BACHELOR'S DEGREE THESIS

Degree in Data Science and Engineering

Title: Sign Language Translation with Pseudo-glosses

Author: Patricia Cabot Álvarez

Tutor: Xavier Giró Nieto

Advisor: Laia Tarrés Benet

Department: Signal Theory and Communications Department

Month and year: June, 2022

Facultat d'Informàtica de Barcelona
Facultat de Matemàtiques i Estadística
Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya
Facultat d'Informàtica de Barcelona
Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona
Facultat de Matemàtiques i Estadística

Degree in Data Science and Engineering

Bachelor's Degree Thesis

Sign Language Translation with Pseudo-glosses

Patricia Cabot Álvarez

Supervised by Xavier Giró Nieto and Laia Tarrés Benet
Signal Theory and Communications Department

June, 2022

Acknowledgements

I would like to thank the Image Group Department and, more precisely, our research group, which has been essential in the development of this project. Especially, I want to acknowledge the work carried out by Xavier Giró Nieto and Laia Tarrés Benet, who have advised and helped me carry out this project this year. Besides, I want to thank Gerard I. Gállego, Amanda Duarte, Maram A. Mohamed, Álvaro Francesc Budria Fernández, Cristina Puntí, Javier Sanz Fayos and Andrea Iturralde Amigó.

Finally, I want to thank my family and friends, who have been a big support for me during these months.

Abstract

Sign Language Translation is an open problem whose goal is to generate written sentences from sign videos. In recent years, many research works that have been developed in this field mainly addressed the Sign Language Recognition task, which consists in understanding the input signs and transcribing them into sequences of annotations. Moreover, current studies show that taking advantage of the latter task helps to learn meaningful representations and can be seen as an intermediate step towards the end goal of translation.

In this work, we present a method to generate automatic pseudo-glosses from written sentences, which can work as a replacement for real glosses. This addresses the issue of their collection, as they need to be manually annotated and it is extremely costly.

Furthermore, we introduce a new implementation built on Fairseq of the Transformer-model approach introduced by Camgoz *et al.*, which is jointly trained to solve the recognition and translation tasks. Besides, we provide new baseline results on both implementations: first, on the Phoenix dataset, we present results that outperform the ones provided by Camgoz *et al.* in their work, and, second, on the How2Sign dataset, we present the first results on the translation task. These results can work as a baseline for future research in the field.

Keywords

Sign Language, gloss annotations, sign videos, Sign Language Translation, Continous Sign Language Recognition, Machine Translation, Deep Learning, Transformer model, Encoder-Decoder model, Sign2Text, Sign2(Gloss+Text), Phoenix2014T, How2Sign, Fairseq.

Resum

La Traducció de la Llengua de Signes és un problema obert que té com a objectiu generar frases escrites a partir de vídeos de signes. En els darrers anys, molts treballs de recerca que s'han desenvolupat en aquest camp van abordar principalment la tasca de Reconeixement de la Llengua de Signes, que consisteix a comprendre els signes d'entrada i transcriure'ls en seqüències d'anotacions. A més, els estudis actuals mostren que aprofitar aquesta darrera tasca ajuda a aprendre representacions significatives i es pot veure com un pas intermig cap a l'objectiu final de traducció.

En aquest treball, presentem un mètode per generar pseudo-glosses automàtiques a partir de les frases escrites, que pot funcionar com a substitució de les glosses reals. Això aborda el problema de la seva adquisició, ja que s'han d'anotar manualment i és extremadament costós.

A més, introduïm una nova implementació basada en Fairseq de l'enfocament del model Transformer introduït per Camgoz *et al.*, que està entrenat conjuntament per resoldre les tasques de reconeixement i traducció. També proporcionem nous resultats de referència per ambdues implementacions: en primer lloc, per la base de dades Phoenix, presentem resultats que superen els proporcionats per Camgoz *et al.* en el seu treball i, en segon lloc, per la base de dades How2Sign, presentem els primers resultats de la tasca de traducció. Aquests resultats poden servir de base per a futures investigacions en el camp.

Paraules clau

Llengua de Signes, annotacions de glosses, videos de signes, Traducció de la Llengua de Signes, Reconeixement de la Llengua de Signes, Traducció Automàtica, Aprenentatge Profund, model Transformer, model Encoder-Decoder, Sign2Text, Sign2(Gloss+Text), Phoenix2014T, How2Sign, Fairseq.

Resumen

La Traducción de la Lengua de Signos es un problema abierto cuyo objetivo es generar oraciones escritas a partir de videos de signos. En los últimos años, muchos trabajos de investigación que se han desarrollado en este campo abordaron principalmente la tarea de Reconocimiento de Lengua de Signos, que consiste en comprender los signos de entrada y transcribirlos en secuencias de anotaciones. Además, los estudios actuales muestran que aprovechar esta última tarea ayuda a aprender representaciones significativas y puede verse como un paso intermedio hacia el objetivo final de la traducción.

En este trabajo presentamos un método para generar pseudo-glosas automáticas a partir de oraciones escritas, que puede funcionar como reemplazo de las glosas reales. Esto soluciona el problema de su recopilación, ya que deben anotarse manualmente y es extremadamente costoso.

Además, presentamos una nueva implementación basada en Fairseq del enfoque del modelo Transformer presentado por Camgoz *et al.*, que se entrena conjuntamente para resolver las tareas de reconocimiento y traducción. También proporcionamos nuevos resultados de referencia para ambas implementaciones: primero, para la base de datos Phoenix, presentamos resultados que superan los proporcionados por Camgoz *et al.* en su trabajo y, segundo, para la base de datos How2Sign, presentamos los primeros resultados en la tarea de traducción. Estos resultados pueden funcionar como base para futuras investigaciones en el campo.

Palabras clave

Lengua de Signos, annotacions de glosas, videos de signos, Traducción de la Lengua de Signos, Reconocimiento de Lengua de Signos, Traducción Automática, Aprenentage Profundo, modelo Transformer, model Encoder-Decoder, Sign2Text, Sign2(Gloss+Text), Phoenix2014T, How2Sign, Fairseq.

Contents

List of Figures

List of Tables

1. Introduction and related work

What is a Sign Language? Sign Language is the fundamental communication channel for deaf people. It is formed by a combination of hand gestures, known as signs, and non-hand elements, such as facial expressions, body poses or mouth movements [20].

There is not a universal Sign Language and they differ depending on the region. This happens because, as any other natural language, they evolved according to the human needs without a conscious planning or premeditation, and are constantly receiving influences from the society they belong to. According to some studies, at least 150 [7] distinct Sign Languages have been registered.

In terms of linguistics, Sign Languages are as rich and complex as any other spoken language [1], and each of them have their own singularities, such as grammar and semantics. Moreover, they are somehow independent from the spoken language from the region they belong to and, therefore, there exists not a one-to-one mapping between the signs and the spoken words.

One of the main issues deaf people still have to face is the communication barrier with the rest of the society, due to the general lack of knowledge in Sign Languages. Overall, this represents an important social problem. In this scenario, many research projects have addressed these issues through different approaches. More precisely, in this field there exists several tasks: Continuous Sign Language Recognition (CSLR), Sign Language Production (SLP) or Sign Language Translation (SLT). For example, the first one consists in understanding sign videos and has been the main focus for computer vision researchers [10], whereas the second consists in producing sign videos from written sentences in a spoken language [21]. Finally, the third is the opposite to the previous, it consists in generating written sequences from sign videos [3]. This last task is what this work tries to address.

Nevertheless, the main difficulty the research in this field faces is the lack of large multi-modal Sign Language parallel corpora. Recently, some datasets have been published, such as PHOENIX2014T [2] or How2Sign [6].

Hence, as aforementioned, this work focuses on the Sign Language Translation task, although it does not start from scratch, but it is the direct continuation of a previous work made by this team. In section 1.2 we provide a detailed description of that work.

The main contributions of this investigation can be summarized as:

1. A method to generate automatic pseudo-glosses to replace manual gloss annotations.

2. A novel implementation based on Fairseq of the Sign Language Transformer model which addresses the joint learning of Sign Language Recognition and Translation.

3. New baseline results of the Sign Language Translation task for two corpora, Phoenix and How2Sign, with several new experiments.

Finally, this work is organized as follows: in Section 2, we give a formal description of the problems and introduce the goals of this project. In Section 3, we present the proposed approaches to address each of the goals and the followed strategy plan. Then, in Sections 4 and 5, we describe the implementation details and provide results for the developed tasks. Finally, in 6, we conclude the research by summarizing the findings and the achieved goals, and presenting possible future work.

1.1 Related work

This research is based on the work introduced by Camgoz *et al.*, **Sign Language Transformers: Joint End-to-end Sign Language Recognition and Translation** [3]. The authors introduce a novel architecture based on Transformers [22] to jointly learn the Continuous Sign Language Recognition and Sign Language Translation in an end-to-end manner, which is the first work to address these issues with this approach. Furthermore, they provide state-of-the-art results on the PHOENIX2014T dataset for the two mentioned tasks that outperform all the previous results.

1.1.1 Sign Language Transformer architecture

In Figure 1 we can observe a general view of the architecture of one Transformer layer. It is formed by an Encoder and a Decoder. The goal of this model is to generate sequences of gloss annotations and sequences of a written sentences of the spoken language from the sign videos.

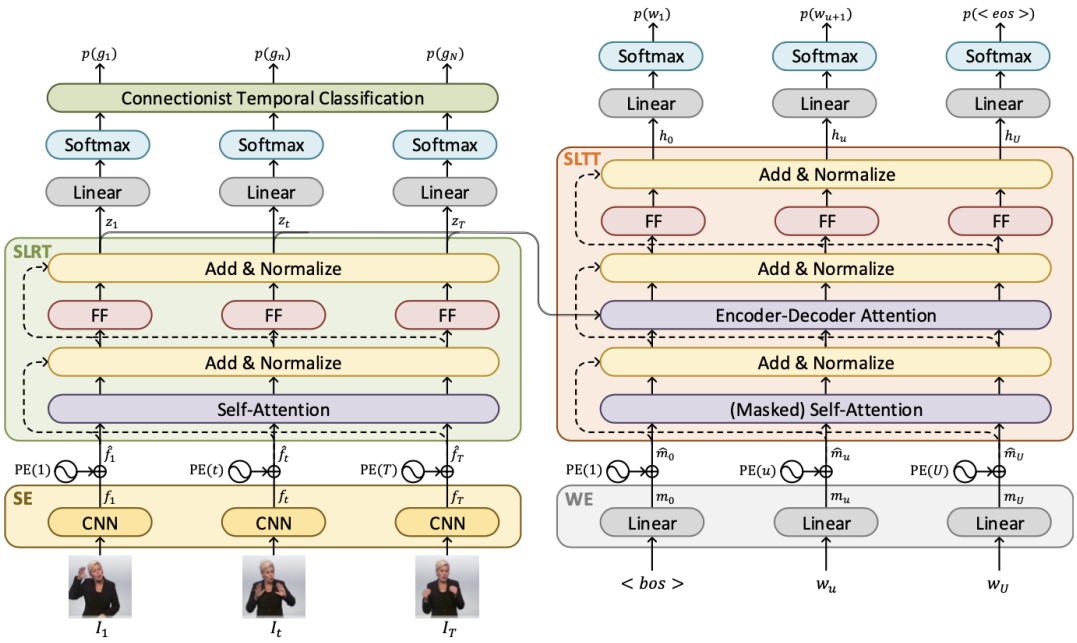


Figure 1: Overview of a single Sign Language Transformer layer provided by the authors in their work [3].

On the one hand, in the left part of Figure 1, there is the Encoder model: Sign Language Recognition Transformer (SLRT). Its goal is to learn meaningful spatio-temporal representations for the later Sign Language Translation from the sign videos and, as a minor task, to recognize and generate gloss annotations.

It receives as input the features extracted from the Spatial Embedding layer. That is to say, the first layer that receives the raw frames corresponds to a pretrained network that generates the embeddings. This network can differ depending on the dataset and the application, so we will further explain the approaches we have used in this work for each corpus. Then, the embeddings are summed to the Positional Encoding layer [22], which adds temporal information to the frame sequence.

Each SLRT layer is formed by a Self-Attention module and a non-linear point-wise feed-forward module. Besides, all operations are followed by residual connections and a normalization step. For the purpose of obtaining the gloss probabilities, the output of the final SLRT layer is fed to a Linear projection module followed by a Softmax activation.

Additionally, in order to train the encoder, they use a Connectionist Temporal Classification (CTC) [8] sequence-to-sequence learning loss function. They mention this is a better choice than using a cross-entropy loss, as it would require a precision the gloss annotations do not commonly have.

On the second hand, in the right part of Figure 1, there is the Decoder model: Sign Language Translation Transformer (SLTT). It aims to generate written sentences from the sign video representations, which is the end goal of this architecture.

It receives as input the output from the Word Embedding layer. It is formed by a linear layer projection which generates a one-hot-encoding vector representation of the words. This layer is initialized and trained from scratch along with the rest of the architecture. After, the embeddings are summed to the Positional Encoding layer, with the same approach as before.

Each SLTT layer follows the classical autoregressive decoder model. It is formed by a Masked Self-Attention module, an Encoder-Decoder Attention module, and a non-linear point-wise feed-forward module, with all operations followed by residual connections and a normalization step. More precisely, the Encoder-Decoder Attention module is responsible for learning the mapping between the spatio-temporal representations from the SLRT model and the representations from the previous SLTT layers. For the purpose of obtaining a sequence of word probabilities, the output of the final SLTT layer is fed to a Linear projection module followed by a Softmax activation.

Besides, in order to train the decoder, they use a cross-entropy loss for each word.

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

The implementation is based on the JoeyNMT toolkit [12]. It is a framework with educational purposes developed in PyTorch[1] to facilitate the implementation of Neural Machine Learning architectures[2]. In this work, the authors named their implementation *Signjoey*, as a combination of JoeyNMT and Sign Language.

1.1.2 Evaluation protocols

Apart from introducing the novel Transformer architecture, they specify the following evaluation protocols to evaluate the model, based on the tasks performed in the Sign Language field.

- *Sign2Text*, which represents the end goal of Sign Language Translation. It consists in generating written sentences from the sign videos without any intermediary representation.
- Gloss2Text, which consists in translating gloss[3] sequences into written sentences. In fact, it is a common translation problem from one language to another.
- *Sign2Gloss2Text*, which represents the current state-of-the-art in Sign Language Translation. Basically, it consists in using Continuous Sign Language Recognition (CSLR) models to extract

[1] PyTorch: an open-source Machine Learning framework: `https://pytorch.org/`
[2] GitHub of the JoeyNMT code: `https://github.com/joeynmt/joeynmt`
[3] Gloss annotations are the direct transcriptions of Sign Language. They are explained in more detail in Section 4

gloss sequences from the sign videos and then input them into a Gloss2Text setup to obtain written sentences.

- *Sign2Gloss*, which consists in generating gloss annotations from the sign videos, that is to say, performing a CSLR task.
- *Sign2(Gloss+Text)*, the main contribution of their work, as it represents the joint learning of Sign Language Recognition and Translation.

The two latter protocols are introduced by the authors in their work.

1.2 Previous work

As aforementioned, this work is the direct continuation of previous research carried out in the I2R-GCED subject during the Autumn 2021 semester. It mainly consisted in reproducing the state-of-the-art results on the Phoenix corpus introduced by Camgoz *et al.* [3] and then adapting the novel architecture to provide baseline results on the multi-modal How2Sign dataset on the translation task.

The main contributions from the previous work can be summarized in two:

1. A new sentence-based alignment for the sign videos from How2Sign, as they had a wrong arrangement of the timestamps.
2. Baseline results on translation for How2Sign that work as starting point for future research on this task.

Although the main goals of the project were achieved, in the work we specify several limitations we faced during its development. First, the servers we are working on have a time limit of 24 hours and, in consequence, non of the models trained on How2Sign converged. Second, the above-mentioned dataset does not currently have gloss annotations. Therefore, the recognition task performed with the Phoenix corpus could not be tackled correctly. Third, the architecture introduced by Camgoz *et al.* has many parameters and we only modified two of them, the number of heads and the number of Transformer layers. However, many others could have been modified in order to optimize the models' performance.

Hence, the first two described issues are addressed in this work. In Section 2, we provide a detailed description of the scope of the project.

2. Goals of the project

Following, we will describe the definitions and the problems related to the Sign Language field and their tasks. Besides, we will explain the general specifications of the project. Finally, we will introduce the goals of this work.

2.1 Sign Language tasks

As we all know, communication for deaf communities is difficult with the rest of the society, due to the general lack of knowledge of Sign Language. For this reason, the developing of effective Automatic Machine Sign Language Translation systems would result in important social benefits. That is why many research studies have tried to address this problem.

More precisely, the task of **Sign Language Translation (SLT)** is to produce written sentences from the spoken language from sign videos. Furthermore, there exists another task in the field which researchers in the computer vision field have been focusing the last years, **Sign Language Recognition (SLR)**. It consists in generating gloss annotations sequences from the sign videos.

Besides, in the work made by Camgoz *et al.* [3], they introduce a novel usage of the recognition task: they see it as an intermediate step towards translation. The idea behind is that the later task is very complex, because the input are videos, and the gloss annotations could help and assist the models to understand and learn the meaning of the signs, to further output sequences in the spoken language.

2.2 Sign Language gloss annotations

The gloss annotations is the name that receives a written Sign Language. They are direct transcriptions of the signs, which differ from the spoken language, as they contain textual information of the sign that is being represented. Besides, these kinds of annotations may include words that do not have an equivalent in the spoken language.

In general, glosses are important because they help to monitor the Sign Language and allow to include additional information such as facial expressions or body gestures, which are a very important part of the communication for the deaf communities.

Nevertheless, their collection is very difficult and costly, as they need to be manually annotated by linguistic experts. That is why it is hard to find a Sign Language corpus with aligned glosses.

2.3 Sign Language parallel corpora

As previously mentioned, the absence of available large parallel Sign Language corpora, with aligned sign videos and written transcriptions in the spoken language, is one of the main issues this research field faces. That is why, in this work, we will focus on working with two relatively new datasets, PHOENIX2014T [2] and How2Sign [6].

2.3.1 PHOENIX2014T

This dataset [2] gathers a collection of over three years (2009-2011) of Sign Language recordings from the weather forecast airings from the German public TV station, along with its aligned written text and gloss transcriptions. In Figure 2 we observe an example of a frame from the corpus.

Figure 2: Example of a frame from Phoenix.

This corpus is former by 8256 entries in total. The vocabulary size of the written text and the gloss annotations is 2889 and 1084, respectively. Besides, all the sentences belong to the weather forecast domain.

Sign features. As we are using this dataset for the translation task, we took advantage of the already extracted sign features [10] from a trained CNN, instead of using the raw frames. This network was pretrained for a sign language recognition task with the Phoenix corpus, in a CNN+LSTM+HMM configuration.

2.3.2 How2Sign

How2Sign [6] is one of the most complete datasets from 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 three hours of 3D pose estimation. In Figure 3 we observe an example of a frame from the corpus.

This corpus is formed by 33116 entries. The vocabulary size is 29408 words. Besides, it includes data from a wide range of domains, transforming this dataset into one of the best options to use in Sign Language tasks, because of its large amount of data and its potential applications.

Sign features. In the case of How2Sign, we decided to follow the same strategy as with Phoenix and use sign features that would function as the Spatial Embeddings (SE), instead of the raw frames. Therefore, we decided to use the sign video embeddings represented as an I3D neural network archi-

Figure 3: Example of a frame from How2Sign.

tecture [4] from [5] to extract the features, which were trained on a Sign Language Recognition task on How2Sign.

Compared to the previous architecture used to extract features, which used 2D Convolution layers at the frame level, these embeddings take into account the temporal dimensions, as the network is built with 3D Convolutional layers over clips of 16 frames. Therefore, this second network is richer in terms of motion information.

2.4 Goals of the project: formal definition

Once the Sign Language related issues have been described, we will transform them into project goals:

- As we commented on the difficulties of finding annotation glosses for these kinds of datasets, we will introduce a way to automatically generate pseudo-glosses.
- Furthermore, we will verify the utility of these pseudo-glosses by using them in a joint Sign Language Recognition and Translation task.
- We will build a novel implementation in the Fairseq toolkit based on the Sign Language Transformers model introduced by Camgoz *et al.* in their work [3], which learns in a joint manner the translation and recognition tasks.
- We will provide novel baseline results for Phoenix and How2Sign. First, with the SLT [3] model by designing new experiments, and second, with the novel implementation introduced in this work.

3. Proposed solution

In order to face the previously explained issues and goals, we decided to split them into two main blocks. The first one refers to solving the obtaining of glosses for a corpus. The second one faces the end goal of this project: to solve the Sign Language Translation task. We will give detail of the strategy plan we followed.

3.1 Automatic pseudo-glosses generation

On the one hand, as it has been exposed, the current method to acquire glosses is by manually annotating them, which requires a huge amount of time. To avoid it, we propose to use a deep learning model to automatically generate annotations that would work as pseudo-glosses.

More precisely, for obtaining these pseudo-glosses, we decided to use the Transformer-based model introduced by Yin *et al.* [24]. This work handles the Sign Language translation from gloss to text (Gloss2Text protocol), as the next step from a Sign Language Recognition (SLR) system which would extract glosses from videos. In our case, we propose to train the model backward, from text to gloss, to obtain automatic glosses from written sentences.

Apart from the code of the model, the authors provide the two datasets they trained their model on, explained in their work: the English dataset, ASLG-PC12 corpus, and the German dataset, PHOENIX-Weather 2014T dataset.

The ASLG-PC12 corpus is a dataset introduced by Othman *et al.* [15] as an answer to the lack of a large parallel corpus in the Sign Language field. Concretely, they proposed a rule-based approach for building a big parallel corpus of English written sentences and American Sign Language glosses. In fact, we took advantage of their work to tackle the same problem with a different methodology.

The PHOENIX-Weather 2014T [2] corresponds to the same dataset previously presented to train the recognition and the translation tasks on. However, the gloss annotations do not coincide for both corpora, which is a positive trait, as we will be generating new different glosses for the dataset and we will be able to compare the recognition results on both of them.

Therefore, we used both parallel corpora to train the model [24] and generate automatic pseudo-glosses for the How2Sign and the Phoenix datasets, respectively.

3.2 Sign Language Translation

On the other hand, the end goal of this work is to solve the Sign Language Translation task. In this direction, we followed two road maps.

First, we propose to continue the previous research and keep working with the architecture introduced by Camgoz *et al.*. That is, training the models on How2Sign until convergence. In fact, the environment is already prepared to train from the checkpoints and by doing this, we should be able to improve the performance obtained previously, which would mean obtaining new baseline translation and recognition results.

Second, and following the concept introduced by the authors, we propose to adapt their Sign Language Transformers work [3] and implement the same model in the Fairseq framework. According

to the definition presented on MetaAI's website[4], Fairseq is a "sequence modeling toolkit for training custom models for translation, summarization, and other text generation tasks". That is to say, this framework allows the easy design and training of sequence-to-sequence models by providing several implementations of reference models, such as Long Short-Term Memory (LSTM) networks and a novel Convolutional Neural network (CNN). Furthermore, they concretely mention that Fairseq is able to train models that achieve state-of-the-art performance on machine translation tasks, which is specially positive for this work's tasks due to the huge amount of time the models spend on processing all the video frames and generating the glosses (in the Recognition task).

[4]Fairseq by MetaAI: `https://ai.facebook.com/tools/fairseq/`

4. Automatic pseudo-glosses generation

As explained previously, the automatic glosses generation was performed by producing them with the Transformer-based model[5] introduced by Yin *et al.* [24]. The code is based on the open-source ecosystem OpenNMT[6], a framework designed in December 2016 by the Harvard NLP group[7] and SYSTRAN[8] for neural machine translation and sequence learning. In the GitHub of their work, along with the code and the data, they also provide a detailed explanation of the steps one has to carry out in order to install and use their code properly.

The first step was the processing of the train and validation data with a pre-process from the OpenNMT library, to prepare them for the following training. The second step was the training of the model. We used the default parameter values proposed in the documentation by the authors. Finally, the third step was the inference, in other words, the generation of the glosses. In the following subsections, we will detail the procedure followed for each dataset, How2Sign and Phoenix, to extract the pseudo-glosses and we will show some examples to illustrate the results. Besides, in this inference step, we have used the same default parameters defined in the documentation, being the most relevant the beam size value equal to 4.

It is important to highlight that, before performing the inference, the sentences passed to the model are required to be normalised. Concretely, the rules followed by the input text were:

- In lower case.
- With the punctuation signs separated from the words.
- Without contractions.

For example, instead of using the sentence "*Let's make coffee.*", the input text would be "*let us make coffee .*".

4.1 Pseudo-glosses for Phoenix

In the case of the Phoenix dataset, the German sentences were normalized, which means that they already satisfied the required rules. Therefore, we only had to gather them into a single file and input it into the trained model as the inference step.

Following, there are some examples of the generated pseudo-glosses, along with their corresponding sentences and the original annotated glosses, to compare them. In Table 1 we can observe that in general both glosses contain nearly the same *annotation words*, which confirms the usefulness of this method to extract glosses without having to annotated them. However, in Table 2 it can be seen that the pseudo-glosses do hardly coincide with the original annotations or with the sentence. These special cases, which are very few, happen because the sentences contain words that have not been seen or hardly seen during the training step. Hence, the model chooses the most probable word, even though it is not correct.

[5]GitHub of the work: `https://github.com/kayoyin/transformer-slt`
[6]OpenNMT official website: `https://opennmt.net/`
[7]Harvard NLP group: `https://nlp.seas.harvard.edu/`
[8]A company pioneer and global leader in machine translation: `https://www.systransoft.com/systran/`

Still, most of the generated glosses correspond to the first case, where their meaning coincides with the respective sentences and are similar to the original annotations.

raw sentence	und nun die wettervorhersage für morgen freitag den sechsten mai .
manual glosses	JETZT WETTER WIE-AUSSEHEN MORGEN FREITAG SECHSTE MAI ZEIGEN-BILDSCHIRM
pseudo-glosses	__ON__ JETZT WETTER WIE-AUSSEHEN MORGEN FREITAG SECHSTE MAI __OFF__

Table 1: Example of the pseudo-glosses generated for a German sentence from Phoenix.

raw sentence	im süden funkeln ohnehin häufig die sterne .
manual glosses	IX SONST REGION STERN SEHEN __EMP__ __EMP__ STERN SEHEN
pseudo-glosses	SUEDWEST SOWIESO STERN KOENNEN SEHEN

Table 2: Example of the pseudo-glosses generated for a German sentence from Phoenix.

4.2 Pseudo-glosses for How2Sign

Unlike the case of the Phoenix dataset, the raw How2Sign sentences not only did not fulfill any of the requirements mentioned above, but had also some spelling errors. For example, instead of having "*what's*", there was "*what?s*". That is why, before performing the glosses generation, we prepared the data for the task.

The process we implemented is the following. First, we replaced all the capital letters with lower case letters. Then, we used a *python* library called *spellchecker* in order to find and correct all the spelling mistakes. In third place, we added a space between all the punctuation signs and the words. Lastly, we replaced the contractions with their full words. In Table 3 we can observe some examples of the raw and the processed sentences.

raw sentence	We're going to work on a arm drill that will help you have graceful hand movements in front of you.
processed sentence	we are going to work on a arm drill that will help you have graceful hand movements in front of you .
raw sentence	Let's make sure that are feet are parallel and are knees are soft.
processed sentence	let us make sure that are feet are parallel and are knees are soft .

Table 3: Example of two sentences from How2Sign before and after the normalization process.

Then, we gathered the new sentences into a file and generated the pseudo-glosses. In Table 4 we can observe the generated annotations for a sentence. In general, the meaning is kept in both forms. Additionally, we can also extract some insights into the structure of these particular glosses. For example, all the pronouns are preceded by the suffix *X-*, while the adjectives are preceded by *DESC-*. Moreover, all the words are replaced by their root form, which means that the syntax of the annotations is far more simple than written English.

raw sentence	Let me demonstrate you this on my back because it's a lot easier.
processed sentence	let me demonstrate you this on my back because it is a lot easier
pseudo-glosses	LET X-I DEMONSTRATE X-YOU THIS ON X-MY DESC-BACK BECAUSE X-IT BE LOT DESC-EASIER .

Table 4: Example of the pseudo-glosses generated for an English sentence from How2Sign.

In Table 5 we can observe the pseudo-glosses for a given sentence where the generation is not as good as expected. We show it because there are some particular cases where the translation repeats

raw sentence	The aileron is the control surface in the wing that is controlled by lateral movement right and left of the stick.
processed sentence	the aileron is the control surface in the wing that is controlled by lateral movement right and left of the stick
pseudo-glosses	JUXTAPOSITION BE CONTROL SURFACE IN WING THAT BE CONTROL BY DESC-20ALSO MOVEMENT DESC-RIGHT AND LEAVE STICK STICK STICK STICK STICK MOVEMENT DESC-RIGHT AND LEAVE STICK AND LEFT STICK STICK STICK STICK STICK STICK STICK STICK STICK STICK BY AND LEAVE STICK STICK STICK BY AND LEAVE STICK FROM WING THAT .

Table 5: Example of the pseudo-glosses generated for an English sentence from How2Sign.

a word or an expression several times, for example, "*STICK STICK STICK*", or generates a word completely different from the original, for instance, converts "*aileron*" into "*JUXTAPOSITION*". This second case was explained previously, as it happened when a word appeared new at inference time. Again, most of the generated annotations are of relatively good quality and preserve the original meaning.

In conclusion, we have succeeded in implementing a method to generate automatically gloss annotations for two datasets in different languages, which done manually would have spend too much time. In order to assess their utility, and quality, we will use these results to train the SLT models, as replacements of the manual gloss annotations.

5. Sign Language Translation

In this section, we show the results obtained from the different experiments with both implementations. First, we give reasons for the usage of normalized sentences from How2Sign in the translation task instead of the raw ones. Then, we describe the implementation and evaluation details. There, we also itemize the important aspects of the Fairseq structure and describe the followed process to build our architecture in that framework. After that, we present the validation and test results obtained first on the architecture from Camgoz *et al.*'s work and then on the novel implementation introduced in this research. Finally, we evaluate the obtained results and compare both implementations based on their performance.

5.1 How2Sign pre-processing

As mentioned previously, the written sentences from How2Sign are not normalized. For instance, in Table 6, we can observe examples of the different variations of the word *what*, and even the last column contains a spelling mistake present in the corpus.

what	what's	what?	what,	what.	what'll	What	what?s

Table 6: Example of variations of the word *what* in the written sentences from How2Sign.

At this point, because we had to process the text with the purpose of obtaining the pseudo-glosses (explained in section 4.2), we considered that it would be beneficial to make use of these new sentences in the translation task. Not only it would reduce the vocabulary size (which means reducing the solution space), but would it unify the different versions of a single word into one and simplify their meaning, facilitating the translation for the model. Therefore, we hypothesize that this modification in the corpus would improve their performance.

	Raw text	Processed text
Number of unique words	29408	15055

Table 7: Number of unique words in the How2Sign sentences.

A piece of evidence that supports our theory is shown in Table 7, where we can observe that the normalization reduced by half the number of unique words in the written sentences.

5.2 Implementation and evaluation details

We have modeled the translation and recognition tasks with two framework implementations: the first one developed by Camgoz *et al.*, Sign Language Tranformers[9] which we will refer as Signjoey, released in September 2020, and second one, a novel implementation introduced in this work with the Fairseq toolkit. Even though each model is designed in a different framework, they both share the same general guidelines in order to jointly learn Sign Language Recognition and Translation.

[9]GitHub of the code: `https://github.com/neccam/slt`

As explained in Section 1, in their work, Camgoz *et al.* introduced several evaluation protocols, based on the architecture design and the corpus involved to solve the Sign Language Translation task. Hence, we focus on training models following the Sign2Text and Sign2(Gloss+Text) protocols.

We will evaluate our models on the previously mentioned datasets, Phoenix and How2Sign. For both corpora, we will train each of the Sign2(Gloss+Text) experiments on different versions of the gloss annotations, in order to study their impact on the models' performance. Hence, we will have the direct written sentences as glosses, the automatic pseudo-glosses generated in this work explained in Section 4 and the manual gloss annotations (only available for Phoenix).

In both cases, most of the implementation details coincide with the ones specified in the authors' work [3]. The Transformer model has a hidden size of 512. The encoder and decoder have 3 layers and 8 heads each, with a dropout value of 0.1. For the training, we use the Adam [9] optimizer and a learning rate of 10^3, with a plateau learning rate schedule, which reduces its value when the performance stops improving. During training and validation steps, the model conducts a beam search decoding for both glosses and sentences, yet the first one corresponds to a CTC Beam Search Decoding function, implemented in TensorFlow[10]. The beam sizes used during training are already established and, at inference time, the Signjoey model performs an iterative process to find the optimal values of the beam size for each decoding. Finally, in the case of the evaluation metrics, on the one hand, for the recognition we use the Word Error Rate (WER) [11], which is a common metric for assessing the speech-to-text accuracy of automatic speech recognition systems. On the other hand, for translation, we use the BLEU score [18] (with an n-gram of 4), which is one of the most common systems for evaluating the quality of machine translations. Additionally, the Signjoey model is designed to output the ROUGE [14] and CHRF [19] metrics for translation, however, we considered focusing only on the BLEU score in order to assess our models' performance.

5.2.1 Signjoey model specific implementation details

In the case of the already implemented Sign Language Transformers architecture based on JoeyNMT, we had to apply some modifications in order to replicate the experiments on How2Sign.

Essentially, we changed the batch size and the validation frequency parameters. This is due to the fact that the volume of data of our experiments on this dataset 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 Phoenix. Hence, we changed both the batch size from 32 to 16 and the validation frequency from 100 to 1000 and kept the original values for Phoenix.

5.2.2 Fairseq model specific implemenation details

In this case, we had to build our architecture for the tasks in the Fairseq toolkit [16]. First, we will present the framework and its structure, in order to obtain a wider view of it.

As mentioned, Fairseq is an environment prepared to implement sequence models and provides a robust organization as a way to facilitate its usage. For our case, these are the most important modules and aspects:

- **Task.** This directory contains the steps to load data, indicate the inputs to the model and its

[10]TensorFlow is a framework that facilitates the implementation and deployment of Machine Learning models: `https://www.tensorflow.org/`

targets, initialize the model and the criterion, specify the train and validation steps (generate task-specific metrics and compute the loss), etc. In general, all the necessary procedures to build a functional architecture. Each of these tasks is registered with a name.

- **Data.** This directory contains data-related scripts, that facilitate the methods to read, integrate, prepare and organize the data when training the model.
- **Models.** This directory contains implementations of multiple variations of Machine Learning sequence models, such as LSTM or Transformers. Each of these models is registered with a name.
- **Criterion.** This directory contains the implementations of several loss functions that can be used during training to evaluate the quality of the model by measuring the distance between the prediction and the ground truth values. Each of these criterion functions is registered with a name.
- **Sequence generator.** This module generates the translation of a given source sequence. That is to say, a model does not output a written text, but a sequence of probabilities for each token. Hence, this module is implemented to perform a beam search decoder and output the sentence from the model output with higher probabilities.
- **Examples.** This directory is formed by directories that contain the specifications of a task. More precisely, it contains the configuration files that define the implementation details. They can be from the path of the data to model parameters such as the number of layers or the optimizer type. All this information is stored in this file, although it can be modified in the command line to train the model by adding the name of the parameter after two dashes (`--`). Besides, it also contains the name of the task, the model, and the criterion.
- **Pre-process.** Inside the directory of a task in the Examples folder, there are also several scripts with the purpose of processing the data before training the model. Some of their tasks are removing incorrect instances (the sequence lengths are too short or too large, or even empty), creating new valuable information derived from the original data, or removing instances not present in any of the parallel corpora.

The Fairseq framework includes other multiple implemented modules necessary to build a complete model, aside from the previously explained. They facilitate the model construction and are transparent to the developer, who only needs to focus on the task-specific parts.

In our case, we are using a Transformer Encoder-Decoder. First, we duplicated the code from a Speech2Text task [23] [17] and started implementing our architecture from it. Following, we will detail the modifications we made in the source code. However, it is important first to remark that the adaptation of the code to the Sign2Text task was started and mainly developed by two members of the research team, Laia Tarrés and Gerard I. Gállego. Hence, what we added and changed were the parts specific to the Sign2(Gloss+Text) task.

- **Task.** We registered a new task named `sign_to_text_and_gloss` by copying and adapting the Speech2Text task script. The major changes were the introduction of two target dictionaries (written text and gloss annotations) and the preparation of two types of targets in the corpus. Besides, we implemented the validation step to compute the loss and to get the BLEU and WER metrics from the generated output sequences.

- **Data.** Here, we implemented a module named `SignFeatsDataset` and modified an existing one named `AddTargetDataset`. The first one managed the preparation of the input sign features, while the second one performed the same process for the target sequences of the model. In our case, we took the existing module in Fairseq and adapted it to our requirements, as we have two types of targets (written sentences and gloss annotations).

- **Model.** We registered a new model named `sign2text_and_gloss_transformer` by copying and adapting the model from the Speech2Text task, named `s2t_transformer`. We decided to select this model as it was also implementing a Transformer Encoder-Decoder. Then, we made some modifications. First, we replaced a convolutional layer with a linear at the beginning of the Encoder, that worked as a subsampler with the purpose of reducing the dimensionality of the input features, as happened in the speech case. Then, we added some modules in order to generate sequences of glosses (apart from the written sentences, as output from the decoder). First, we added a linear layer to work as a CTC projection, with output length the vocabulary size of the gloss annotations. Second, we created a function that took this output and computed the logarithmic softmax in order to generate a sequence of probabilities. The idea behind this implementation was taken from a commit in the Fairseq repository in GitHub where they added an auxiliary CTC loss for the Speech2Text task[11].

- **Criterion.** We registered a new criterion named `label_smoothed_cross_entropy_with_ctc`. First, it computes the label smoothed cross-entropy loss with the output sequences from the decoder, and then, it adds the CTC loss computed with the output of the linear+softmax modules from the encoder weighted with an input parameter. Again, the idea behind this implementation was taken from the same commit aforementioned.

- **Sequence generator.** We built a new generator named `SequenceGeneratorGloss`. It generates a sequence of predicted gloss annotations from the output of the linear+softmax modules from the encoder. The structure of this generator is similar to the original one, but here we replaced the beam search decoder by the CTC beam search decoder function from TensorFlow[12].

- **Examples.** In this directory, we created two configuration files, one for each corpus. There, we specified the paths of the data, the names of the task, model, criterion, and all the input arguments of the model, the training, the validation step, etc.

- **Pre-process.** We implemented two new pre-process files, one for each corpus. For both, the pre-process consists in deriving new columns with the start and end frame for each sentence in a video, generating two dictionary models one for each target (text and glosses), and removing those sentences with sign videos too long or too short (as a subtraction between the end and the start frame).

The generation of the dictionary models requires a special description. In this step, what it basically does is a glossary of predetermined vocabulary size where it gathers all the words (or subwords) from the text it receives as input. We are using the SentencePiece module[13]. It is described to

[11] CTC auxiliary loss commit: `https://github.com/facebookresearch/fairseq/commit/52658402c5f37ccc4c6b796c44c0115af3b6eca1`

[12] Documentation of the `tf.nn.ctc_beam_search_decoder` function: `https://www.tensorflow.org/api_docs/python/tf/nn/ctc_beam_search_decoder`

[13] GitHub to the SentencePiece module: `https://github.com/google/sentencepiece`

be an unsupervised text tokenizer and detokenizer mainly for Neural Network-based text generation systems where the vocabulary size is predetermined. It has several input arguments, and we want to focus on two of them: the first one is `vocab_size`, which refers to the dictionary size, and the second one is `model_type`, which refers to the kind of word tokenization algorithm will be used (or none at all, if we choose to use words). In this scenario, we decided to make experiments to decide which was the best configuration. Therefore, when we use the `model_type` as words, the `vocab_size` is the number of unique words in the text, while when we use `vocab_size` with a subword tokenizer (in this case, unigram[14] [13]), the `vocab_size` is half the previous number of unique words. In Camgoz *et al.*' work, they use the sentences at word level, so we will be able to compare both approaches.

Additionally, we kept the original value for the batch size (32). However, in the case of validation frequency, it is performed after finishing an epoch, not after a an arbitrary number of iterations during the training.

5.3 Results on Signjoey

Following, we will introduce the results obtained with the Signjoey architecture from the SLT work[3] with Phoenix and How2Sign. In the first case, we will deal with the reproducibility of the Sign Language Transformers work [3] and introduce novel experiments to the task. In the second one, we will begin the study by presenting the results obtained from the previous work of this team as a starting point.

5.3.1 Results on Phoenix

First, we reproduced the results provided by Camgoz *et al.* in their work. It can be observed in Table 8 that the performance of our model is very similar to theirs, mostly in the BLEU-4 score, which is the task metric we are focusing on.

	Validation		Test	
[Recog. and Transl.] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU4 ↑
Authors' best results on Recog.	24.61	22.12	24.49	21.80
Authors' best results on Trans.	24.98	22.38	26.16	21.32
Our best results	41.50	20.68	40.50	20.40

Table 8: Comparison between the authors' results and our results obtained with the same setup.

Nevertheless, our contribution is the introduction of new baseline translation results on Phoenix with variations from their setup. That is to say, as explained previously, apart from training the recognition task on the manual gloss annotations, we proposed to replace them with the written sentences and pseudo-glosses in order to assess the impact of having different versions of the glosses in the translation results, as well as not using glosses at all and not learning the recognition task.

In Table 9 we can observe the performance of the aforementioned experiments. As expected, none of the other models improve the results obtained with the manual glosses. Still, the BLEU-4 score is very similar in all the cases and it is only when we use the German sentences, that the performance decreases more significantly. In fact, the experiment with pseudo-glosses and the one

[14]The unigram segmentation is an algorithm introduced by Kudo based on the subword units probabilities.

that only performs translation achieve similar results. Moreover, this tendency can also be seen in Figure 4, which shows the evolution of the BLEU-4 during the training on the different setups. From there, we can extract that all the models behave similarly, standing out the experiment with manual glosses.

	Validation		Test	
[Recog. and Transl.] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU4 ↑
Only Translation task	-	19.77	-	19.52
German written sentences	85.96	18.23	85.18	18.17
Automatic pseudo-glosses	76.87	19.29	76.65	19.16
Manual gloss annotations	**41.50**	**20.68**	**40.50**	**20.40**

Table 9: Comparison of our results on Phoenix with Signjoey when modifying the gloss annotations.

However, it is important to remark that, even though the good translation results are achieved with all the models, the case of the recognition does not happen the same. In the aforementioned table, the gap in the WER results between the experiment with manual glosses and the other two is considerable. Besides, this difference can also be observed in Figure 5, where the evolution of the WER metric during the epochs is similar and far worse in the experiments that do not use the original annotations.

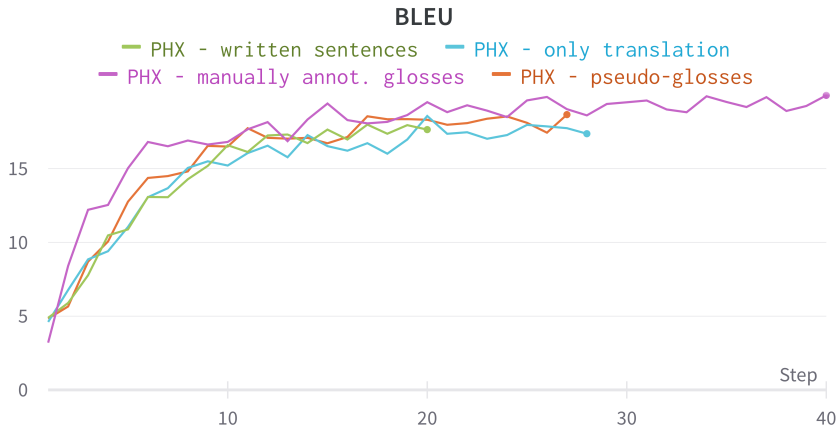


Figure 4: BLEU-4 metric for validation during training with Signjoey on the different Phoenix corpora based on the annotation glosses.

To sum up, the experiments' results on Phoenix show what we already knew: the highest performance is achieved with the setup introduced by Camgoz *et al.*. That is to say, it is the best layout in the case of this parallel corpus, where the manual glosses are available. Even so, we provide novel baseline results on the task that can lead to further research when there are no available annotation glosses.

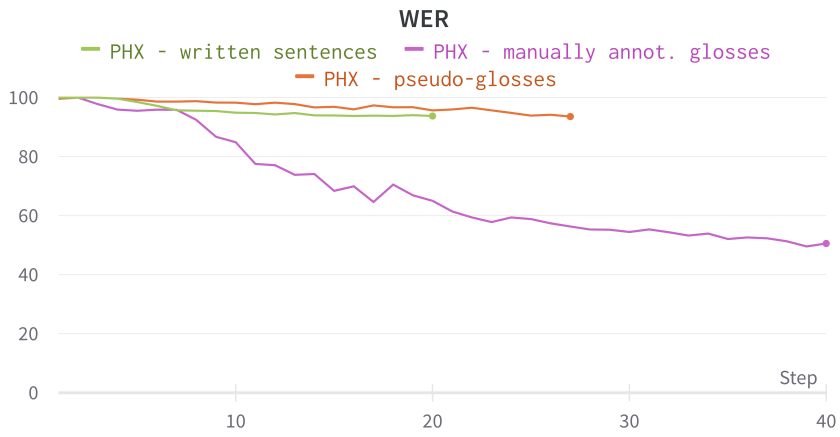


Figure 5: WER metric for validation during training with Signjoey on the different Phoenix corpora based on the annotation glosses.

5.3.2 Results on How2Sign

As mentioned in Section 1, we adapted the architecture to have a trainable model on How2Sign. We obtained the results shown in Table 10. In general, the results were very poor, and even more, if we compare them to the ones obtained on Phoenix. Even so, we provided these baseline results as the first step toward Sign Language Translation on the novel corpus How2Sign with the intention of improving them. And that is what we are doing in this work.

	Validation		Test	
[Recog. and Transl.] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU4 ↑
Baseline model (3L 8H)	99.40	1.75	99.55	1.76
Best results' model (3L 4H)	98.02	2.24	98.40	2.21

Table 10: Results on How2Sign from our previous research work with Signjoey.

As future lines of work, we considered several options. Two of them were obtaining glosses for the dataset, which we already did by generating them automatically, and retraining the current models from the checkpoints until they converged. These two guidelines have led us in the first part of our research working with Signjoey.

Hence, we first figured out how to take advantage of the already implemented modules in the environment to retrain the models from the checkpoints.

Similar to what we did with Phoenix, we decided to train our experiments with different versions of the glosses: with the English written sentences and with the pseudo-glosses, generated in this work and explained in Section 4. Besides, we also trained the model with the translation task only. And, of course, we did not make use of the manual glosses, as they are not currently available for this dataset.

Apart from having different gloss annotations, we also proposed to study the impact of normalizing the written sentences. Therefore, we will show the experiment's results on the corpus with raw and processed text.

Raw text.

On the one hand, in the case of the raw text, we can observe in Table 11 that all experiments overcome the previous results, as they were trained until convergence. Besides, it shows that the best translation performance by far is achieved with pseudo-glosses. This confirms two of our assumptions: the obvious one, that training a model until convergence improves its performance, and the second, that taking advantage of glosses (even though they are not manually annotated) actually helps to learn the task. This statement was first introduced by Camgoz *et al.*, and our work was to prove it for a different corpus, How2Sign.

	Validation		Test	
[Recog. and Transl.] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU4 ↑
Only Translation task	-	3.87	-	3.31
English written sentences.	**97.69**	3.65	97.82	3.15
Automatic pseudo-glosses	96.98	**4.07**	**97.06**	**3.81**

Table 11: Comparison of our results on How2Sign with Signjoey with raw sentences when modifying the gloss annotations.

Additionally, in Figures 6 and 7 we can observe the evolution during the training of the experiments of the BLEU-4 and the WER metrics, respectively. The first one shows a very similar behavior of the three experiments and a better performance of the model with pseudo-glosses. Nevertheless, in the second one, we can see that the experiment with written sentences reaches better results for the metric. Still, the gap between both experiments is minimum (around 1).

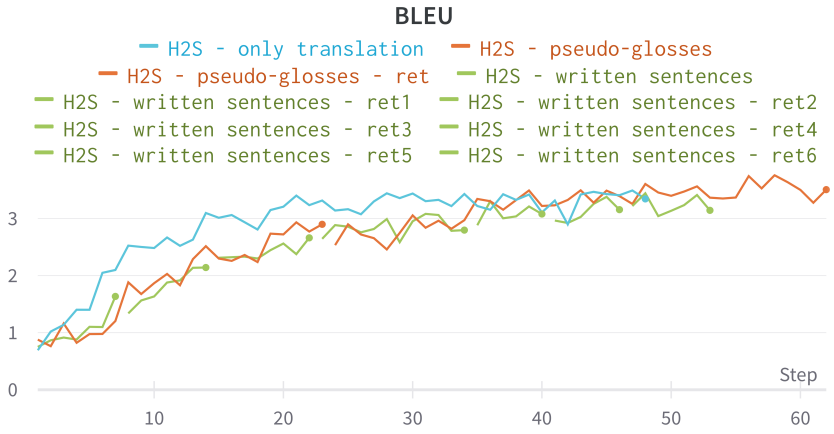


Figure 6: BLEU-4 metric for validation during training with Signjoey on the different How2Sign corpora with raw text based on the annotation glosses.

Although the good news of improving previous performance, we can observe some problems with these results. First, the translation results are not significant, that is to say, the output generated sequences by the model do not maintain the meaning of the input sentences. For example, Table 12 shows the predicted output by the model with the best performance until now (with the pseudo-glosses) and its reference sentence and they have nothing to do with each other. One of the theories

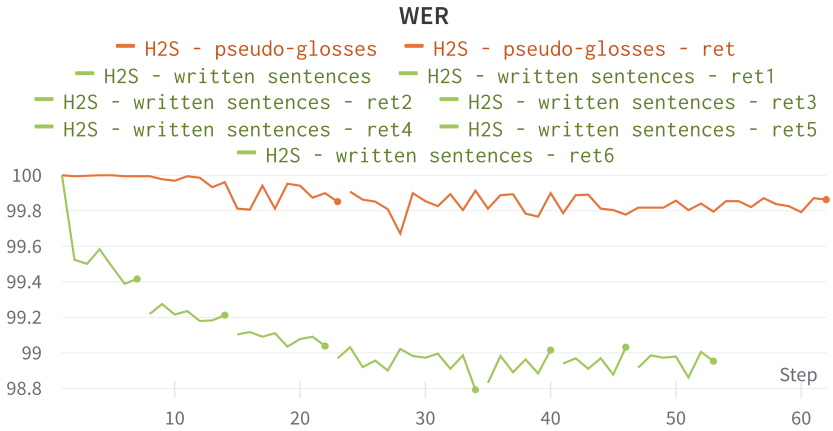


Figure 7: WER metric for validation during training with Signjoey on the different How2Sign corpora with raw text based on the annotation glosses.

we have on this outcome is the magnitude of the solution space. That is to say, the vocabulary size is too big. For this reason, we proposed to replace the raw text with the normalized one.

Text reference	this is a little device i found, it makes it easier to keep in the blind to make it stand up.
Text hypothesis	this is a little bit more than a french bread to make sure you add the sauce add some sauce add the sauce to it.

Table 12: Example of a generated sequence with the Signjoey model trained on How2Sign with pseudo-glosses.

Besides, another issue with these results is the huge time consumption to train a model. For example, in the aforementioned figures, the experiment with written sentences took six retrainings from the checkpoints to converge. That is to say, almost 168 hours. We consider it is too costly to handle. The reason behind this problem is the CTC beam search decoder used for the glosses (and that is why the experiment with only translation took less than 24 hours to finish). This module is implemented in TensorFlow and is extremely time-consuming, especially when the vocabulary size (of the glosses) is large.

Therefore, as all the pieces of evidence indicated we should reduce the solution space, we now present the results on How2Sign with the normalized text.

Pre-processed text.

On the other hand, in the case of the pre-processed text, we can observe the results in Table 13. In general, they considerably outperform the translation results obtained with the raw text in every kind of experiment. Again, the model that uses the pseudo-glosses achieves the highest performance compared to the others. Additionally, contrary to the previous case, the use of written sentences improves the BLEU-4 scores. However, even though the WER metric improves respective to previous results, there is no such a significant enhancement.

Moreover, Figures 8 and 9 present the progress of the BLEU-4 and WER metrics during the epochs, respectively. In the first figure, we can observe that the experiment with English sentences as annotations seems to outperform the results obtained with the pseudo-glosses, although at inference time the latter mentioned achieves higher scores for validation and test. Besides, the experiment that

	Validation		Test	
[Recog. and Transl.] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU4 ↑
Only Translation task	-	5.44	-	4.99
English written sentences.	**95.25**	6.00	**95.74**	5.53
Automatic pseudo-glosses	96.53	**6.14**	96.68	**5.67**

Table 13: Comparison of our results on How2Sign with Signjoey with pre-processed sentences when modifying the gloss annotations.

only trains the translation task also reaches similar results. In the second figure, the model with written sentences performs better than the one that uses pseudo-glosses. Yet again, the difference is almost irrelevant.

Furthermore, it is important to remark that now the models took much less time to train than with raw text: now, instead of taking seven retrainings to converge, the experiment with English sentences took only two retrainings. This outcome is also relevant, as we have solved a major problem.

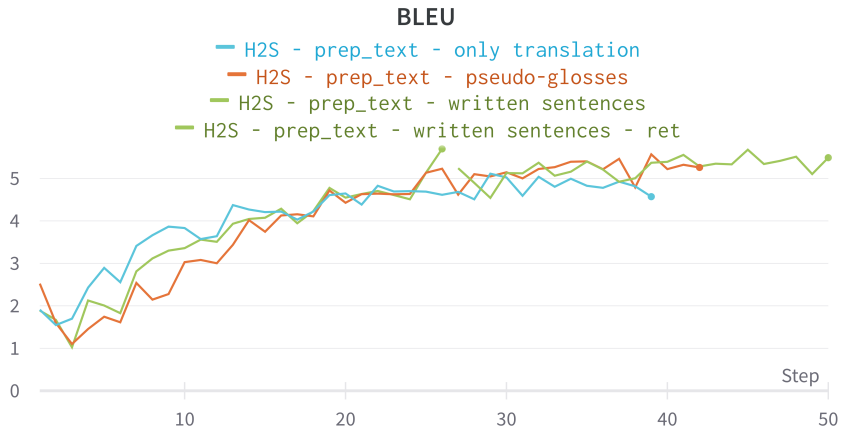


Figure 8: BLEU-4 metric for validation during training with Signjoey on the different How2Sign corpora with processed text based on the annotation glosses.

In conclusion, the usage of normalized text outmatches the previous experiments' results and provides novel baseline results on the Sign Language Translation task. Besides, another reason for this improvement has been the usage of the pseudo-glosses as annotations, compared to the usage of written sentences introduced in our previous research work.

Nevertheless, the current output sequences are still not suitable for translation. For example, in Table 14 we can observe two examples of these generated sequences by the model with pseudo-glosses, the one that achieves better performance until now. They do not share the same meaning with their reference.

The quality of the model is not good enough to be used in a real Sign Language Translation task. Hence, we will provide the results of our proposed solution to improve the current performance: a novel implementation of the Sign Language Transformer model [3] in the Fairseq toolkit.

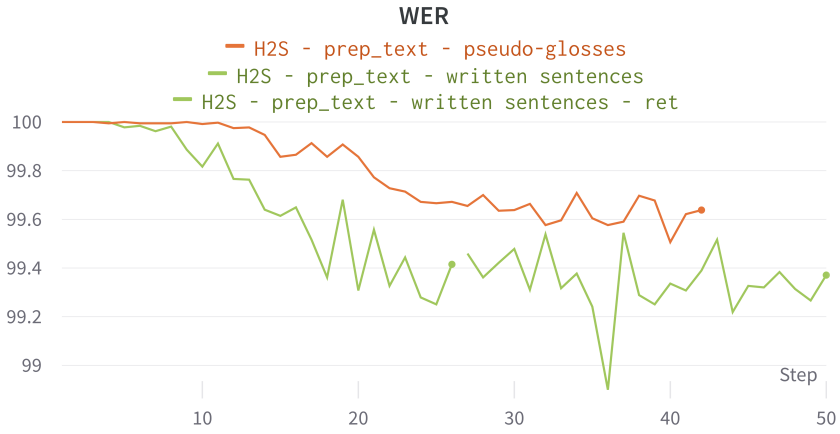


Figure 9: WER metric for validation during training with Signjoey on the different How2Sign corpora with processed text based on the annotation glosses.

Text reference	and this is how some repair tips for setting up your tent
Text hypothesis	this is how you repair a soccer tent
Text reference	jonesy is going to sit back
Text hypothesis	so this is a really tricky part

Table 14: Examples of two generated sequences with the Signjoey model trained on How2Sign.

5.4 Results on Fairseq

Following, we will present the results obtained with our implementation of the model in Fairseq on Phoenix and How2sign.

As mentioned in the implementation details (Section 5.2), the vocabulary used in the predictions is created in the pre-processing step, before training the model, and the method used allows us to choose the tokenization type for the words. Therefore, in this part of the work, not only are we assessing the impact of the different versions of the glosses, but we are also evaluating the usage of two different dictionary types: with words and with subwords (using unigrams).

5.4.1 Results on Phoenix

First, we will introduce the results with the dictionary of words, as it is the same configuration used in the Sign Language Transformers work [3], and then with subwords. After, we will compare and conclude which approach is more appropriate for this setup.

Dictionary of words.

On the one hand, in Table 15 we can observe the performance of the experiments on Phoenix. There are many aspects to comment on this outcome.

On the first hand, it is clearly obvious that the experiment that performs only translation achieves the best performance. Besides, the gap with the others in the BLEU-4 score is higher than 3 for both sets. Furthermore, these results not only improve our previous results on the Signjoey implementation,

but they even outperform the results provided by Camgoz *et al.* in their work [3] shown in Table 8. We will discuss it in more detail later on in Section 5.5. On the second hand, the experiments with manual glosses and pseudo-glosses show similar performance in the BLEU-4 score, however, in the WER metric there is a large gap. This behavior is similar to the previous one obtained, so it resulted as expected. Finally, it is very remarkable that the performance of the experiment with German sentences as glosses obtained a BLEU-4 score of half that of the others. This may indicate that replacing the annotations with the written sentences in this implementation penalizes the translation task instead of helping guide the training in the correct direction.

	Validation		Test	
[Recog. and Transl] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU-4 ↑
Only Translation task	-	**23.29**	-	**24.82**
German written sentences	76.81	9.55	77.47	8.74
Automatic Pseudo-glosses	65.75	18.68	66.15	19.29
Manual gloss annotations	**39.04**	19.74	**39.4**	21.28

Table 15: Comparison of our results on Phoenix with Fairseq when modifying the gloss annotations with dictionary of words.

Additionally, Figures 10 and 11 show the evolution during the training of the BLEU-4 and WER metrics, respectively. In the first one, we can observe what we stated before: first, the experiment with only the translation tasks achieves higher performance by far compared to the others, and second, the experiment with the written sentences as glosses gets poor results. Besides, the other two reach the expected BLEU scores (around 19). In the second figure, we can still see the difference in the recognition results between using the manual glosses and their replacements: the performance is evident to be far worse for the two substitutes than for the original. And, between the two of them, the evolution is better in the experiment that uses pseudo-glosses.

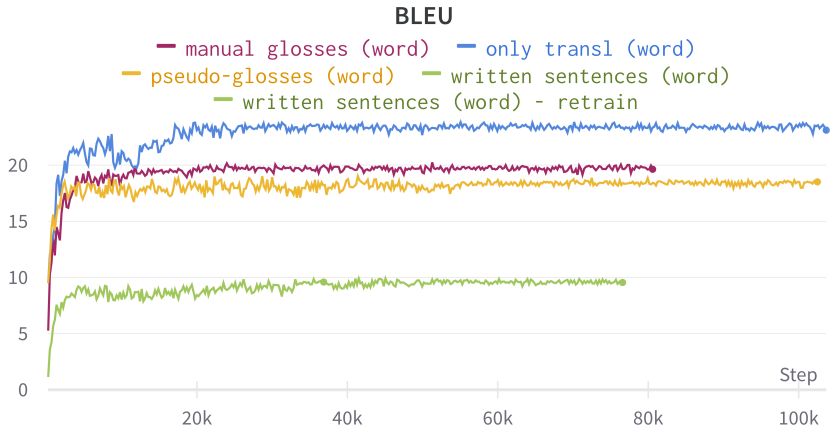


Figure 10: BLEU-4 metric for validation during training with Fairseq on the different Phoenix corpora based on the annotation glosses with dictionary of words.

Dictionary of subwords (unigram).

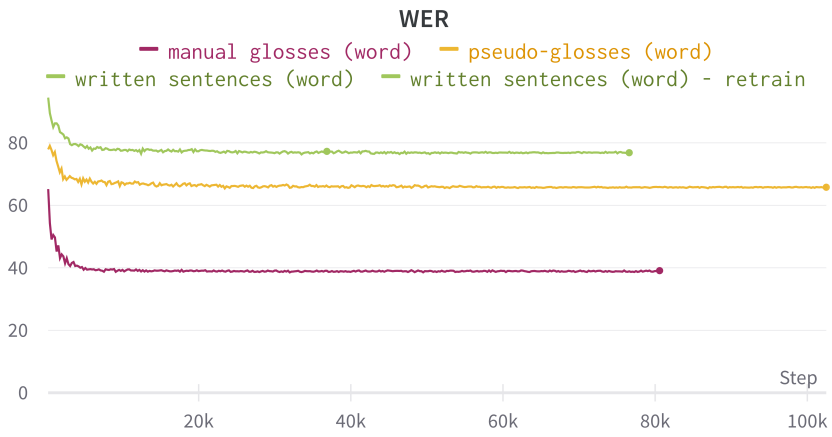


Figure 11: WER metric for validation during training with Fairseq on the different Phoenix corpora based on the annotation glosses with dictionary of words.

On the second hand, we can observe the results of the experiments using a dictionary of subwords in Table 16. Remarkably, the experiment that uses pseudo-glosses achieves a higher performance in translation, improving even the results obtained with the dictionary of words. On the contrary, the other experiments have worsened their performance compared to the previous results. However, excluding the case of the experiment with pseudo-glosses, the rest of the configurations maintain the order of the performance achieved previously.

	Validation		Test	
[Recog. and Transl] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU-4 ↑
Only Translation task	-	20.04	-	18.02
German written sentences	79.45	7.13	80.12	6.9
Automatic Pseudo-glosses	72.72	**21.64**	72.70	**22.32**
Manual gloss annotations	**41.22**	17.23	**41.23**	17.18

Table 16: Comparison of our results on Phoenix with Fairseq when modifying the gloss annotations with dictionary of subwords.

Besides, in Figure 12 we observe that the evolution of the BLEU-4 score for each experiment confirms what we just explained. It is remarkable the gap between the results of the experiment that uses the written sentences and the others, which already showed poor performance in the experiment with the words' dictionary. There, we commented that using this approach (written sentences as a replacement for the annotations) could penalize the learning of the translation task and, with this other model, we are observing the same behavior.

Moreover, Figure 13 displays the progress of the WER metric during the training. There, we can see that the experiment with manual glosses gets similar results as with the rest of the approaches and achieves the best scores compared to the experiments that use replacements instead of annotations.

Comparison between words and subwords.

Following, we will compare the two approaches used to create the dictionaries: words and subwords

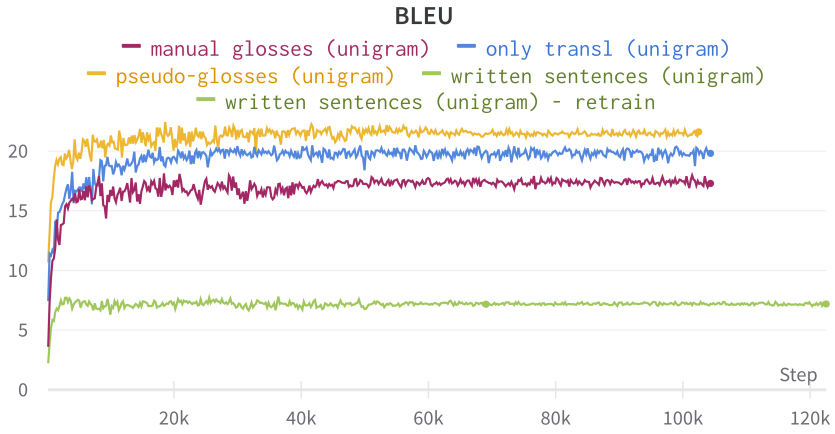


Figure 12: BLEU-4 metric for validation during training with Fairseq on the different Phoenix corpora based on the annotation glosses with dictionary of subwords.

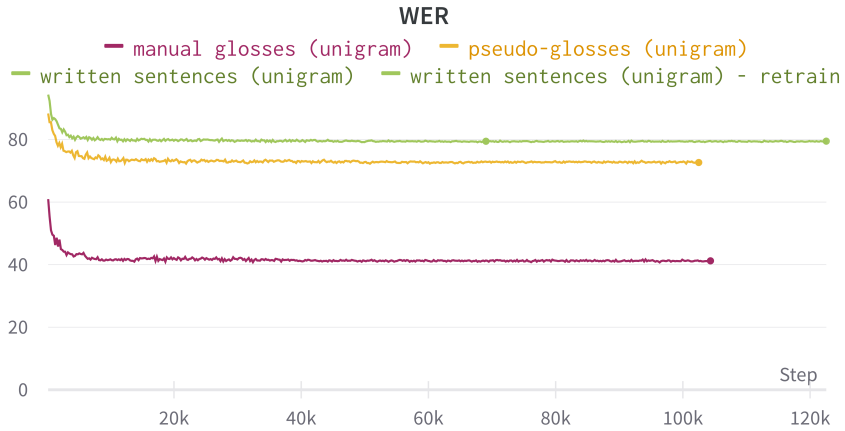


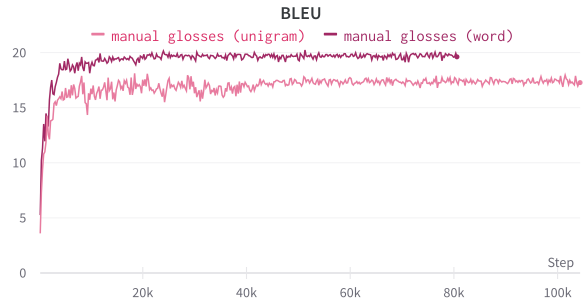
Figure 13: WER metric for validation during training with Fairseq on the different Phoenix corpora based on the annotation glosses with dictionary of subwords.

(with the unigram method).

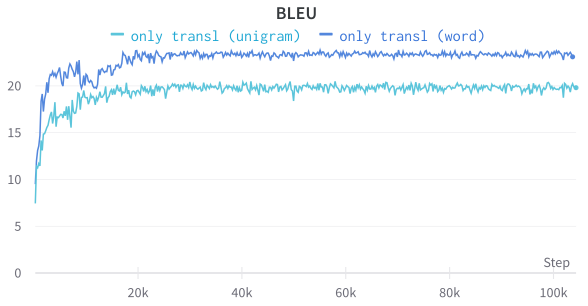
In Figure 14 we can observe the difference in the performance for each experiment when using words and unigrams. In general, the models that use the the first approach obtain much better results (in the subfigures, the darker lines). Nevertheless, we have to exclude the case of the pseudo-glosses, where the experiment with subwords (in yellow) outperform the other (in orange).

Hence, we can conclude that in the case of the Phoenix corpus, using a dictionary of words (and not splitting them) is the best approach for the translation task. We hypothesize that this might be because it compacts the number of words used throughout the dataset. However, there is still one case (the experiment with pseudo-glosses) that does not follow this behavior.

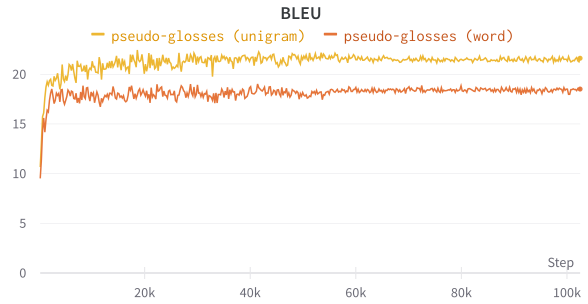
Outstanding results.



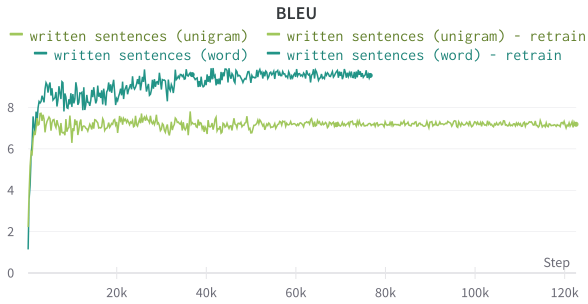
(a) Experiment with manual glosses as annotations.



(b) Experiment of only translation task.



(c) Experiment with pseudo-glosses as annotations.



(d) Experiment with written sentences as annotations.

Figure 14: Comparative of the BLEU-4 metric trained on Fairseq of the four types of experiments on Phoenix with dictionary of words vs dictionary of subwords (unigram).

As we observed above, the experiment that performs only translation with a dictionary of words achieves the best results. Not only that, but it even outperforms the baseline results introduced by Camgoz *et al.* in their work [3] when training jointly the recognition and translation tasks. That is to say, with this experiment we introduce new baseline results in the Sign Language Translation task for this dataset.

Furthermore, we have repeated the training of the experiment with the same seed (1) and two more (2, 3) in order to ensure the results are consistent. They can be found in Table 17. In general, the four executions achieve similar performance and the average result in the last row is very close to all the results. Moreover, Figure 15 shows that the evolution of the BLEU-4 score for the different executions is almost identical. Therefore, all these evidences give solidity to our statement.

Therefore, we have also accomplished one of our main goals, which was to outperform the current results on the translation task with the new implementation based on Fairseq.

Additionally, in Table 18 we can observe examples of sequences generated by the mentioned model. The predicted sentences are almost identical to their references, which confirms the outstanding performance of the architecture.

5.4.2 Results on How2Sign

Following, we will present the results obtained on How2Sign. Again, we will begin with the experiments made with dictionaries of words, the same approach as in the SLT work [3], and then we will show

	Validation	Test
Translation Tasks	BLEU-4 ↑	BLEU-4 ↑
Original (first model)	23.29	24.82
Model with seed=1	**23.64**	23.5
Model with seed=2	22.3	**25.51**
Model with seed=3	23.14	23.49
Average	23.09	24.33

Table 17: Comparison of our results on Phoenix with Fairseq with the experiment that performs only translation when modifying the seed with dictionary of words.

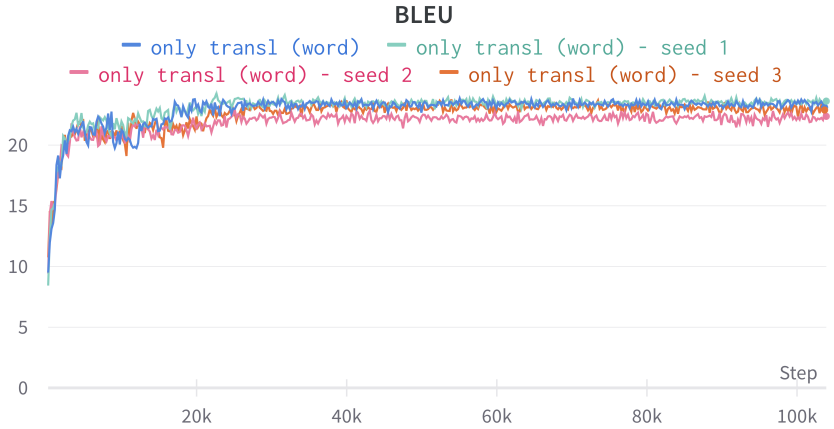


Figure 15: BLEU-4 metric for validation during training with Fairseq on Phoenix for the only translation experiment with different seeds.

Text reference	da mehr wieder samstag
Text hypothesis	da mehr wieder samstag
Text reference	das? morgen westen zieht nun mitteleuropa himmel
Text hypothesis	das? morgen westen nun mitteleuropa himmel
Text reference	mal gewitter und??? wetter in wird am in
Text hypothesis	mal und gewitter und wird? in

Table 18: Examples of three generated sequences with the Fairseq model trained on Phoenix.

the experiments with subwords.

Furthermore, as in Section 5.3.2 we already proved that using the processed sentences from How2Sign improved significantly the models' performance, we will not train the models on the raw sentences, because we considered it irrelevant.

It is important to remark that for these experiments we do not show the evolution curve of the WER metric during the training. It is because, apart from being extremely expensive (it would take several days to train each model), we do not have the manual annotations available for this corpus, so it would not be a real recognition task. Therefore, we only computed the sequences of English

sentences and computed the BLEU-4 score during the training. Even so, we generated the gloss predictions at inference time, in order to be able to compare the results with the previous from the other implementation.

Dictionary of words.

On the one hand, we can observe in Table 20 the result for the experiments on How2Sign with a dictionary of words. In general, they are very similar in their performance, both in recognition and translation. In this case, the experiment with better results is the one that uses the written sentences as glosses.

Besides, Figure 16 shows that the evolution is very similar in the three cases, which confirms what we mentioned before.

	Validation		Test	
[Recog. and Transl] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU-4 ↑
Only Translation task	-	5.06	-	4.31
English written sentences	91.32	**5.15**	92.35	**4.64**
Automatic Pseudo-glosses	**90.68**	5.04	**91.59**	4.22

Table 19: Comparison of our results on How2sign with Fairseq when modifying the gloss annotations with dictionary of words.

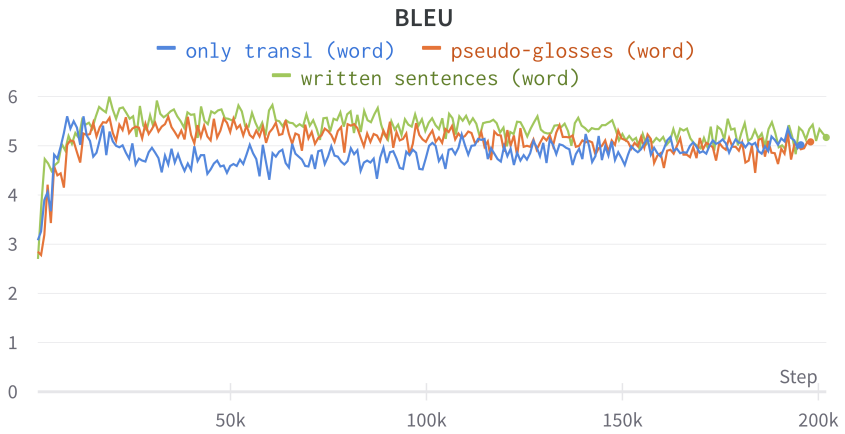


Figure 16: BLEU-4 metric for validation during training with Fairseq on the different How2Sign corpora based on the annotation glosses with dictionary of words.

Dictionary of subwords (unigram).

On the other hand, we can see in Table 20 the performance of the experiments made with dictionaries of subwords (with the unigram approach). We observe two behaviors. First, the case of the experiment that performs only translation and the one that uses the pseudo-glosses, which achieve significantly good results. And, second, the experiment that uses English sentences, which performs poorly. Again, using written sentences worsens the results instead of helping to *supervise* the translation task. It is interesting, though, that the recognition result is better in this experiment than with the other (which reaches higher a BLEU-4 score). Besides, these results outperform the previous

obtained on this corpus. We will discuss this outcome later. Moreover, in Figure 17 we can observe what we already commented.

	Validation		Test	
[Recog. and Transl] Tasks	WER ↓	BLEU-4 ↑	WER ↓	BLEU-4 ↑
Only Translation task	-	**13.81**	-	**13.63**
English written sentences	**88.96**	5.46	**85.4**	4.52
Automatic Pseudo-glosses	98.64	13.73	98.82	13.4

Table 20: Comparison of our results on How2sign with Fairseq when modifying the gloss annotations with dictionary of subword.

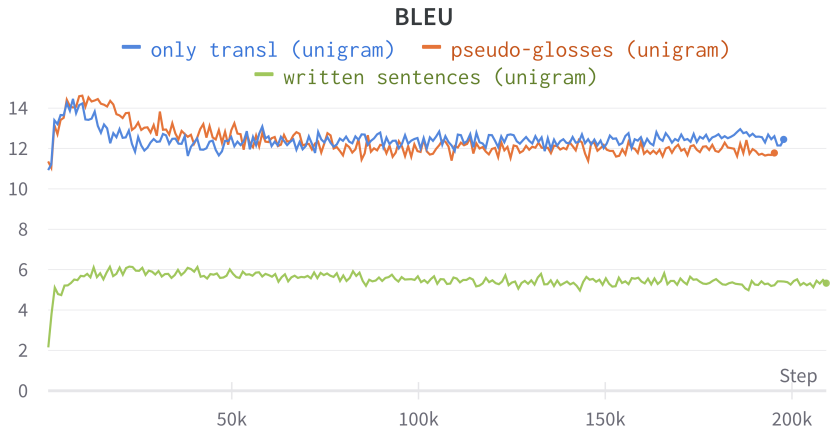
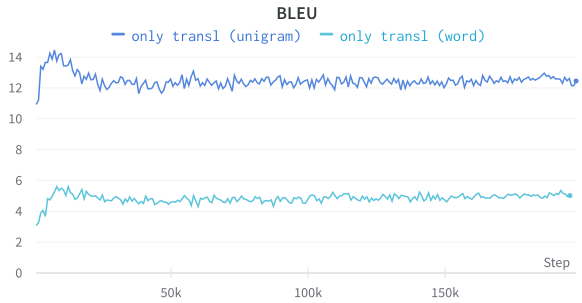


Figure 17: BLEU-4 metric for validation during training with Fairseq on the different How2Sign corpora based on the annotation glosses with dictionary of subwords.

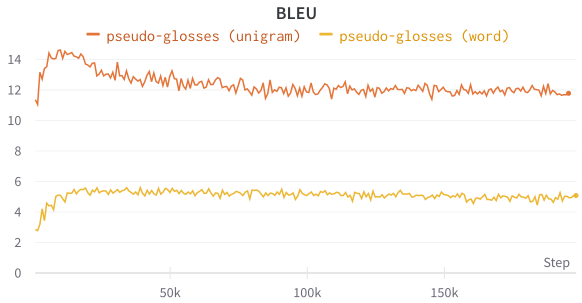
Comparison between words and subwords.

Similar to what we did with Phoenix, we will compare the difference between using dictionaries built with words and with subwords. In Figure 18 we have three subfigures that display the evolution of the performance for each experiment when using words and subwords. Even though in the case of the experiment with written sentences (Subfigure 18c the performance is almost identical for both approaches, using subwords to create the vocabulary is a better choice for this dataset. It can be clearly observed in Subfigures 18a and 18b, where the gap between both experiments is very significant.

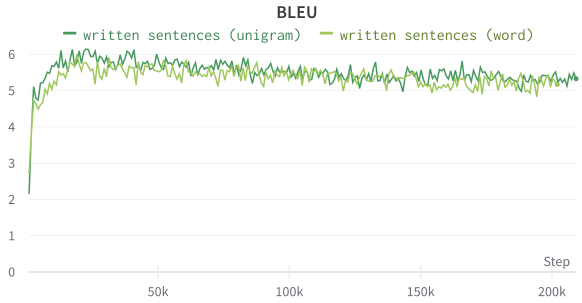
In general, for the How2Sign corpus, building the dictionary with subwords means achieving much better translation performance. These results differ from the ones obtained on Phoenix, where the vocabulary formed by words obtained better results. This may have several reasons. We hypothesize that the number of unique words in the How2Sign written sentences is considerably large and, as a consequence, the translation task is very difficult. Then, by using a subword approach, we are reducing significantly the solution space: in this case, by half (from 15030 words to 7500). Hence, this results in an important improvement in the performance of the models. Besides, it can also be due to the characteristics of the English language, in which most of its lexicon is formed by a root added to prefixes and suffixes and, for this reason, these words can be easily broken into subword units.



(a) Experiment of only translation task.



(b) Experiment with pseudo-glosses as annotations.



(c) Experiment with written sentences as annotations.

Figure 18: Comparative of the BLEU-4 metric trained on Fairseq of the four types of experiments on How2Sign with dictionary of words vs dictionary of subwords (unigram).

Besides, the results obtained with the dictionary of subwords (both the experiment with only translation and with pseudo-glosses) outperform all the previous results on the task for this dataset. This means that we have succeeded in our goals of providing new baseline results for this corpus.

In addition, in Table 21 we can observe examples of generated sequences by the model trained only on the translation task with a dictionary of subwords, which is the one that obtained higher BLEU-4 scores. Unfortunately, in general, the sentences do not coincide with their references. They do share some words, mainly the most used terms in English such as pronouns (you) or prepositions (the). However, it is more difficult for the model to recognize the important terms that carry the meaning of the sentence. Still, some of these relevant words have been identified by the model: for example, the words *bit* or *face* in the last two examples. Besides, the first example shows a perfect translation, which is of course not common, but we wanted to remark that in some cases the model performs correctly.

5.5 Comparison

After introducing all the results on Phoenix and How2Sign obtained with both implementations, Signjoey and Fairseq, we will now compare them and discuss which model is more suitable for the task.

On the one hand, for the Phoenix dataset, we can observe that the performance obtained with Signjoey (Table 9) and with Fairseq with a dictionary of words (Table 15) have very similar results for

Text reference	you the then you theed best and talkings
Text hypothesis	you the then you theed best and talkings
Text reference	you the a not to the bit m to the right
Text hypothesis	you the that and to the bit to the gets
Text reference	you the that to the school these in have sort do to the face yellow is to the especially
Text hypothesis	to the thinking these you the are your to the face clay and to the face clay

Table 21: Examples of three generated sequences with the Fairseq model trained on How2Sign.

the experiments. The exception is the case of the experiment with written sentences as annotations. Besides, the experiment that performs only translation achieved better results than any other model and, therefore, we provide new baseline results on the Sign Language Translation task for this dataset.

On the other hand, for the How2Sign dataset with the pre-processed sentences, we can see that the performance obtained with Signjoey (Table 13) is slightly better than the one achieved with the same configuration with Fairseq (Table 19). However, we have introduced a new approach for generating the vocabulary (with subwords) and, with it, we have achieved much better performance for this dataset (20). Therefore, we also provide novel baseline results for How2Sign on the task that outperforms all the previous ones. Again the experiment that achieves the higher BLEU-4 results is the one that performs only the translation task.

6. Conclusions

In this work we make a research investigation in the Sign Language Translation field. More precisely, we have achieved the goals we presented for this project. First, we present an approach to automatically collect annotations from the written sentences. Then, we introduce a novel implementation to address the jointly learning of recognition and translation based on the Camgoz *et al.*'s work [3]. And, finally, we provide baseline results, as well as new experiments, on the Sign Language Translation task on two corpora in this field, Phoenix and How2Sign.

First, we present a novel method to generate automatic pseudo-glosses from the sentences as a replacement for the Sign Language annotations. This solution comes from the large cost of annotating them manually and the current necessity of collecting them for Sign Language tasks.

Furthermore, we introduce a novel implementation built on the Fairseq framework based on the Sign Language Transformer architecture [3] built on JoeyNMT (named Signjoey). This model is based on the idea of learning jointly the recognition and the translation tasks, having the first one as a *supervision* to help the Encoder part learn the meaning of the sign videos to improve the performance of the second, which is the end goal of this work. This implementation has proved to achieve similar results on the tasks as well as the previous SLT architecture introduced by Camgoz *et al.* and, in some cases, even better results.

In the results part, on the one hand we have trained our models on the Phoenix parallel corpus. Apart from executing the same experiments introduced in the work made by Camgoz *et al.*, we have also presented two new ones: replacing the gloss annotations with automatic pseudo-glosses and with the written sentences from the spoken language. This way, we tackle the fact that having manual annotations is not realistic for all the datasets. Therefore, we provide new results on these experiments with the Signjoey architecture and then with our novel implementation based on Fairseq. With the second, we show that building the vocabulary with words is a better approach than using subwords. And, finally, we introduce new baseline results on the Sign Language Translation task with our implementation with the experiment that only performs translation, without any intermediate assistance of the recognition task. We show examples of the sentences generated by this model, which prove its excellent performance.

On the other hand, we have trained our models on the How2Sign parallel corpus. In this case we also executed the same experiments as with the previous dataset, except that in this case the manual annotations are not currently available. First, we improve the results from the previous research made by this team, by training the models until convergence with the Signjoey architecture. Besides, we present the results obtained with our implementation of the model based on Fairseq, where we prove that using a subword approach to create the dictionary is a much better choice than words. This is because the solution space of this dataset is significantly large and by the particularities of the English language, we can reduce its vocabulary by half by dividing the words. Hence, we provide new baseline results for this corpus on the translation task with the experiment that only performs the translation task, without the supervision of annotations, although the experiment that used pseudo-glosses achieved very similar performance. Even so, we show that the sequences are not translated perfectly and the important meaning of the sentences is not perfectly learned by the model. Therefore, there is yet a lot of work to do in this field. Still, this investigation introduces a good starting point to further research.

Furthermore, we observe in the results that using automatic pseudo-glosses actually achieves very

similar performance than using the original manual glosses in the Phoenix dataset case. In general, this replacement approach got much better results for both corpora than using the written sentences, especially in our implementation of the architecture. Hence, we give evidences that replacing the glosses, with its high cost of annotating, and using automatic pseudo-glosses is a suitable approach.

Finally, we can observe that the best models we have trained do not make use of the recognition task as a *supervisor* to assist during the training. For this reason, we introduce the question whether it is necessary to use the gloss annotations to extract the meaning of the sign videos. It is important to consider this option, as the glosses are not collected for free, they come with a high annotation cost. At least for the translation task, we have shown that we can obtain similar, and even better, performance when training without this intermediate *assistance*. Then, it would be interesting to continue researching in this direction in order to determine the impact of using recognition for the end goal of the translation task and, as we are hypothesizing, glosses are not what we need to reach our goal.

References

[1] Anne Baker, Beppie van den Bogaerde, Roland Pfau, and G. M. Schermer. The linguistics of sign languages. *John Benjamins Publishing Company*, 2016.

[2] 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.

[3] 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.

[4] 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.

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

[6] 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, 2021. https://how2sign.github.io/.

[7] 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.

[8] Alex Graves, Santiago Fernández, Faustino Gomez, , and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. *Proceedings of the ACM International Con- ference on Machine Learning (ICML)*, 2006.

[9] Diederik P. Kingma and Jimmy Ba. Adam. A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[10] 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.

[11] 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.

[12] Julia Kreutzer, Jasmijn Bastings, and Stefan Riezler. Joey nmt: A minimalist nmt toolkit for novices. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*, 2019.

[13] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

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

[15] Achraf Othman and Zouhour Tmar. English-asl gloss parallel corpus 2012: Aslg-pc12, the second release. *Fourth International Conference On Information and Communication Technology and Accessibility ICTA'13, Hammamet, Tunisia*, 2013.

[16] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[17] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[18] 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.

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

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

[21] 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.

[22] 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.

[23] Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. fairseq s2t: Fast speech-to-text modeling with fairseq. In *Proceedings of the 2020 Conference of the Asian Chapter of the Association for Computational Linguistics (AACL): System Demonstrations*, 2020.

[24] Kayo Yin and Jesse Read. Better sign language translation with stmc-transformer. *In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain (Online). International Committee on Computational Linguistics*, 2020.