Sign Language Translation based on Transformers for the How2Sign Dataset

Patricia Cabot Álvarez
patricia.cabot@estudiantat.upc.edu

Xavier Giró Nieto
xavier.giro@upc.edu

Laia Tarrés Benet
laia.tarres@upc.edu

Abstract

The end goal of Sign Language Translation is to either produce spoken sentences from sign videos or generate sign videos from their corresponding written transcriptions. In this situation, this task has been address in multiple approaches in recent years. Moreover, it has been proved that taking advantage of the sign gloss representations improves substantially the model's performance in this task.

Therefore, in this work we replicate the state-of-the-art Transformer-based approach on the task and evaluate it on the multimodal American Sign Language How2Sign dataset. Furthermore, we provide baseline recognition and translation results that represent an starting point to further research on the topic.

In addition, we provide a new sentence-based alignment for the How2Sign videos, as their current alignment was with speech, which we have used to tackle the Sign Language Translation task properly.

1. Introduction

Sign Language is the basic communication channel for deaf people. It consists of manual articulations combined with non-manual elements, such as facial expressions, body poses or mouth motions [12]. Together, it results in a very complex natural language[1].

Contrary to common belief, each region developed and currently uses its own Sign Language, which have been listed to exist at least 150 different Sign Languages [6].

Furthermore, a Sign Language has its own linguistic singularities, such as grammar or semantics, as any other, and are different from the language that may be spoken in the region they belongs to. Hence, there is not a direct one-to-one mapping from the signs to its spoken translation, as it is more complex.

Additionally, there exist the gloss annotations. Glosses are the direct transcription of Sign Language, that differ from the spoken translation because they contain textual information about the sign and may include words which do not have an equivalent in the spoken language.

Unfortunately, communication for deaf communities with the rest of the society is very difficult, due to their lack of knowledge on Sign Languages. In this scenario, Automatic Machine Translation for Sign Language appears to be a practical solution and could have a significant beneficial impact. In fact, the task of Sign Language Translation (SLT) is divided into two goals: the first is to produce sign videos given the spoken language [13], whereas the second one is to generate written sentences from sign videos [2].

In the last years, researchers in the computer vision field have focused more on Sign Language Recognition (SLR) than in Sign Language Translation, a task that consists in generating sequences of glosses from sign videos. However, the recognition task may also be seen as an intermediate step towards Machine Translation, a supervised *assistance* that guides the training of translation models in a correct way [1]. In fact, it has been proved that the joint training of recognition and translation actually improves significantly the models' performance [2].

The main problem Sign Language Translation faces is the lack of available parallel corpus of sign videos and their aligned spoken transcriptions. Recently, some Sign Language datasets have been published, such as PHOENIX2014T [1] or How2Sign [5].

In this paper, we adapt the novel Transformer-based model introduced by Camgoz *et al*. [2], an approach that outperformed all the previous works in both tasks, recognition and translation, and provided state-of-the-art results.

The main contributions of this work can be summarized in two statements: (1) We adapt the state-of-the-art Sign Language Transformer model to How2Sign and provide baseline results that work as starting point for future research on this setup; (2) We generate new sentence-alignment for the sign videos from How2Sign, as the current available alignment was with speech.

Finally, this paper is organised in the following way: In Section 2, we summarise the work made by Camgoz *et al*., which is the foundation of this paper. In Section 3, we reproduce the results obtained by the authors with the same setup, by training the Sign Language Transformer model

[1] A natural language is any language that has evolved naturally in humans through use and repetition without conscious planning or premeditation.

on PHOENIX2014T. In Section 4, we narrate the adaption process of the model to train it with How2Sign. In Section 5, we provide the implementation and evaluation details, the experiments results on How2Sign and compare our best model to the reproduced results on PHOENIX2014T. Finally, in Section 7 we conclude our work by recapitulating the outcome, enumerating the limitations we faced and presenting future lines of work.

2. Related Work

Our research is essentially based on the work made by Camgoz *et al*. [2], as we have adapted it to the How2Sign dataset [5]. We address the machine translation task, and secondly recognition, of Sign Language, which has been researched for many years.

Sign Language Transformers. [2] In their work, the authors introduce a state-of-the-art approach for Sign Language recognition and translation in an end-to-end manner. They present a novel architecture based on Transformers [14], which is the current model used to address sequence-to-sequence tasks.

In Fig. 1 it can be observed an overview of the architecture of a single Transformer layer, which follows the encoder-decoder classic model. It is designed to generate the English sentence translations from the sign language videos, with intermediate gloss supervision.

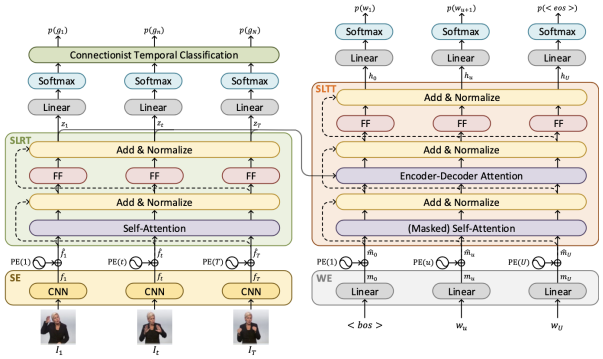

Figure 1. Overview of the architecture of a single Sign Language Transformer layer provided by the authors in their work [2]

On the left side of the image, one can find the encoder model, named SLRT. It correspond to the Sign Language Recognition Transformer. It receives as input the output of the Spatial Embedding layer, which extracts the sign video features from the raw frames. It is summed to the positional encoders[14], responsible of adding the temporal coherence of the frame sequence. The SLRT model follows the classic encoder architecture, as can be seen in the figure, with a Self-Attention module, with all operations followed by residual connections and a normalization step. Its goal is to recognize and predict the glosses corresponding to the input sign videos and, more importantly, learn meaningful spatio-temporal representations for the further sign language translation. In order to train the encoder, they used the Connectionist Temporal Classification (CTC) sequence-to-sequence learning loss function, instead of a cross-entropy loss, as they mention it would require much more precision in the gloss annotations, which is not commonly available.

On the right side of the image, one can find the decoder model, named SLTT. It corresponds to the Sing Language Translation Transformer. It receives as input the output of the Word Embedding layer, which computes a one-hot-vector representation of the English transcriptions into a dense space, and is summed to the positional encoders. The main goal is to generate English sentences from the sign video representations. The SLTT model follows the classic autoregressive decoder architecture, as can be seen in the figure, with a Masked Self-Attention module and an Encoder-Decoder Attention module, with all operations followed by residual connections and a normalization step. It is important to remark that in the Encoder-Decoder Attention module, the spatio-temporal representations learned from the SLRT are combined with the representations learned from the previous Masked Self-Attention module from the decoder and, there, it learns the mapping between the sign videos and the English transcriptions. In order to train the decoder, they implemented a cross-entropy loss for each word.

The network is trained by minimizing jointly the weighted sum of the recognition and the translation loss multiplied by two hyper-parameters.

They trained the model with the PHOENIX2014T [1], a dataset which contains parallel sign language videos, gloss annotations and their written translations, in the weather forecast domain from the German TV. We give further details on the dataset in the following section.

Moreover, they specify the following protocols performed to evaluate their model, stated in [1]:

- *Sign2Text*, which represents the end goal of translating from a sign video to its spoken transcription, without any intermediary representation.

- *Gloss2Text*, a text-to-text translation problem, which consists in translating from the sign glosses to the written sentences.

- *Sign2Gloss2Text*, which represents the current state-of-the-art in Sign Language Transformers. It mainly consists of recognizing the glosses from the sing videos and then using them to predict the written transcriptions.

Additionally, the authors introduced two new evaluation protocols:

- *Sign2Gloss*, which basically consists in sign language recognition from sign videos to glosses.

- *Sign2(Gloss+Text)*, the main contribution of their work, as it represents the joint learning of sign language recognition of glosses and translation to written sentences.

In conclusion, the work presented by Camgoz *et al*. introduces a novel architecture based on Transformers for learning jointly sign language recognition and translation and provides state-of-the-art results for both tasks, outperforming previous models on the tasks.

3. Reproducibility of SLT

For the purpose of replicating the work with the How2Sign dataset, we first reproduced their results with their same setup.

3.1. PHOENIX2014T dataset

This dataset [1] gathers a collection of Sign Language recordings from the weather forecast airings from the German public TV-station, along with its written text and gloss transcriptions.

The dataset version provided by the authors has the text, gloss and sign video already aligned in a structured framework.

Sign features. The sign information released together with the SLT code does not contain the raw video frames, but the features extracted from a pretrained CNN [7]. This network was pretrained for a sign language recognition task with the PHOENIX2014T dataset, in a CNN+LSTM+HMM configuration. Hence, when we replicated the work with the How2Sign dataset, we also had to extract the features from the sign videos beforehand.

3.2. Code modifications

In order to reproduce the experiment, we had to carry out the following:

First, we prepared the environment for the task. As they did not indicate the exact python version they were using, we struggled configuring the versions of the required packages. Therefore, some of the final package versions differed from the specified in the requirements.

Second, we modified the data path in the configuration file. Additionally, we had another problem, as the available version of the dataset in our server corresponded to an older one.

3.3. Results

After configuring the setup, we trained the SLT model, which took one hour approximately. The obtained results, showed in Tab. 1, are similar to the ones provided by the authors [2].

4. Adaptation to How2Sign

Once we managed to reproduce the author's results, we started adjusting it to our setup.

4.1. How2Sign dataset

How2Sign [5] represents one of the most complete datasets in the Sign Language field. It is a multimodal and multiview American Sign Language (ASL) dataset, which consists of more than 80 hours of parallel corpora of sign videos and their respective speech, English transcripts and depth. Additionally, it provides with three hours of 3D pose estimation.

Nevertheless, the corresponding gloss annotations are not available and, hence, we worked without them. In their place, we used the English text. That is to say, in the recognition task, instead of predicting the gloss annotations, we tried to predict the English transcriptions. This means that the recognition and translation tasks are equivalent in this setup. We have decided to fill the gloss part of the dataset with the English sentence because we considered that it was a better temporary solution than filling it with random or arbitrary words, as this way the model could still learn some meaningful spatio-temporal representations.

4.1.1 Datasets comparison

Apart from the language and other elements, here exists three remarkable differences between the two datasets.

On the first hand, as stated previously, the PHOENIX2014T dataset contains the gloss annotation aligned with the video frames, while they are not available for the How2Sign dataset and we replaced them with the English sentences.

On the second hand, the volume of data available in How2Sign cannot be compared to PHOENIX2014T. The total number of entries in the first is 33116, whereas in the second is 8256. That is to say, there are four times more entries in How2Sign.

On the third hand, How2Sign includes data from a wide range of domains, while the PHOENIX2014T only includes entries from the weather forecast domain. For instance, the number of unique words in the first is 24703, whereas in the second is 2889. Hence, this difference will impact the model's performance, due to the large gap between both frameworks.

In conclusion, for both the volume and the domain of data, we expected to obtain more general and realistic results for the task than the ones obtained with PHOENIX2014T, even though the sign glosses were not available for our setup.

	DEV					TEST				
Sign2(Gloss+Text) Task	WER	BLEU-1	BLEU-2	BLEU-3	BLEU-4	WER	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Authors' best results on Recog.	24.61	46.56	34.03	26.83	22.12	24.49	47.20	34.46	26.75	21.80
Authors' best results on Trans.	24.98	47.26	34.40	27.05	22.38	26.16	46.61	33.73	26.19	21.32
Our results (Recog. + Trans.)	48.92	44.30	31.47	24.35	19.88	48.06	44.62	31.80	24.40	19.79

Table 1. Comparison between the authors' results and our results obtained with the same setup.

4.2. Re-alignment

As explained previously, How2Sign contains the speech and English transcripts corresponding to the sign videos. Then, the current generated alignment of the video frames was with the speech and not with the text, as there was a delay between the two modalities. That is to say, there was an incorrect alignment of the videos and the sentences. Therefore, we first solved this problem and re-aligned the video frames with the English transcripts. In Fig. 2, there is a visual example of both alignments, given a sign video.

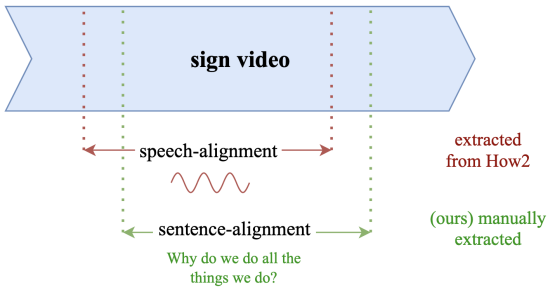

Figure 2. Example of the difference between the alignment based on speech or on sentence.

In their work [4], Duarte *et al*. highlight the importance of this change of framework, from speech-alignment to sentence-alignment, using the new re-alignment we generated. Moreover, they provide a quantitative comparison and show the improvement in the models performance when working with sentence-alignment.

4.3. Dataset cleaning

As a consequence to the new alignment between the videos and the sentences, it resulted that some sentences had a starting and ending points out of range in the number of frames that conformed a video. This resulted in empty tensors, which cannot be used in our task. Thus, we chose to remove them from the final dataset that we trained our model on, even though some further processing will be needed to correct these inconsistencies.

4.4. I3D feature extraction

As mentioned in previous sections, the SLT model is not trained with raw sign videos, but with extracted features that function as Spatial Embeddings (SE). In Fig. 1, this corresponds to the first layer the video frames are feed to and its output is the input to the Encoder Transformer (SLRT).

Our first approach was to try to compute our sign features with the same configuration that Camgoz *et al*. used. However, the exact architecture (CNN+LSTM+HMM) [8] was not specified in their work and neither was the code published. Hence, we studied other approaches.

Then, we used the sign video Embeddings represented as an I3D neural network architecture [3] from [5] to extract the features, which were trained in a Sign Language Video Retrieval task with How2Sign.

5. Results and comparison

Following, we show the results obtained from the different experiments with our setup. First, we give a description of the implementation and evaluation details. Then, we narrate the process we have followed to optimize the desired final model, illustrating it with the intermediate results from the different experiments. Finally, we compare our best results with the reproduced results with PHOENIX2014T, which have been already reported in Tab. 1.

Even so, it is important to remark that these results are provisional, as they were not obtained until the model converged, but after 24 hours, which was the time limit to run an execution in our servers.

5.1. Implementation and evaluation details

As we are using mostly the same configuration than the authors [2], most of the implementation details coincide with the ones specified in their work.

The model design corresponds to the work developed by Camgoz *et al*., Sign Language Tranformers[2], released in September 2020.

The architecture is based on Transformers[14] and unifies both the recognition and the translation tasks into a single model, which the authors showed that improved the performance. However, as in our setup the recognition task is performed with the English sentences, and not with the gloss annotations, we diminished its importance and focused mainly in the translation results.

The evaluation metrics also coincide with the ones used by the authors [2]. For the recognition task we use the Word

[2]https://github.com/neccam/slt

Error Rate (WER) [8] and for the translation task, we use the BLEU [10] scores (with n-grams from 1 to 4). Both are the most common metrics to measure the performance of speech recognition and automatic translation systems. Additionally, the environment is also configured to output the ROUGE [9] and CHRF [11] metrics for the translation task, although we considered to focus on the firstly mentioned in order to assess our models' performance.

5.1.1 Model adjustment

Although most of the model configuration remained untouched, in order to replicate the experiments with our setup, we had to modify some details.

More precisely, we changed the batch size and the validation frequency parameters. This is due to the fact that the volume of data for our experiment requires much more GPU memory and the time consumption in the validation step takes an amount of time that cannot be spent at the same number of iterations as with the authors' setup. Hence, we changed both the batch size to 16 and the validation frequency to 1000.

Furthermore, we also modified the validation step. After the training of a model finishes (either because the model converges or because the execution reaches its time limit, as in our case), there is an additional feature to evaluate at inference time. There, first it performs an iterative process to find the best beam search decoding parameter, with widths from 1 to 10, for both the recognition and the translation tasks on the validation dataset and, when it finishes, it performs the beam search decoding with the best parameters on the test dataset. However, as the recognition beam search size increases, the more time consumption it takes. Hence, in our case it did not finish the iterative decoding after 24 hours of execution. To address this difficulty, we decided to perform this iteration only over the translation beam search decoding and set the highest possible width value to the recognition beam search.

5.2. Model's optimization

Once we had a trainable model, we started researching on the optimal configuration, in order to validate whether the setup provided by the authors was the most suitable for our dataset or it required some modifications. Thus, we executed several experiments and evaluated their performance.

Given a baseline model, we modified two of its parameters: the number of Transformer layers and the number of heads. Its baseline values were 3 and 8, respectively. Therefore, by increasing or decreasing the model's size, our goal was to increment its potential without overfitting at the same time. That is to say, we searched for the best model's combination of parameters. It is important to emphasise that we mainly focused on the translation metrics (BLEU), as it is the task we are addressing in this work.

First, we explored the impact of the number of Transformer layers. It can be observed in Tab. 2 that the best performance is obtained with 2 layers. This result differs from the baseline, where the number of layers was 3. Nevertheless, we can notice in Fig. 3 how the Translation Loss for the model with 2 layers starts increasing after 5 epochs, along with other experiments, whereas the model with 3 layers maintains a decreasing tendency throughout all the epochs. Furthermore, in Fig. 4 it can be observed that the evolution during the training for the experiment with 3 layers shows the best development, as it reaches very similar scores to the experiment with 2 layers, but with 4 epochs less. Hence, even though the experiment with 3 layers did not obtain the highest results, we considered we should not disparage this option. For this reason, from this point the following experiments were all built with 2 and 3 Transformer layers, separately.

		DEV			TEST		
# Layers	# Epochs	WER	BLEU-1	BLEU-4	WER	BLEU-1	BLEU-4
1	8	99.35	15.37	1.65	99.52	15.31	1.55
2	**14**	**99.18**	15.92	**1.95**	**99.33**	15.61	**1.89**
3	10	99.40	**15.95**	1.75	99.55	15.33	1.76
4	12	99.32	15.70	1.73	99.40	15.17	1.82
5	5	99.42	10.59	1.11	99.61	10.59	1.08
6	7	99.40	15.32	1.31	99.60	**15.68**	1.26

Table 2. Effect of the number of Transformer layers.

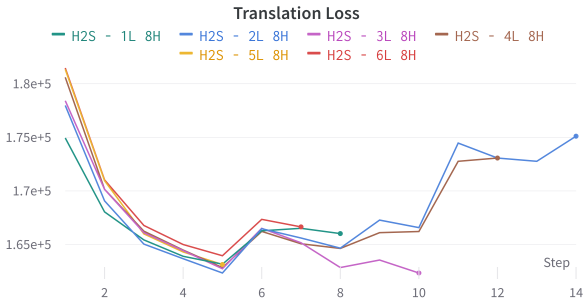

Figure 3. Translation Loss for validation at each epoch based on the number of Transformer layers.

On the one hand, we researched the impact of modifying the number of heads with 3 layers. In Tab. 3 it can be observed that the performance with 4 heads is outstanding compared to all the previous experiments. This result differs from the baseline, where the number of heads is 8. Moreover, the evolution of the BLEU-4 score during the training shown in Fig. 5 supports this assumption, as the growth of the model with 4 heads undoubtedly exceeds the other two. Thus, the optimal number of heads for 3 layers is 4.

On the other hand, we investigated the effect of the number of heads with 2 layers, as in the previous tests when

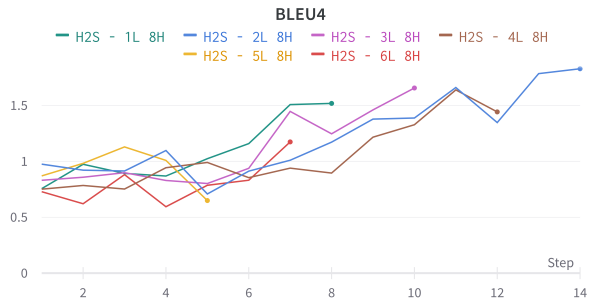

Figure 4. BLEU-4 metric for validation at each epoch based on the number of Transformer layers.

		DEV			TEST		
# Heads	# Epochs	WER	BLEU-1	BLEU-4	WER	BLEU-1	BLEU-4
4	**13**	**98.02**	**17.73**	**2.24**	**98.40**	**17.40**	**2.21**
8	10	99.40	15.95	1.75	99.55	15.33	1.76
16	13	99.42	13.71	1.63	99.63	13.87	1.44

Table 3. Effect of the number of heads with 3 layers.

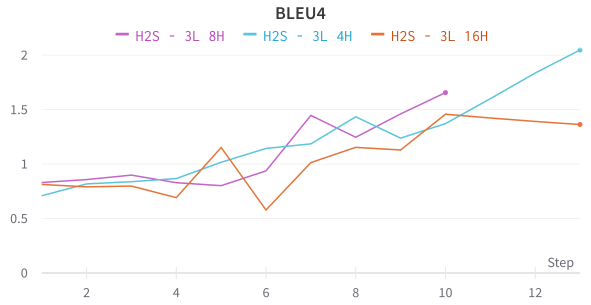

Figure 5. BLEU-4 metric for validation at each epoch based on the number of heads with 3 layers.

comparing the number of layers, this setup obtained the best scores. In Tab. 4, we can observe that the best performance corresponds again to the experiment with 8 heads. Therefore, the optimal number of heads for 2 layers is 8.

		DEV			TEST		
# Heads	# Epochs	WER	BLEU-1	BLEU-4	WER	BLEU-1	BLEU-4
4	12	99.35	14.80	1.76	99.48	14.75	1.79
8	**14**	**99.18**	**15.92**	**1.95**	**99.33**	**15.61**	**1.89**
16	8	99.60	12.04	1.50	99.69	11.75	1.51

Table 4. Effect of the number of heads with 2 layers.

Following, we compared the results obtained by the two experiments with best performance: the model with 3 layers and 4 heads and the model with 2 layers and 8 heads. First, when comparing their scores from Tab. 3 and Tab. 4, one can observe that the first configuration (3 layers and 4 heads) obtained considerably better results. For instance, there is a difference of 0.29 and 0.32 in the BLEU-4 metric on the validation and test, respectively. That is to say, the model with 3 layers and 4 heads is, without a doubt, the optimal. Furthermore, Fig. 6 shows that the tendency in the evolution of the BLEU-4 metric for this experiment is far more prominent than with the other experiment. And, additionally, in Fig. 7 it can be seen that the experiment with 2 layers and 8 heads has an increasing tendency in the Translation Loss, which could result in worse results in future training, whereas the Translation Loss for the model with 3 layers and 4 heads is nearly constantly decreasing.

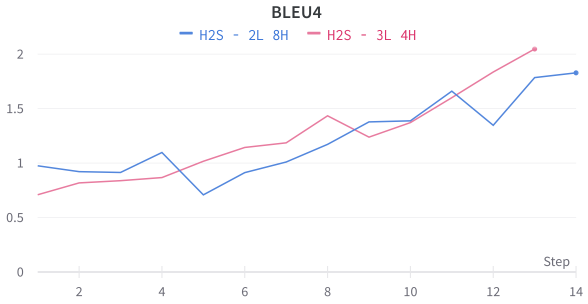

Figure 6. Comparison of the BLEU-4 metric for validation at each epoch for the experiment with 2 layers with 8 heads and the experiment with 3 layers and 4 heads.

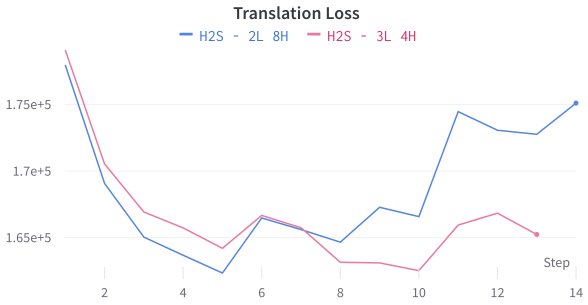

Figure 7. Comparison of the Translation Loss for validation at each epoch for the experiment with 2 layers with 8 heads and the experiment with 3 layers and 4 heads.

Finally, we compared our best results on the How2Sign datasets with the results we obtained when reproducing the authors' setup with PHOENIX2014T, already shown in Tab. 1. We display this comparison in Tab. 5. There, we observe that the results obtained with our setup (with How2Sign) are nothing alike the results obtained with PHOENIX2014T. There are some explanations we considered for this outcome:

First, the PHOENIX2014T dataset is restricted to a very specific domain, the weather forecast. Hence, it is easier for the model to guess the translation of a sign video, as the solution space is much smaller (the number of unique

	DEV					TEST				
[Recog. and Trans.] Tasks	WER	BLEU-1	BLEU-2	BLEU-3	BLEU-4	WER	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Reproduced results with PHX	48.92	44.30	31.47	24.35	19.88	48.06	44.62	31.80	24.40	19.79
Our bests results with H2S	98.02	17.73	7.94	4.13	2.24	98.40	17.40	7.69	3.97	2.21

Table 5. Comparison between our reproduced results with PHOENIX2014T and out best results with How2Sign.

words in How2Sign is approximetely 10 times larger than in PHOENIX2014T).

Second, non of the models trained with How2Sign converged after 24 hours. That is to say, they did not reach its optimal performance, whereas the model trained with PHOENIX2014T took fewer time to train (around one hour and a half) and it converged. This means that the difference between the scores should decrease if we retrained our models until they converged, which in fact is one of our future lines of work.

Additionally, in [4], Duarte *et al*. mention they modelled the Sign Laguage Transformer approach from [2] with the sign video Embeddings they trained for the Retrieval task on How2Sign, the same features we use in this work. They kept all the model parameters identical to the baseline and the BLEU results obtained coincide with ours: 1.74 and 17.08 for BLEU-4 and BLEU-1, respectively. Our results on the baseline were 1.76 and 15.33 for BLEU-4 and BLEU-1 on the test. Hence, this similarity on the performance gives solidity to our results.

6. Conclusions and Future Work

In this work, we described the process of adapting the Sign Language Transformer model to our setup. This work mainly addresses two fundamental tasks: recognition and translation of sign language videos, learned jointly.

We trained the model with the multimodal American Sign Language dataset, How2Sign. It represents a much more complete and extensive dataset in the Sign Language field than PHOENIX2014T, the one used by the original authors. Moreover, we took advantage of the I3D features computed with the How2Sign video frames for a Retrieval task and used them as Spatial Embeddings.

Therefore, with this work we provide a first view of the results one can obtain with this approach, which we plan to improve in the future.

Nevertheless, there are several limitations we have faced during the development of our research that should be addressed:

On the first hand, we already mentioned that the servers we work on have a time limit of 24 hours and, as a result, non of the models we trained actually converged. As the environment is already designed to train from the checkpoints, we plan to continue our research by retraining our current models until they converge. As a consequence, they should improve their performance considerably.

On the second hand, the annotation glosses are not available, hence the recognition task itself cannot in fact be fulfilled. This represents a major problem, due to the fact that the original authors show that the joint learning of the translation and recognition tasks improves substantially the performance.

On the third hand, in this work we have modified some parameters from the baseline model. However, apart from those we focused on, there are other parameters such as the several learning rates or the dropout value that could be modified to upgrade the model. Therefore, another step should be to modify other parameters.

References

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

[2] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[3] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *Proceedings of the Tenth Workshop on Statistical Machine Translation*, 2017.

[4] Amanda Duarte, Samuel Albanie, Xavier Giró i Nieto, and Gül Varol. Sign language video retrieval with free-form textual queries, 2021. arXiv:2201.02495.

[5] Amanda Duarte, Shruti Palaskar, Lucas Ventura, Deepti Ghadiyaram, Kenneth DeHaan, Florian Metze, Jordi Torres, and Xavier Giro i Nieto. How2sign: A large-scale multimodal dataset for continuous american sign language, 2020. https://how2sign.github.io/.

[6] David M. Eberhard, Gary F. Simons, and Charles D. Fennig. "sign language", ethnologue: Languages of the world, 2021. https://www.ethnologue.com/subgroups/sign-language.

[7] Oscar Koller, Necati Cihan Camgoz, Richard Bowden, and Hermann Ney. 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*, 2019.

[8] Oscar Koller, Jens Forster, and Hermann Ney. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *Computer Vision and Image Understanding (COMPUT VIS IMAGE UND)*, 2015.

[9] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL*, 2004.

[10] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2002.

[11] Maja Popovic. Chrf: character n-gram f-score for automatic mt evaluation. *Proceedings of the Tenth Workshop on Statistical Machine Translation*, 2015.

[12] Wendy Sandler and Israel Diane Lillo-Martin. Sign language and linguistic universals. *Cambridge: Cambridge University Press*, 2006.

[13] Stephanie Stoll, Necati Cihan Camgoz, Simon Hadfield, and Richard Bowden. Text2sign: Towards sign language production using neural machine translation and generative adversarial networks. *International Journal of Computer Vision (IJCV)*, 2020.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *31st Conference on Neural Information Processing Systems (NIPS 2017)*, 2017.