End-To-End Sign Language Translation via Multitask Learning

Anonymous ACL submission

Abstract

Sign language translation (SLT) is usually seen as a two-step process of continuous sign language recognition (CSLR) and gloss-to-text translation. We propose a novel, Transformer-based (Vaswani et al., 2017) architecture to jointly perform CSLR and sign-translation in an end-to-end fashion. We extend the ordinary Transformer decoder with two channels to support multitasking, where each channel is devoted to solve a particular problem. To control the memory footprint of our model, channels are designed to share most of their parameters among each other. However, each channel still has a dedicated set of parameters which is fine-tuned with respect to the channel's task. In order to evaluate the proposed architecture, we focus on translating German signs into English sequences and use the `RWTH-PHOENIX-Weather 2014 T` corpus in our experiments. Evaluation results indicate that the mixture of information provided by the multitask decoder was successful and enabled us to achieve superior performance in comparison to other SLT models.

1 Introduction

Sign languages (SLs) are the main medium of communication for people with hearing problems. In such languages, linguistic phenomena are in conjunction with other factors such as body movements, poses, and facial expressions. Accordingly, existing tools designed to process spoken languages are not directly applicable to SLs. It involves translating sign videos to a target language and this makes this task relatively harder compared to traditional Neural Machine Translation (NMT) task. In this paper, we particularly focus on *translating* these languages and propose a tailored solution to interpret signs from video frames and translate them into text sequences in a target language.

One approach to SLT is to view the process as a combination of three tasks, *viz.* sign segmentation, sign language recognition (SLR), and gloss-to-word translation. In text sequences, punctuation marks and white spaces help segment them into fundamental units. Silent regions, namely pauses, between phonemes play the same role in speech processing tasks (van Hemert, 1991). However, the task of segmentation is not very straightforward when working with SLs and a SL processing task may require some sort of segmentation (Santemiz et al., 2009; Khan et al., 2014). The purpose of sign segmentation is to be clear about the input units, their boundaries, and see how to feed the model. Once segmentation is completed, a next step would be understanding/recognizing information carried out by signs, which is referred to as SLR in the literature. What SLR generates is a sequence of special tokens known as sign language glosses. The final step, translation, takes glosses and transforms them into words in the target language.

Performing each of these tasks separately requires dedicated models and datasets, which could be quite challenging. Camgoz et al. (2020) proposed a much simpler but more effective solution. They treated the aforementioned three-step pipeline as an end-to-end process of transforming video frames into target-language words and show that their approach can in-fact outperform other conventional methods. In their model, SLT is carried out via a single neural network and there is no clear step defined for segmentation or SLR. The network, itself, decides how to set boundaries and use information stored in video frames to accomplish the task.

Our approach to SLT is also to develop an end-to-end model. We propose a Transformer (Vaswani et al., 2017) model which relies on multitasking. Similar to Camgoz et al. (2020), we do not feed our model with segmented units and let the network decide how to process the video frames. However, on the target side (i.e, on the decoder side), we explicitly force the model to generate sign glosses and transcribe source signs into a target language. This form of training defines a better objective for the network, and it clearly learns what input video frames are processed for and how internal representations should be generated in order to serve the target tasks. Camgoz et al. (2020) and other similar models only provide the network with one generic task/objective (to perform SLT), whereas we decompose it into more tangible and detailed goals, and this is the main distinctive feature about our model.

Our aim for using multi-task learning is based upon exploiting the representation bias in the dataset, which helps the model to learn better internal representations that related tasks might prefer. Specifically, our proposed method is based on the hard parameter sharing paradigm for multi-tasking (Caruana, 1993), where tasks specific layers are placed after the hidden shared layers. For fair comparison with our proposed hard parameter sharing based model, we also train a baseline model ($D_{SEP}++$), which implements the soft parameter sharing paradigm of multi-task learning framework.

Our main contribution in this work can be summarized as follows:

- Exploiting available gloss sequence at both encoder and decoder side effectively, which performs better than the prior state-of-the-art (Camgoz et al., 2020). In the unavailability of gloss sequence, We use a multitasking objective, where beside decoding source sign into fixed target language (i.e, German); we also translate source sign into a different target language (i.e, English). To train our decoder, we translate target side German sentence into English via an NMT model. This auxiliary multitasking objective outperforms baseline transformer.

- Our proposed approach is task agnostic and similar multitasking objectives can be applied for the other tasks too.

2 Related Work

The SLT systems were introduced in the early 2000s (Bungeroth and Ney, 2004) where language models were used to construct sentences by recognizing the isolated signs(Chai et al., 2013). However, there was no sign of directly converting videos into sentences i.e., end-to-end SLT system until recently. For the SLT system, a large annotated dataset is required but creation and annotation of sign videos is a laborious task. A few datasets from linguistic sources (Hanke et al., 2010; Schembri et al., 2013) and broadcast interpretation (Cooper and Bowden, 2009) were available which are either weak (subtitles) or very few to build models which would work on a large domain of discourse.

The CSLR methods (Koller et al., 2017, 2016) (designed to learn from weakly annotated data) were infeasible, as researchers assumed that sign videos and their annotations share the same temporal order. With the creation of SL datasets such as `RWTH-PHOENIX-Weather 2012` (Forster et al., 2012), `RWTH-PHOENIX-Weather 2014` (Forster et al., 2014), or `KETI` (Ko et al., 2019) made it possible for the researchers to directly work on video frames and invent models to interpret signs/meanings residing in them.

SLR models utilized convolutional modules to encode the video frames and recurrent mechanisms to capture temporal structures and dependencies in between frames (Koller et al., 2017; Camgoz et al., 2017). SLT models also benefited from similar technologies for translating information into actual sentences (Gehring et al., 2017; Glorot and Bengio, 2010). Researchers customized this pipeline based on their own needs, e.g. Ko et al. (2019) augmented network inputs with keypoints extracted from human faces, hands, and body parts. Graves et al. (2006) proposed the connectionist temporal classification (CTC) loss which is useful when working with weakly annotated datasets. Due to its success, CTC quickly turned into a mainstream loss function in sequence-to-sequence applications. Camgoz et al. (2020) embedded the CTC loss into Transformers (Vaswani et al., 2017) to learn the continuous sign language recognition and translation.

3 Methodology

Current state-of-the-art for SLT (Camgoz et al., 2020) relies on a Transformer-based architecture [1] in which the encoder is fed with sign video frames and the decoder produces translations conditioned on encoder's representations. In this framework,

[1] We assume that the reader is familiar with Transformers so we skip related details.

the encoder is trained to act as a gloss generator and this makes it possible to perform SLR and SLT simultaneously. Our model also follows a similar process but via a different and better architecture.

While our best performing model implements the same encoding process as in Camgoz et al. (2020), our decoder is equipped with a multitasking strategy where SLT is decomposed into two tasks of *i)* sign-to-spoken language conversion where source (German in our case) signs are converted to the source tokens. Then we have *ii)* gloss sequence prediction that provides additional annotations to facilitate the SLT process. In case of the unavailability of gloss annotations a complementary second task is proposed, where we translate source signs into a target language. Figure 1 illustrates the high-level design of our architecture.

As the figure shows, the decoder has three channels, namely D_{ts}, D_g and D_{tr} for transcribing input frames and generating gloss tokens and translation, respectively. All these channels share parameters of their first n blocks with each other. This feature helps us control the memory footprint of our model. Moreover, exchanging information in between channels yields richer internal representations. In addition to those n blocks, each of D_g and D_{tr} has one additional block whose parameters are *not* shared. Therefore, both D_g and D_{tr} have $n+1$ and D_{ts} has n blocks. Dedicated blocks are designed to reach better performance and mitigate the complexity of multitasking. *It is to be noted that the best performing architecture does not train D_g and D_{tr} simultaneously. Also, we only train D_{tr} to facilitate our complementary translation task, when we can not train D_g due to the unavailability of gloss sequences.*
The following sections describe the encoding and decoding process of our proposed model.

3.1 Encoding Sign Videos

The encoder takes a sign-video V as its input. We segment V into frames [f_1, f_2,...,f_F], then each frame is spatially embedded using a particular Inception network (Szegedy et al., 2016) which is pre-trained and fine-tuned convolutional model for the SLR purposes (Koller et al., 2019). Intermediate embeddings generated by the convolutional module are then passed through *batch normalization* and *rectified linear units* (Nair and Hinton, 2010) in order to enrich internal representations. Impact of these units and how they boost the test-time performance are comprehensively discussed in Camgoz et al. (2020).

Transformers are non-recurrent networks, so in order to maintain the temporal order of frames we augment embeddings with position information, as shown in Equation 1:

$$\begin{aligned} I_t &= \text{CNN}(f_t) \\ \hat{I}_t &= I_t + \text{PosEmb}(t) \end{aligned} \tag{1}$$

where CNN(.) refers to the convolutional model and PosEmb(t) is the embedding correlates with the *t*-th time step. This process is identical to positional encoding proposed by Vaswani et al. (2017). $\hat{I}_t$ is an intermediate representation that consists of intra-frame spacial and inter-frame positional information. Each processed frame $\hat{I}_t$ is passed through multiple encoder blocks and is transformed into an output vector z_t, as shown in Equation 2:

$$z_t = \text{Encoder}(\hat{I}_t) \tag{2}$$

3.1.1 Enriching Encoder Representations

Our Encoder serves a strong, multi-channel decoder so it is supposed to provide as rich information as possible. In our experiments we realized that only encoding sign videos is not sufficient enough and we need a more explicit way of teaching the encoder about its role and form of representations it should deliver. To this end we tried to inject gloss-level information by forcing the encoder to generate gloss labels in addition to its main task. In other words, we treat the encoder as a sequence labeler to solve the $P(G|V)$ problem, with G being a sequence of glosses. The encoder consumes video frames and it generates which glosses are related to those frames. This is an ordinary sequence-to-sequence problem which can be solved via an ordinary loss function such as cross-entropy. However, framing the problem that way requires an accurately-labeled dataset, which is not practical in our setting. Instead, we used the CTC loss which provides weaker supervision but satisfies our needs.

The log-likelihood of a gloss sequence given the input frames can be computed as shown in Equation 3:

$$\log p_\theta(G|V) = \log \sum_{a \in \beta^{-1}(G)} p_\theta(a|V) \tag{3}$$

where θ is a set of all encoder parameters and $\beta(G)$ returns all the possible alignments. For more details

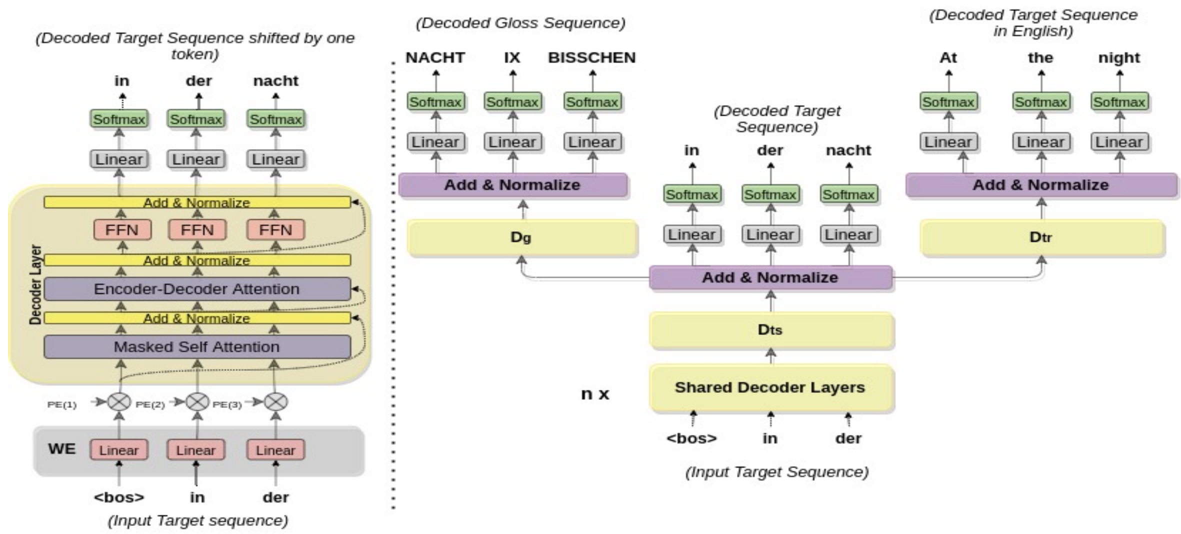

Figure 1: **Left:** The architecture of an ordinary transformer decoder. **Right:** The architecture of the proposed model that relies on a triple-channel decoder. D_{tr} and D_g denote two dedicated decoder blocks for translating input sequences into the target language (English) and gloss sequences, respectively. Aside from these two channels there is a third one, namely D_{ts}, which transcribes the input and generates real German words. The backbone of D_{tr} and D_g channels are shared and they only differ in the last block, i.e. the first **n** blocks but the last dedicated ones are shared in between channels. Therefore, each of D_{tr} and D_g have n shared and 1 dedicated blocks. D_{ts} has only n blocks with no additional, dedicated block and all its n blocks share parameters with other channels.

about the fundamentals of CTC and gloss-frame alignments, see Graves et al. (2006) and Camgoz et al. (2020), respectively. Computing $p_\theta(G|V)$ is intractable, and so the summation in the equation can be simplified as in Equation 4:

$$p_\theta(a|V) = \prod_i p(a_i|V;\theta) \tag{4}$$

where frame-level gloss probabilities are directly obtained from the encoder which is connected to a *Softmax* function through a projection layer in our architecture.

3.2 Multi-Channel Decoding

Our decoder is essentially a Transformer-based sequence generator and follows the same structure as other ordinary decoders (Vaswani et al., 2017). Therefore, it is a stack of Transformer blocks with all the positional encoding, masking, self-attention, encoder-decoder attention, position-wise feed-forward, and layer-norm components. We are also faithful to the original configuration of these components.

Although the main skeleton of our decoder relies on Transformers, ours has multiple output channels instead of one. The first channel D_{ts} transforms the video information to source-side words and can be used as a transcriber. Essentially, D_{ts} is used to generate German sentences corresponding to source sign videos. Finally, the second channel denoted by D_g decodes the gloss sequences. These channels exchange information among each other through shared parameters and this helps the decoder be aware of the target language, source language, and auxiliary annotations about the input frames at the same time, and we show empirically this is the main origin of our model's superiority. A natural question arises if the gloss sequences are unavailable, our proposed model is essentially a transformer architecture which cannot exploit gloss sequences both on encoder and decoder sides. In that case, the second channel of the decoder, D_g is useless. To alleviate this issue, we use a separate channel D_{tr} in place of D_g which is to be used for generation of target tokens corresponding to the input video frames in another language other than the language in which D_{ts} is trained on. (for our dataset, we generate sentences in English via D_{tr}, which are machine translated from the available German sentences).

We follow the structure as shown in Figure 1 to implement our decoder. Each channel of the decoder is trained by computing the cross-entropy loss of its generated tokens, as shown in Equation

5:

$$\mathcal{L}_{CH} = 1 - \prod_{t=1}^{T} \sum_{l=1}^{L_{CH}} p(\hat{w}_t^l) p(w_t^l | s_t) \qquad (5)$$
$$CH = \{D_{tr}, D_{ts}, D_g\}$$

where w_t^l denotes the probability distribution of the l-th target token at time step t whose ground-truth label is provided by $\hat{w}_t^l$. Each channel generates a different token, e.g. w is a target-language token for D_{tr}, whereas D_g works with glosses. L_{CH} shows the length of the vocabulary side that each channel works with. s_t is the internal state of the decoder which is computed as shown in Equation 6:

$$s_t = \text{Decoder}(w_{t-1} | w_{1:t-1}, z_{1:F}) \qquad (6)$$

As the equation shows, generation of each token at a particular time step is conditioned on all the previously generated target words ($w_{1:t-1}$) as well as encoder's outputs ($z_{1:F}$) for the input video segment.

According to Equation 5, each channel has a dedicated loss. We also define an auxiliary loss for the encoder ($\mathcal{L}_{enc}$). Therefore, the final loss for training our model is a composition of four loss terms, as shown in Equation 7:

$$\mathcal{L} = \lambda_{tr}\mathcal{L}_{D_{tr}} + \lambda_{ts}\mathcal{L}_{D_{ts}} + \lambda_g\mathcal{L}_{D_g} + \lambda_{enc}\mathcal{L}_{enc} \qquad (7)$$

λ assigned to each loss is a weight to control the contribution of each loss to the translation process.

3.3 Motivation of modeling choice

Motivation for the individual decision are as follows:

- According to the prior state-of-the-art work Camgoz et al. (2020), exploiting gloss annotations via forcing the encoder of the model to generate gloss sequences serves the decoder to perform better in SLT. Based on this line of thought, we utilize gloss annotation by using a separate channel in the decoder (D_g) which decodes gloss sequence. Empirical results suggest that this choice improved the performance of the prior state of the art performance from 21.32 to 22.59 BLEU-4 score. Please refer to 1.

- Annotating gloss sequence is costly and laborious and our model is reduced to a baseline transformer architecture in the unavailability of gloss sequence. To counter this problem, we design a relatively easy (proxy) task which can boost the accuracy of baseline transformer. To this end, we propose a third channel of the decoder (D_{tr}) which decodes the sign videos into an additional language (here we choose English) rather than corresponding gloss sequence.

4 Experimental Study

4.1 Datasets

To train our models and in the interest of fair comparisons we selected the `RWTH-PHOENIX-Weather 2014 T` dataset[2] (Camgoz et al., 2018). It contains the sign language videos along with their gloss annotations and translations in German.

To train our proposed model in the unavailability of the gloss sequences, we extend their train set by translating German spoken language sentences into English. For translation, we make use of the NMT system developed as a *WMT-19* submission by Ng et al. (2019)[3]. We provide an example from our training set in Table 1.

Firstly, we normalize punctuation & tokenize our target side of the dataset. Following tokenization, we use Byte Pair Encoding Scheme (BPE) (Sennrich et al., 2016), as currently used by almost all state-of-the-art NMT systems, to pre-process the target side of our dataset. This solves the problem of out-of-vocabulary (OOV) words in the test set as BPE encodes unknown words as a sequence of sub-words.

4.2 Hyper-parameter optimization

We employ Grid Search based hyper-parameter optimization. A set of initial estimates of the following hyper-parameters are chosen:

$$\begin{aligned} batch\_size &\in \{16, 32, 64, 128\} \\ num\_attention\_heads &\in \{4, 8\} \\ \lambda_g &\in \{0.2, 0.5, 0.7, 1.0\} \\ \lambda_{enc} &\in \{1.0, 2.0, 5.0, 10.0\} \\ num\_enc, num\_dec &\in \{3, 4, 5, 6\} \end{aligned} \qquad (8)$$

For all the experiments, we set $\lambda_{ts} = 1.0$. With a specific set of hyper-parameter our model is set to run.

After the hyper-parameter search is complete, we

[2] Link : `RWTH-PHOENIX-Weather 2014 T`
[3] WMT19 Fairseq

Gloss	NORDWEST HEUTE NACHT TROCKEN BLEIBEN SUEDWEST KOENNEN REGEN ORT GEWITTER DAZU
Text	im nordwesten bleibt es heute nacht meist trocken sonst muss mit teilweise kräftigen schauern gerechnet werden örtlich mit blitz und donner
Signer	Signer08
Name	train/11August_2010_Wednesday_tagesschau-5
Sign Video	 . . . .
English Translation	In the northwest, it will remain mostly dry tonight, with some heavy showers expected with thunder and lightning

Table 1: An example from the `RWTH-PHOENIX-Weather 2014 T` dataset used for training.

Tasks	**DEV**				**TEST**			
Sign to Text w/o gloss supervision	BLEU-1	BLEU-2	BLEU-3	BLEU-4	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Sign2Text (Camgoz et al., 2020)	45.54	32.60	25.30	20.69	45.34	32.31	24.83	20.17
Our Sign2Text [1]	45.43	32.67	25.38	20.74	45.45	32.68	25.24	20.52
D_{SEP}	44.52	31.96	25.00	20.58	45.17	32.82	25.45	20.90
$D_{SEP}++$ [2]	**46.55**	**34.08**	**26.50**	**21.63**	**46.56**	**34.04**	**26.39**	**21.59**

Table 2: Comparison between baseline models. *For more about our baseline models, see section 4.3*

[1] We replicated their experiment but with employing label smoothing parameter of **0.2**

[2] Separate decoder network (D_{SEP}) with extra gloss level supervision on the encoder side (see section 3.1.1)

use **best hyper-parameter choices** ($batch\_size = 32$, $num\_enc = 3$, $num\_dec = 3$, $\lambda_{enc} = 5.0$, $\lambda_g = 0.7$ and $num\_attention\_heads = 8$) output by the hyper-parameter optimizer to train and test our models. Note that the best performing model setup assigns $\lambda_{tr} = 0$ in the availability of gloss sequence (ref. Table 3). Adam (Kingma and Ba, 2014) is used as our preferred optimizer to train the models with an initial learning rate of 10^{-3} (β1=0.9, β2=0.998) and a weight decay of 10^{-3}. We use *plateau learning rate scheduler* which tracks the development set performance. We evaluate our model on the development set after every 200 iterations of training steps and if the BLEU-4 score (Papineni et al. (2002)) does not increase for 15 evaluation steps, learning rate is reduced by a factor of 0.7 until it reaches 10^{-7}, after which the training is stopped. While testing our proposed model, we use beam search to decode the target tokens with beam-length varying from 1 to 10.

We tune hyper-parameters of the baselines as well as our proposed method to compare their performance on an equal footing. Without hyper parameter tuning, our method has a performance score of 22.4±0.2 BLEU-4 over a range of hyper parameter choices (with fixed no of encoder and decoder layers of 3, $\lambda_{enc} = 5.0$, $\lambda_{tr} = 0$, $\lambda_{ts} = 1.0$). Though we report only the best performance score of 22.59 BLEU-4 score, the lowest performance of 22.2 is still better than the state-of-the-art score proposed in Camgoz et al. (2020), which is 21.32 .

4.3 Baseline Models

We design two baseline models for our experiments. The design decision is based on the premise that we cannot use any gloss-level supervision while training the baseline models. This entails having a fair comparison with our proposed architecture which uses internal gloss-level annotations.

4.3.1 Ordinary Transformer Network

We train an ordinary transformer model by setting hyper-parameters associated with the joint loss term (see equation 7) λ_{enc}, λ_g, λ_{tr} to zero. It alleviates any gloss-level supervision and our triple-channel decoder works as a single decoder which directly decodes German spoken language sentences from the sign language videos. This model has the poorest performance of 20.52 BLEU-4 score.

4.3.2 Separate Decoder Network (D_{SEP})

Instead of our proposed model, which is equipped with multitasking by exploiting a shared decoder representation via D_g, baseline model D_{SEP} has 2 separate decoders which do not share any information with each other. In Figure 2, Dec_T and Dec_G refer to two separate decoders which use the same encoder representation to predict the target sequence and gloss sequence from the input sequence, respectively. Dec_T and Dec_G are respectively complementary to that of D_{ts} and D_g in our proposed model. As the decoders in this architecture (D_{SEP}) do not share any previous decoder layers as our proposed architecture does, this baseline model suffers from weak supervision

Tasks	DEV				TEST			
Sign to Gloss to Text	BLEU-1	BLEU-2	BLEU-3	BLEU-4	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Sign2Gloss2Text (Camgoz et al., 2020)	47.73	34.82	27.11	22.11	48.47	35.35	27.57	22.45
Sign2Gloss → Gloss2Text (Camgoz et al., 2020)	47.84	34.65	26.88	21.84	47.74	34.37	26.55	21.59
End-to-End Sign to (Gloss+Text)	BLEU-1	BLEU-2	BLEU-3	BLEU-4	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Recog. Sign2(Gloss+Text) (Camgoz et al., 2020)	46.56	34.03	26.83	22.12	47.20	34.46	26.75	21.80
Trans. Sign2(Gloss+Text) (Camgoz et al., 2020)	47.26	34.40	27.05	22.38	46.61	33.73	26.19	21.32
$T^{5.0}_{0.0,0.0}$	45.03	32.31	24.92	20.26	46.66	33.20	25.81	21.07
Our $T^{0.0}_{0.7,0.0}$	44.85	33.17	26.16	21.61	44.66	32.78	25.62	21.08
Our $T^{5.0}_{0.7,0.2}$	47.42	35.21	27.77	22.90	47.07	34.57	26.96	22.05
Our $T^{5.0}_{0.7,0.0}$	**48.01**	**35.46**	**27.94**	**23.05**	**47.59**	**35.16**	**27.60**	**22.59**

Table 3: Comparison between state-of-the-art and our model. Here, $T^{\lambda_{enc}}_{\lambda_g,\lambda_{tr}}$ denotes our proposed architecture. For different values of λ_{enc}, λ_g and λ_{tr}, we tabulate their effects on test BLEU-4 score. We show three of our best results and tabulate them accordingly. *For all experiments, we set* $\lambda_{ts} = 1.0$. $T^{5.0}_{0.0,0.0}$ refers to our re-implementation of state-of-the-art architecture(Camgoz et al., 2020) with the same training setting described in Camgoz et al. (2020)

of gloss annotations and thus perform somewhat poorly (BLEU-4 score of 20.90) compared to our proposed model with $\lambda_{enc} = 0$ (BLEU-4 score of 21.08).
When equipped with encoder side gloss sequence decoding, by setting $\lambda_{enc} = 5.0$, performance of D_{SEP} is increased to 21.59. We call this enhanced D_{SEP} as D_{SEP++}

4.4 Results & Comparisons

Sign to Text tasks with gloss supervision can be divided into two parts, namely **Sign2Gloss2Text** and **Sign2(Gloss+Text)**. We discuss these in briefly and compare with our proposed method.

4.4.1 Models with mid-level gloss supervision

Sign2Gloss2Text uses intermediate gloss level representation. It is a two-step process. The first step uses a CSLR (Continuous sign language recognition) model to generate the gloss sequences corresponding to a sign video. In the second step, the generated glosses are fed to train an NMT model which acts as a *Gloss2Text* translator, translating gloss sequences into a sequence of spoken language words. A variation of *Sign2Gloss2Text* is known as **Sign2Gloss → Gloss2Text**. This is similar to *Sign2Gloss2Text*, but instead uses best performing *Gloss2Text* network instead of training it from the scratch. For both of these architectures, we list the state-of-the-art scores in Table 3.

4.4.2 End-to-End models

The second category of tasks (*Sign2(Gloss+Text)*) essentially refers to learning both the gloss sequences and textual representations jointly, as done in Camgoz et al. (2020). Our model is an extension over the approach used in Camgoz et al. (2020). Table 3 shows that our model with best performing setup obtains a BLEU-4 score of 22.59, which is 0.79 absolute increase from the score of 21.80 obtained by Camgoz et al. (2020) for *Sign2(Gloss+Text)* tasks. The improvement was found to be statistically significant over the prior state-of-the-arts using bootstrap hypothesis testing [4] to test the Null Hypothesis (H_0) that the same system generated the two hypothesis translations, using the technique utilized in Camgoz et al. (2020) and our proposed method. At 95% confidence level, P-Value comes out to be 0.029. This entails that H_0 can be rejected, subsequently firming the claim that our method is better than the existing state-of-the-arts.

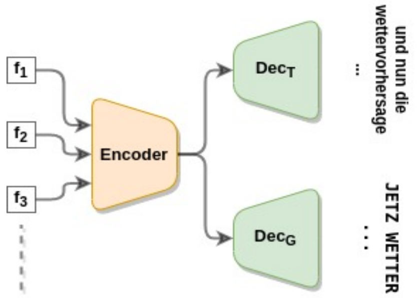

Figure 2: Architecture of our baseline model with one encoder and two separate decoders. Here, f_1, f_2,......,f_n are the spatial representation of the video frames obtained from the pre-trained CNN. Dec_T and Dec_G denote two separate decoders for decoding text and gloss sequence, respectively. For Sign2Text experiments, we drop Dec_G.

4.5 Ablation experiments

Performance of our proposed architecture depends on the choice of the weights (λ_{enc}, λ_{tr}, λ_g) associated with the loss term (7) used to train our model. We perform a ablation study to show the effect of

[4] Bootstrap Hypothesis Testing

hyper-parameter variations. Firstly, we consider our baseline models and consider how their performance changes if the gloss-level supervision at the encoder side (c.f 3.1.1) is added. Secondly, we consider our proposed model and compare it with their baseline counterparts.

Models	METRICS	
Models for Sign2(Gloss+Text)	BLEU-4	ROUGE
$T^{0.0}_{0.0,0.0}$	20.52	45.92
$T^{0.0}_{0.0,0.5}$	20.79	47.03
D_{SEP}	20.90	46.41
$T^{5.0}_{0.0,0.0}$ †	21.07	46.00
$T^{0.0}_{0.7,0.0}$	21.08	46.06
$D_{SEP}++$	21.59	47.69
$T^{5.0}_{0.7,0.2}$	22.05	48.25
$T^{5.0}_{0.7,0.0}$	**22.59**	**48.82**

Table 4: Comparison between proposed models with different loss weights. $T^{\lambda_{enc}}_{\lambda_g,\lambda_{tr}}$ denotes our proposed architecture. For different values of λ_{enc}, λ_g and λ_{tr}, we tabulate their effects on test BLEU-4 score.

† $T^{5.0}_{0.0,0.0}$ is the re-implementation of the architecture from Camgoz et al. (2020) using their choice of hyper-parameters.

We can conclude the following based on the Table 4. The model without any gloss-level supervision ($T^{0.0}_{0.0,0.0}$) has the lowest BLEU-4 score of 20.52. Gloss-level supervision using separate decoder network (D_{SEP}) boosts the baseline accuracy from 20.52 to 20.90. Training $D_{SEP}++$ which uses the architecture from D_{SEP} along with an added objective of enriching encoder representation (refer to Section 3.1.1) could subsequently increase the performance of D_{SEP} from 20.90 to 21.59. Following this increasing trend of performance we hypothesize that adding gloss level supervision, both at the encoder and decoder side, is the most useful multitasking approach to follow.

We follow the previous experiments using our proposed model. Our baseline D_{SEP} uses extra supervision from gloss sequences employing two separate decoders, implementing a soft parameter sharing paradigm for multi-tasking. For fair comparison with D_{SEP}, we run our proposed model which implements a hard parameter sharing paradigm of multi-tasking ($T^{0.0}_{0.7,0.0}$). This uses a shared backbone of n layers of decoder and 2 task-specific decoder layers. It boosts up the performance of D_{SEP} from 20.90 to 21.08, subsequently showing that *using representation from shared layers could boost multitasking performance when compared to separately obtained representations.* $T^{5.0}_{0.7,0.0}$ denotes our proposed model with an added objective of training the encoder with an auxiliary loss $\mathcal{L}_{enc}$, thereby setting $\lambda_{enc} = 5.0$. It gives a huge boost in terms of BLEU-4 score. This achieves the new state-of-the-art score of 22.59, with an impressive ROUGE score of 48.82.

Note that our re-implementation of the state-of-the-art (Camgoz et al., 2020) ($T^{5.0}_{0.0,0.0}$) and our proposed model with decoder only multi-tasking ($T^{0.0}_{0.7,0.0}$) have the similar performance, thereby firming our belief that exploiting gloss sequence in the target side is as useful as it is for the source side. Though our dual channel decoder has a dedicated channel (D_{tr}) for German to English translation, training it with D_g and D_{ts} harms the overall performance (by setting $\lambda_{tr} = 0.2$)[5]. When gloss annotations are unavailable, we can use German to English translation as a proxy task to improve the baseline performance. It is facilitated by only training two channels, D_{tr} and D_{ts}. $T^{0.0}_{0.0,0.5}$ surpasses the performance of the baseline *Sign2Text* model slightly (from 20.52 to 20.79 absolute improvement in BLEU-4 score).

We hypothesize that the marginal improvement is due to the fact that data used to train D_{tr} is obtained from an NMT model and performance could be improved more if we obtain gold standard human translation.

5 Conclusion

In this paper, we have proposed a transformer based novel architecture to perform the task of CSLR and SLT in an end-to-end fashion. Findings of this research can be summarized below:

- Exploiting intermediate sequences in an end-to-end fashion (e.g. gloss sequences) can be an effective approach to train the SLT models.
- If the gloss sequences are available, we can use some other task as a proxy for improving the performance of baseline model and we hypothesize that the task design is important.

As our approach is both model and task agnostic, extending our approach to other language understanding (NLU) tasks using various deep learning architectures is a promising research direction and in future we would like to explore that direction.

[5]For comparison, see BLEU-4 score of $T^{5.0}_{0.7,0.2}$ and $T^{5.0}_{0.7,0.0}$ in Table 4

References

Jan Bungeroth and Hermann Ney. 2004. Statistical sign language translation. In *Workshop on representation and processing of sign languages, LREC*, volume 4, pages 105–108. Citeseer.

Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, and Richard Bowden. 2017. Subunets: End-to-end hand shape and continuous sign language recognition. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3075–3084. IEEE.

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

Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. 2020. 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.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann.

Xiujuan Chai, Guang Li, Yushun Lin, Zhihao Xu, Yili Tang, Xilin Chen, and Ming Zhou. 2013. Sign language recognition and translation with kinect. In *IEEE Conf. on AFGR*, volume 655, page 4.

Helen Cooper and Richard Bowden. 2009. Learning signs from subtitles: A weakly supervised approach to sign language recognition. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2568–2574. IEEE.

Jens Forster, Christoph Schmidt, Thomas Hoyoux, Oscar Koller, Uwe Zelle, Justus H Piater, and Hermann Ney. 2012. Rwth-phoenix-weather: A large vocabulary sign language recognition and translation corpus. In *LREC*, volume 9, pages 3785–3789.

Jens Forster, Christoph Schmidt, Oscar Koller, Martin Bellgardt, and Hermann Ney. 2014. Extensions of the sign language recognition and translation corpus rwth-phoenix-weather. In *LREC*, pages 1911–1916.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Thomas Hanke, Lutz König, Sven Wagner, and Silke Matthes. 2010. Dgs corpus & dicta-sign: The hamburg studio setup. In *4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies (CSLT 2010), Valletta, Malta*, pages 106–110.

Shujjat Khan, Donald G Bailey, and Gourab Sen Gupta. 2014. Pause detection in continuous sign language. *International journal of computer applications in technology*, 50(1-2):75–83.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Sang-Ki Ko, Chang Jo Kim, Hyedong Jung, and Choongsang Cho. 2019. Neural sign language translation based on human keypoint estimation. *Applied Sciences*, 9(13):2683.

Oscar Koller, Cihan Camgoz, Hermann Ney, and Richard Bowden. 2019. 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*.

Oscar Koller, Hermann Ney, and Richard Bowden. 2016. Deep hand: How to train a cnn on 1 million hand images when your data is continuous and weakly labelled. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3793–3802.

Oscar Koller, Sepehr Zargaran, and Hermann Ney. 2017. 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*, pages 4297–4305.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814. Omnipress.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Pinar Santemiz, Oya Aran, Murat Saraclar, and Lale Akarun. 2009. Automatic sign segmentation from continuous signing via multiple sequence alignment. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 2001–2008. IEEE.

Adam Schembri, Jordan Fenlon, Ramas Rentelis, Sally Reynolds, and Kearsy Cormier. 2013. Building the british sign language corpus. *Language Documentation & Conservation*, 7:136–154.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. 2016. Inception-v4, inception-resnet and the impact of residual connections on learning.

Jan P van Hemert. 1991. Automatic segmentation of speech. *IEEE Transactions on Signal Processing*, 39(4):1008–1012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.