and soft-sign only on the CNN features yield the most stable and balanced performance in terms of development and test set for BLEU-4 and ROUGE scores. We believe this is due to the already scaled nature of the word embeddings and the additional stability and non-linearity introduced by applying soft-sign and batch norm to the projected CNN features. Therefore, we utilize this embedding setup for our experiments in the main manuscript.

Table 1. Effects of using different embedding setups on hand channel features to spoken language translation performance.

Feature Embedding			Word Embedding			Dev Set				Test Set			
						BLEU-4		ROUGE		BLEU-4		ROUGE	
Scaling	BatchNorm	SoftSign	Scaling	Batch Norm	Soft-Sign	Best	mean ± std	-	mean ± std	-	mean ± std	-	mean ± std
✗	-	-	✗	-	-	**15.46**	**15.08** ± 0.25	**40.57**	**39.39** ± 0.55	**15.98**	**15.23** ± 0.54	**39.97**	**39.44** ± 0.80
✓	-	-	✗	-	-	13.96	13.39 ± 0.36	37.51	37.07 ± 0.33	14.29	13.77 ± 0.47	37.91	37.54 ± 0.60
✗	-	-	✓	-	-	14.54	14.20 ± 0.26	37.86	38.02 ± 0.70	14.80	14.58 ± 0.46	38.35	38.38 ± 0.72
✓	-	-	✓	-	-	13.52	12.88 ± 0.36	36.93	36.14 ± 0.66	13.99	13.29 ± 0.53	37.72	36.58 ± 0.81
✗	✓	-	✗	✗	-	**16.35**	**15.64** ± 0.46	**41.30**	**40.41** ± 0.55	**16.09**	**15.60** ± 0.50	**40.89**	**40.15** ± 0.70
✗	✗	-	✗	✓	-	14.49	14.07 ± 0.25	37.96	37.86 ± 0.58	14.72	14.33 ± 0.38	37.49	37.53 ± 0.43
✗	✓	-	✗	✓	-	15.17	14.69 ± 0.30	39.92	39.10 ± 0.56	15.50	14.99 ± 0.64	39.83	39.26 ± 0.81
✗	-	✓	✗	-	✗	**16.40**	**15.74** ± 0.55	**41.90**	**40.73** ± 0.63	14.95	**15.63** ± 0.48	39.31	39.93 ± 0.57
✗	-	✗	✗	-	✓	15.75	15.10 ± 0.35	39.72	39.47 ± 0.46	**16.33**	15.41 ± 0.55	40.74	39.76 ± 0.67
✗	-	✓	✗	-	✓	15.98	15.50 ± 0.35	41.02	40.24 ± 0.51	15.64	15.49 ± 0.48	**40.79**	**40.02** ± 0.59
✗	✓	✓	✗	✗	✗	**16.44**	**15.70** ± 0.41	40.79	40.45 ± 0.64	**16.18**	15.63 ± 0.65	**40.62**	40.07 ± 0.80
✗	✗	✗	✗	✓	✓	15.15	14.18 ± 0.42	39.30	37.78 ± 0.77	15.14	14.30 ± 0.53	38.41	37.53 ± 0.82
✗	✓	✓	✗	✓	✓	15.59	15.04 ± 0.33	39.83	39.40 ± 0.46	14.85	14.99 ± 0.51	39.26	39.30 ± 0.65
✗	✓	✓	✗	✗	✓	15.98	15.62 ± 0.21	**41.63**	**40.63** ± 0.62	15.97	**15.65** ± 0.49	40.54	**40.24** ± 0.74
✗	✗	✓	✗	✓	✓	15.09	14.70 ± 0.26	39.36	38.95 ± 0.25	14.70	14.80 ± 0.49	39.38	38.87 ± 0.67

2 Qualitative Examples

In this section we share translation examples generated by our best performing model. As the ground truth spoken language annotations of the RWTH-PHOENIX-Weather-2014**T** (PHOENIX14**T**) dataset are in German, we share both the original German translations and their equivalent word-by-word translations in English. As can be seen in Table 2, we also categorize the results into three categories, namely *Good*, *Mediocre* and *Poor* translations, to give further insight to the reader on the limitations of the current approach.

We categorize translations as *Good* or *Mediocre* when the produced sentences convey the same or similar information as the reference sentences. These examples follow the standard grammar with few exceptions. We classify translations as *Poor* when the model fails to understand and translate the conveyed information in sign videos. Most of these examples contain repetitions. In some cases, the model is not able distinguish some sign glosses from another, such as named entities like locations or numbers which occur in limited contexts in the training data. One way to address this issue might be to utilize pretrained spoken language models to improve the produced translations.

Table 2. Spoken language translations produced by our best Multi-Channel Transformer model.

Good Translations:
Reference: und nun die wettervorhersage für morgen dienstag den ersten februar . (and now the weather forecast for tomorrow tuesday the first of february .) Ours: und nun die wettervorhersage für morgen dienstag den ersten februar . (and now the weather forecast for tomorrow tuesday the first of february .)
Reference: der sorgt wieder für wolken die regen im bergland auch schnee bringen . (it provides clouds again and the rain in the mountains also brings snow .) Ours: im übrigen land fällt gebietsweise regen im bergland auch schnee . (in the rest of the country there is rain in the mountains and snow in some areas.)
Reference: die neue woche beginnt wechselhaft und kühler . (the next week starts variable and colder .) Ours: auch am montag wechselhaft und deutlich kühler . (also on monday variable and significantly colder .)
Mediocre Translations:
Reference: ab sonntag wird es wieder milder dabei gibt es viele wolken zeitweise fällt regen im nordwesten windig . (from sunday on it will be more mild with many clouds partly rain in the northwest windy .) Ours: am sonntag mehr wolken als sonne hier und da regen im westen ist es windig . (on sunday more clouds than sun from time to time rain in the west windy .)
Reference: im osten und südosten auch schnee oder schneeregen . (in the east and south-east also snow or sleet .) Ours: im südosten schnee oder schnee . (snow or snow in the south-east .)
Reference: westlich des rheins und im nordosten bleibt es meist trocken . (west of the rhine and in the northeast it remains mostly dry .) Ours: im westen und südwesten bleibt es noch im nordosten trocken . (in the west and southwest it remains still dry in the northeast .)
Poor Translations:
Reference: deutschland liegt morgen unter hochdruckeinfluss der die wolken weitgehend vertreibt . (germany will be under the influence of high pressure tomorrow which will largely dispel the clouds .) Ours: deutschland liegt morgen über deutschland nach deutschland . (germany is tomorrow over germany to germany .)
Reference: am freitag insgesamt viele wolken die regen bringen . (on friday overall many clouds bringing rain .) Ours: am freitag gibt es am freitag viele wolken . (on friday there are on friday many clouds)
Reference: dazu weht ein starker wind vor allen dingen wieder über vorpommern aus südost . (in addition a strong wind blows before all things again over vorpommern from southeast .) Ours: es weht ein kräftiger nordostwind . (a strong north-easterly wind is blowing .)

References

1. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: Proceedings of the International Conference on Machine Learning (ICML) (2015)
2. Nwankpa, C., Ijomah, W., Gachagan, A., Marshall, S.: Activation Functions: Comparison of trends in practice and research for deep learning. arXiv:1811.03378 (2018)
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is All You Need. In: Proceedings of the Advances in Neural Information Processing Systems (NIPS) (2017)